HTTP Server Planning, Installing, and Using

Version 5.2
Contents

Summary of Changes for IBM HTTP Server Version 5.2 for OS/390  xi
SC31-8690-04 for OS/390 Releases 8 and 9  xi
Certificate Authority (CA) utility  xi
Configuration directives  xi
IBM Key Management Utility (IKEYMAN)  xii
OS/390 Debug Tool for C/C++ GWAPI programs  xii
SMF performance records  xii
SSI environment variables  xii
Web server start options  xii
SC31-8690-03 for OS/390 Release 8  xii
Configuration directives  xii
Messages  xiii
Web server activity and performance statistics  xiii
SC31-8690-02 for OS/390 Release 8  xiii
Configuration directives  xiii
Error recovery and problem determination  xiv
Logging and reporting  xiv
Messages  xv
Programming information  xv
Updated and improved sections  xvi

Welcome!  xvii
Easy-to-use graphical interface  xvii
Comprehensive product information  xvii
Integrated user assistance  xvii
Books  xvii
Support services  xviii

Part 1. Planning  1

Chapter 1. Planning for installation  3
Summary of Changes  3
Requirements and planning considerations  3
ABEND recovery performed by the Web server  3
Access control with RACF and other SAF-based security products  4
Browser requirements  4
Fast Response Cache Accelerator  6
MVSDS DLL OS/390 data set considerations  7
OS/390 UNIX System Services considerations  7
Performance considerations  8
Protection setup considerations  8
Reporting programs used by the Web server  9
Security considerations  9
WebSphere Application Server Java-based servlet engine  9
Workload Management considerations  9
APARs and service updates  10

Chapter 2. Migration considerations  11
Accessory script directives  11
IMWHTTPD program  11
Java-based servlet engine  12
Proxy server and caching proxy server  12
Security  13
Configuration and key management changes  13
Certificate authority utility changes  13
Server IP Address  14

Part 2. Installing  15

Chapter 3. Installing your secure server  17
Before you install  17
Review planning information  17
Check for information updates and hints/tips on the Web  17
Perform OS/390 installation procedure  17
Install the Java Development Kit  18
Completing and customizing your Web server installation  18
Step 1. Create the OS/390 UNIX System Services group ID and the Web server user ID  18
Step 2. Create the OS/390 UNIX System Services user ID  19
Step 3. Permit SMF to RACF  19
Step 4. Turn on program control for RACF  19
Step 5. Review access control user ID options and create surrogate user IDs  20
Step 6. Activate the Lotus Notes Adapter function and HTTPD_local_security function (optional)  22
Step 7. Enable the Web server to support OS/390 Workload Management (optional)  22
Step 8. Make TCP/IP configuration adjustments as needed  22
Step 9. Copy and customize the Web server PROC  23
Step 10. Configure installed files by running setup.sh  24
Step 11. Customize your Web server configuration file httpd.conf.  27
Step 12. Set up security  27
Step 13. Set up proxy server support (optional)  27
Step 14. Install and configure WebSphere Application Server for servlet support (optional)  28
What’s next?  28

Part 3. Using  29

Chapter 4. Starting the server  31
Performance considerations when starting the Web server  31
Starting the server from the start-up PROC  31
Starting the server using the httpd command  31
Starting the server from the OS/390 UNIX shell  31
Starting multiple instances of the server  32
Starting the server with Workload Management running  .................................................. 32  
Letting WLM start the queue server address space ................................................................. 32  
Prestarting the queue server address space ............................................................................. 33  
Restarting the server .................................................................................................................. 33  

Chapter 5. Viewing the Web server’s default Front Page ......................................................... 35  

Chapter 6. Stopping the server .................................................................................................. 37  

Part 4. Basic Configuration ....................................................................................................... 39  

Chapter 7. Getting started ........................................................................................................ 41  
How do I start serving pages? ....................................................................................................... 41  
How do I serve directory listings? ............................................................................................... 41  
How do I configure the server? .................................................................................................... 42  
Using the Configuration and Administration forms ............................................................... 42  
Editing the configuration file ..................................................................................................... 43  
How do I control access to server administration functions? .................................................. 44  
Controlling access to the Configuration and Administration forms ...................................... 44  
Copying files .............................................................................................................................. 44  
What files should I back up? ....................................................................................................... 44  

Chapter 8. Setting up a secure server ....................................................................................... 45  
Before you begin ........................................................................................................................ 45  
Tips and frequently asked questions ......................................................................................... 46  
IBM Key Management Utility .................................................................................................. 46  
HTTP Server CA utility ............................................................................................................. 46  
Security configuration examples .............................................................................................. 47  
Security concepts, options, and support ................................................................................... 47  
Security concepts ....................................................................................................................... 47  
What is a secure communication? .............................................................................................. 47  
What is encryption? .................................................................................................................... 47  
What is authentication? .............................................................................................................. 48  
What is a Public Key Infrastructure? .......................................................................................... 49  
What is the Secure Sockets Layer protocol? ............................................................................ 50  
Security options for the HTTP Server ..................................................................................... 51  
Options for protecting server resources .................................................................................. 51  
Options for setting up secure communications ........................................................................ 51  
SSL support for multiple IP addresses ..................................................................................... 52  
Support for specifying the encryption level to be used ............................................................. 52  
Stronger encryption option for financial and banking Web servers ....................................... 53  
Certificate authorities supported by the HTTP Server ............................................................ 54  
Buying a certificate from an external CA provider .................................................................... 54  
Acting as your own CA .............................................................................................................. 55  
Encryption support for the HTTP Server .................................................................................. 55  
Supported public-private key sizes ........................................................................................... 56  
Supported SSL cipher specifications ....................................................................................... 56  
Hardware encryption ................................................................................................................ 57  
Setting up access control and secure connections .................................................................... 57  
Accessing the online help examples ......................................................................................... 58  
Setting up protection for server resources .............................................................................. 58  
Setting up secure connections using an external CA ............................................................... 61  
Setting up secure connections using self-signed certificates .................................................. 62  
Setting up SSL support for multiple IP addresses ................................................................... 65  
Changing the server’s default encryption settings ................................................................... 65  

Part 5. Advanced Configuration ............................................................................................... 67  

Chapter 9. Customizing cache management with the Fast Response Cache Accelerator ......... 69  
Overview ................................................................................................................................... 69  
Planning considerations ............................................................................................................. 69  
Access to WLM ........................................................................................................................... 69  
Unique subsystem name for WLM ............................................................................................. 69  
TCP/IP stack name ..................................................................................................................... 69  
Enabling and configuring the Cache Accelerator ...................................................................... 69  
Monitoring and managing the Cache Accelerator .................................................................... 70  
RMF reports ............................................................................................................................... 70  
DISPLAY TCPIP command ....................................................................................................... 70  

Chapter 10. Customizing logs and reports ............................................................................. 71  
Tailoring the logs your server keeps ............................................................................................ 71  
Overview of log types ................................................................................................................... 71  
Overview of log setup options ..................................................................................................... 73  
Sample scenario for configuring log files ..................................................................................... 78  
Tailoring the reports your server creates .................................................................................... 78  
Reporting options and considerations ....................................................................................... 78  
Using the HTLOGREP or Web usage mining reporting programs ......................................... 79  
Creating a report template ......................................................................................................... 81  
Viewing reports .......................................................................................................................... 82  
Sample scenarios for configuring reports .................................................................................... 82  
Using the Web usage mining statistics reports ......................................................................... 85  
Running Web usage mining from the command line ................................................................. 87  
Web usage mining and multiple servers .................................................................................... 88  
Logging information with System Management Facilities ..................................................... 88  

Chapter 11. Customizing your Web site .................................................................................... 89  
Displaying page count, date, time, and text on a Web page .................................................... 89  
Configuration instructions .......................................................................................................... 89  
Samples ....................................................................................................................................... 90  
Options ....................................................................................................................................... 90  
Using server-side includes to insert information into CGI programs and HTML documents .... 93  
Considerations for using server-side includes .......................................................................... 93  
Preparing to use server-side includes ....................................................................................... 94  
Format for server-side includes ................................................................................................. 94  
Directives for server-side includes ............................................................................................ 95  
Using server-side image maps to enable clickable images ...................................................... 101  
Syntax ......................................................................................................................................... 102  
Examples .................................................................................................................................... 102  

iv HTTP Server Planning, Installing, and Using
# Chapter 12. Managing your Web server

- Modes of Operation for the Web server ............................................. 105
  - Standalone Server mode .......................................................... 105
  - Scalable Server mode ............................................................. 106
  - Running Multiple Servers ........................................................ 106
- Workload Management Enablement for the Web server .................... 107
  - Workload management overview .............................................. 107
  - Configuring workload management to support the Web server ...... 107
  - How workload management panels are interpreted .................... 112
  - Configuring your environment for workload management ............ 113
  - Configuring your application environment name for workload management .................................................. 113
- Server Activity Monitor ............................................................. 113
  - Enabling the Server Activity Monitor ...................................... 114
  - Accessing statistics and usage information ................................ 114
  - Web server activity statistics ................................................ 114
  - Network activity statistics .................................................... 117
  - Access log entries ..................................................................... 117
- Simple Network Management Protocol ......................................... 118
  - SNMP commands and protocol ................................................. 118
  - Object IDs and variable names for the HTTP Server MIB .......... 118
  - Creating an e-mail address to receive SNMP problem reports .... 134
  - Providing a security password for SNMP .................................. 134
  - Enabling and disabling SNMP support ..................................... 135
  - Turning the SNMP support on and off from the IMWHTTPD program .................................................. 135
- System Management Facilities ..................................................... 135
  - Turning SMF support on and off from the IMWHTTPD program .... 136
  - Turning SMF on and off with the OS/390 operator console MODIFY command .................................................. 136
  - Controlling the logging of information by SMF ......................... 136
  - SMF software requirements ..................................................... 137
  - SMF record formats .................................................................. 137
  - OS/390 console commands ....................................................... 142

# Chapter 13. Rating Web sites and serving rated Web information ........ 143

- PICS overview ............................................................................. 143
- Who can rate Web sites ............................................................. 143
- How Web clients use PICS .......................................................... 144
- How the HTTP Server helps you manage PICS labels .................. 145
  - PICS for Web site administrators ........................................... 145
  - PICS for rating services and label bureaus .............................. 146
  - How to manage PICS labels from a central file ...................... 147
  - Storing the PICS files on your server ...................................... 148
  - Managing PICS labels for your own Web site ....................... 148
  - Starting a PICS rating service and label bureau ..................... 148
  - How to create PICS labels ....................................................... 149
  - PICS label extensions ............................................................... 149
  - How to request PICS label information .................................... 150
  - How to update the PICS configuration file .............................. 151
- Using the online Configuration and Administration forms .............. 151
- Editing the PICS configuration file manually ........................... 151
- Using wildcards in the PICS configuration file ......................... 153

# Chapter 14. Retrieving LDAP information ..................................... 155

- Introduction and concepts ......................................................... 155
  - X.500 overview ........................................................................ 156
  - LDAP overview ....................................................................... 156
- Querying the LDAP server ......................................................... 157
  - LDAP search filters .................................................................. 157
  - Examples of LDAP search filters ............................................. 158
- Configuring LDAP on the HTTP Server ....................................... 158
  - Using LDAP to protect files ..................................................... 158
  - Using LDAP to share configuration information ....................... 161

# Chapter 15. Running your server as a proxy .................................. 163

- Web Traffic Express proxy features ........................................... 163
- Setting up your proxy server ....................................................... 163

# Chapter 16. Running your server with multiple IP addresses or virtual hosts . 165

- Requirements for using multiple IP addresses or virtual hosts .... 165
  - Multiple IP addresses ............................................................... 165
  - Virtual hosts ............................................................................ 165
- Setting up your Web server to use multiple IP addresses or virtual hosts .................................................. 166
  - Welcome pages ....................................................................... 166
  - Mapping rules ......................................................................... 166
  - Access Control ........................................................................ 167

# Part 6. Programming .................................................................... 169

# Chapter 17. Writing Common Gateway Interface programs ............... 171

- Overview of the CGI ................................................................. 171
  - CGI and dynamic documents ............................................... 172
  - Uses for CGI .............................................................................. 172
- FastCGI support ................................................................. 173
  - Overview .................................................................................. 173
  - Updates to the Web server configuration file for FastCGI ........ 173
  - Updates to the FastCGI configuration file ............................. 173
  - Installing the FastCGI Developer’s Kit ................................... 175
- Forms and data processing .......................................................... 178
  - Creating HTML documents that reference CGI programs .... 178
  - Sending information to the server ............................................ 181
  - Processing the information ....................................................... 181
  - Returning output .................................................................... 184
  - Protecting your programs ....................................................... 184
  - Environment variables ............................................................. 184
  - Processing standard search (ISINDEX) documents ................ 184
  - Passing SSL environment variables to a CGI program .......... 185
Chapter 18. Writing GWAPI programs 197
Overview of the GWAPI 197
General procedure for writing GWAPI programs 198
Guidelines for writing GWAPI programs 198
Server request process 199
Application functions 201
Predefined functions and macros 206
GWAPI configuration directives 212
Compatibility with other APIs 214
Porting CGI programs 214
GWAPI reference information 214
Authentication and authorization 214
Environment variables 217
GWAPI on OS/390 217
OS/390 GWAPI REXX applications 217
Example GWAPI REXX service executable program 224
Example GWAPI REXX data filter executable program 225
Storing GWAPI programs 226
HTCodePage_t, the other data type 226
GWAPI samples 226
Debugging C/C++ GWAPI programs 226
Overview of support and restrictions 227
Getting started 227
Troubleshooting hints and tips 228

Chapter 19. Accessing LDAP information with the LDAP API 229
Overview of LDAP API 229
Accessing LDAP configuration information 229
Sample LDAP program 230

Part 7. Appendixes 231
Appendix A. Certificate authority utility 233
Overview of certificate authority options 233
Before you begin 234
Migrate existing CA utility files 234
Install WebSphere Application Server and make required jvm.properties file updates 234
Verify that you are using a compatible browser 234
Ensure required MIME types are in the server configuration file 235
Create key databases and certificates using IKEYMAN 235
Copy CA key database and certificate into the CA utility directory 235
Export your CA key database into the CA utility 235
Enable client certificates for %%CLIENT%% support (optional) 235
Accessing the HTTP Server CA utility 236
Using HTTP Server CA to process client and server certificates 236
Overview of CA process 236
Administrator tasks 236
User tasks 237
Webmaster tasks 239

Appendix B. Commands 241
cgiparse command 241
Syntax 241
Flags 241
Examples 243
Exit statuses 244
cgiutils command 244
Syntax 244
Flags 244
Examples 246
htadm command 246
Syntax 246
Flags 246
Examples 249
httpd command 249
Syntax 249
Flags 249
Tracing flags 251
Signal handling 252
Examples 253
IMWHTTPD program 253
Syntax 253
Parameters 254
Signal handling 259
Examples 260
IMWJWM PROC (workload management) 260
wwwcmd command 261
Syntax 261
Parameters 261
Action 261
OS/390 MODIFY console command 261
Syntax 261
Parameters 262
Tracing parameters 262
Examples 264
OS/390 Workload Management console commands 265
WLM mode 265
Application environments 265
WLM systems 266

Appendix C. Configuration directives 267
Overview of directives 267
Quick reference 267
Directive description and syntax 274
Access control - Set up access control for the server 275
DefProt - Specify default protection setup for requests that match a template 275
Protect - Activate protection setup for requests that match a template 278
Contents  

vii
Map - Change matching requests to a new result string 362
Pass - Accept matching requests 364
Redirect - Send matching requests to another server 365
InheritEnv - Specify which environment variables are inherited by CGI programs 366
DisInheritEnv - Specify which environment variables are dis inherited by CGI programs 367
Security - Set up secure connections for the server 367
KeyFile - Set name for key database file 367
NormalMode - Turn port on or off for HTTP 368
SSLCipherSpec - Specify levels of encryption supported by the server 369
SSLClienAuth - Select the type of SSL client authentication 370
SSLMode - Turn port on or off for SSL 371
SSLPort - Set port for SSL security 371
SSLServerCert - Associate a key database with an IP address 371
SSLV2Timeout - Set SSL V2 session timeout value 372
SSLV3Timeout - Set SSL V3 session timeout value 373
SSLX500CA - Specify location of trusted CA certificates 373
SSUX500Host - Specify host name or IP address of the X.500 directory server 373
SSUX500Port - Specify X.500 directory server port number 374
SSUX500UserID - Specify user ID for the LDAP connection to the X.500 directory server 374
SSUX500Password - Specify user ID password for the LDAP connection to the X.500 directory server 375
System Management - Define system management settings 375
AppEnv - Specify application environment for workload management 375
AppEnvConfig - Specify a set of directives to tailor the application environment 376
PluginDefault - Specify default Plugin action 378
PluginExclude - Specify Plugin not to load during initialization 378
PluginInclude - Specify Plugin to be loaded during initialization 378
SNMP - Enable and disable SNMP support 381
SNMPCommunityName - Provide a security password for SNMP 381
WebMasterEmail - Create an e-mail address to receive SNMP problem reports 382
Timeouts - Close connections automatically 382
InputTimeout - Specify time allowed for the client to send a request 382
OutputTimeout - Specify maximum time for sending output to the client 383
ScriptTimeout - Specify time allowed for a program to complete 383
Tuning - Define performance and scalability settings 383
CacheLocalFile - Specify files you want to load in memory at start up 384
CacheLocalMaxBytes - Specify maximum amount of memory to use for file caching 384
CacheLocalMaxFiles - Specify the maximum number of files for caching 385
EnableFRCA - Turn dynamic caching on or off 385
FRCAccessLog - Specify a log file path and name for dynamic caching requests 385
FRCAEntries - Specify the maximum number of files to be dynamically cached 386
FRCAEnabled - Specify URLs to be dynamically cached 386
FRCAFileSize - Specify size of the dynamic cache 386
FRCAOnly - Specify dynamic caching only for the dynamic cache 387
FRCAServerCert - Associate a key database with an IP address 387
FRCAStackName - Specify the TCP/IP stack that supports the dynamic cache 388
FRCAWLMParms - Specify parameters for Workload Management 389
ListenBacklog - Specify the number of listen backlog client connections for the server to carry 389
LiveLocalCache - Specify whether the cache is updated when a cached file is modified 390
MaxActiveThreads - Specify the maximum number of threads to have active 390
MaxPersistRequest - Specify the maximum number of requests to receive on a persistent connection 390
PersistTimeout - Specify time to wait for the client to send another request 391
ServerPriority - Specify the priority you want your server to have on your system 391
UseACLs - Specify whether ACL files will be checked 391
UseMetaFiles - Specify whether meta files will be used 392

Appendix D. Data set naming reference 393

Appendix E. Environment variables 395
Overview 395
Variables with values that are read-only 395
Variables with values that can be set or created 399
Server-Side Include variables 403

Appendix F. GWAPI MVDS DLL Service 405
Preparing the MVDS DLL Configuration File 405
Specifying the MVDS DLL configuration directives 405
Summary of Changes for IBM HTTP Server Version 5.2 for OS/390

SC31–8690–04 for OS/390 Releases 8 and 9

This book contains information previously presented in SC31-8690-03 and SC31-8690-02 for OS/390 Releases 8.

This book is available in softcopy formats only. The most current version is available in HTML and PDF formats on the Web site at URL:
http://www.ibm.com/software/websphere/httpservers/doc52.html

Technical changes or additions are indicated by a vertical line to the left of the change.

Certificate Authority (CA) utility

New configuration requirements for using the Web server’s CA utility, HTTP Server CA, are outlined in “Install WebSphere Application Server and make required jvm.properties file updates” on page 224.

Configuration directives

Bounce directive
With APAR PQ31375, the Bounce (-B) start option is set on by default, and the new Bounce directive is provided to change the default setting. There is no option on the httpd command to override the default of Bounce on.

To set Bounce off, you must set the the Bounce directive off in the httpd.conf configuration file and omit -B from the Web server startup command. For more information, see the following sections:
- “httpd command” on page 249
- “Bounce — Specify the default start option for the sockets setting SO_REUSEADDR” on page 288

FRCAVirtualHost directive
With APAR PQ28274, the new FRCAVirtualHost directive enables you to indicate to the Fast Response Cache Accelerator cache whether multiple virtual host names or IP addresses are being used. For more information, see “FRCAVirtualHost — Indicate to the dynamic cache whether multiple virtual hosts or IP addresses are being used” on page 388.

Timeout and MaxPersistRequest directives
For performance reasons, the program default and initial configuration file settings for the following timeouts have been lowered by APAR PQ31524:
- “InputTimeout - Specify time allowed for the client to send a request” on page 382
- “OutputTimeout - Specify maximum time for sending output to the client” on page 383
- “ScriptTimeout - Specify time allowed for a program to complete” on page 383
- “PersistTimeout - Specify time to wait for the client to send another request” on page 391
APAR PQ31524 also raised the program default setting for MaxPersistRequest. Specify the maximum number of requests to receive on a persistent connection on page 390.

IBM Key Management Utility (IKEYMAN)

Updates have been made to the following sections:
- “Migrate existing key ring files” on page 410
- “User interface task reference” on page 411
- “Command reference” on page 418

OS/390 Debug Tool for C/C++ GWAPI programs

Information and Web URLs have been updated for the OS/390 Debug Tool. For more information, see “Debugging C/C++ GWAPI programs” on page 226.

SMF performance records

The names and descriptions for the System Management Facilities (SMF) performance records have been updated. These records are listed in the table, “SMF performance record data area (record type 103, subtype 02)” on page 139.

SSI environment variables

With APAR PQ28612, the server-side include (SSI) LAST_MODIFIED environment variable has been changed. For more information, see “echo - specify environment variables” on page 97.

Web server start options

Bounce (-B) start option
With APAR PQ31375, the Bounce (-B) start option is set on by default, and the new Bounce directive is provided to change the default setting. There is no option on the httpd command to override the default of Bounce on.

To set Bounce off, you must set the the Bounce directive off in the httpd.conf configuration file and omit -B from the Web server startup command. For more information, see the following sections:
- “httpd command” on page 249
- “Bounce — Specify the default start option for the sockets setting SO_REUSEADDR” on page 288

SC31–8690–03 for OS/390 Release 8

This book contains information previously presented in SC31-8690-02 for OS/390 Release 8.

This book is available in softcopy formats only. The most current version (SC31-8690-04) is available in HTML and PDF formats on the Web site at URL: http://www.ibm.com/software/websphere/httpservers/doc52.html

Technical changes or additions are indicated by a vertical line to the left of the change.

Configuration directives
The following directive has been added:
Summary of Changes for IBM HTTP Server Version 5.2 for OS/390

Messages
For the most current information on messages, see the Messages section of the Web-based WebSphere Troubleshooter for OS/390.

To link to the Troubleshooter, go to URL:

New
The following messages have been added to “Appendix H. Messages” on page 421:
- IMW0333E-IMW0365E
- IMW0368E-IMW0412E

Changed
The following message has changed in “Appendix H. Messages” on page 421:
- IMW3502I

Web server activity and performance statistics
With APAR PQ28258, new Server Activity Monitor statistics and performance SMF records have been added to help monitor Web server status and performance:
- New information and statistics have been added to the section, “Server Activity Monitor” on page 113.
- New performance SMF records have been added. For details, see the table in “SMF performance record data area (record type 103, subtype 02)” on page 139. The new records begin at offset AVE+132.
- New Web server statistics have been added to message IMW3502I. For a description of this message, see “Appendix H. Messages” on page 421.

SC31–8690–02 for OS/390 Release 8
This book contains information previously presented in the following books for Version 5.1 HTTP Server for OS/390 Version 2 Release 7:
- HTTP Server Planning, Installing, and Using
- HTTP Server Web Programming Guide

This book is available in hardcopy and softcopy formats. The most current version (SC31-8690-04) is available in HTML and PDF formats on the Web site at URL:
http://www.ibm.com/software/websphere/httpservers/doc52.html

Technical changes or additions are indicated by a vertical line to the left of the change.

Configuration directives

Directives
The following directives have been added:
- “Recovery — Customize ABEND recovery performed by the Web server” on page 292
- “DebugToolAddr - Identify the workstation running the Remote Debugger” on page 312
- Logging and reporting directives:
Error recovery and problem determination

ABEND recovery
The Web server now has the ability to recover from ABENDs. The action taken by the Web server is determined by the setting on the new Recovery directive and the type of processing the Web server is doing at the time of the ABEND. For more information, see the following sections of this book:
- “ABEND recovery performed by the Web server” on page 3
- “Recovery — Customize ABEND recovery performed by the Web server” on page 292

Trace options
The descriptions of the httpd command, IMWHTTPD program, and the OS/390 MODIFY console command in “Appendix B. Commands” on page 241 have been updated due to the following changes to Web server trace options (APAR PQ28959):
- To make the -vv trace more readable and less verbose, some trace points have been moved to the -mtv trace. Therefore, the -vv trace no longer provides maximum tracing.
- Tracing of trace stacks has been moved from the -mtv trace to the debug trace option.
- Modules MODULE_STACKS, MODULE_NETMON, and MODULE_CREATORS can no longer be traced using the -mtv trace option. You must now use the -debug option to turn on tracing of these modules.

The level of tracing provided is -v trace, first level; -vv trace, second level; -mtv, third level; and -debug for maximum tracing. To trace caching, use the -vc option.

Note: We recommend that you do not turn tracing on unless instructed to do so by IBM support personnel. IBM support personnel can give you guidance on the most appropriate trace for your problem.

Troubleshooter
- Troubleshooting hints and tips have been moved to the Web-based WebSphere Troubleshooter for OS/390.
  To link to the Troubleshooter, go to URL:

Logging and reporting
New information has been added on specifying a third-party reporting program and changing reporting options for the default reporting program, HTLOGREP. For more information, see “Tailoring the reports your server creates” on page 78.
Messages

For the most current information on messages, see the Messages section of the Web-based WebSphere Troubleshooter for OS/390.

To link to the Troubleshooter, go to URL:

New

The following message has been added to "Appendix H. Messages" on page 421:
• IMW0162E

Changed

The following messages have changed in "Appendix H. Messages" on page 421:
• IMW0085E
• IMW3517I-IMW3520I
• IMW3528I

Programming information

Beginning with V5.2 HTTP Server, there will no longer be a separate Programming Guide.

The following new programming sections have been added to this book:
• “Chapter 17. Writing Common Gateway Interface programs” on page 171
• “Chapter 18. Writing GWAPI programs” on page 197
• “Chapter 19. Accessing LDAP information with the LDAP API” on page 229
• “Appendix E. Environment variables” on page 395

These sections contain OS/390-specific information that was previously included in the V5.1 HTTP Server Web Programming Guide for OS/390 and workstation platforms.

GWAPI information

OS/390 Debug Tool: Support is now available for using the OS/390 Debug Tool to debug C/C++ GWAPI programs. For more information, see “Debugging C/C++ GWAPI programs” on page 223.

Multi-threaded GWAPI programs (APAR PQ28276): Support is now available for writing multi-threaded GWAPI programs. For more information, see “Guidelines for writing GWAPI programs” on page 198.

Environment variables: The following new GWAPI environment variables have been added:
• SERVER_CFG_PORT
• SERVER_CFG_SSLPORT
• WQ_SUBSYS

For a description of these variables, see “Appendix E. Environment variables” on page 394.

REXX Data Filter example: A new REXX Data Filter example has been added. For more information, see “Example GWAPI REXX data filter executable program” on page 223.
Welcome!

The HTTP Server is a scalable, high-performance Web server that brings you state-of-the-art security, dynamic caching capabilities, advanced server statistic reporting, and site indexing. It allows you to exploit Java to build dynamic, personalized Web sites and use the Platform for Internet Content Selection (PICS) to both rate and filter Web content. With the HTTP Server, you can establish an effective presence on the World Wide Web, reach customers and suppliers around the world, and conduct secure electronic commerce.

For the latest HTTP Server product offerings, information, and news, visit the Web site at URL:

http://www.ibm.com/software/websphere/httpservers/

For the most current version of this book and related product documentation, go to the HTTP Server Web site Library page at URL:

http://www.ibm.com/software/websphere/httpservers/doc52.html

Easy-to-use graphical interface

The HTTP Server has a Web-based graphical user interface you can use to remotely configure and administer the server. It includes a set of HTML Configuration and Administration forms that dynamically update the server’s configuration files, along with integrated online information that provides on-the-spot assistance to help you complete the tasks.

The interface has a multi-frame window for improved navigation. You always have a list or index of topics available in one frame while you work with the forms and information in another frame. Because the interface employs HTML framesets and Java applets, it requires a frame-enabled browser that supports the JDK 1.1. See “Browser requirements” on page 4 for specific requirements.

Comprehensive product information

The HTTP Server has a comprehensive package of product information that is available in many formats: integrated user assistance; books in BookManager, HTML, PDF, and hardcopy formats; and information updates on the Web site.

Integrated user assistance

User assistance is available in the product interface to help you complete tasks and fill out the Configuration and Administration Forms. Click the question mark (?) in the upper right-hand corner and select:

- How do I? for task-level help
- Field descriptions for help on a specific form
- Index for a list of all help topics

Books

Updates on the Web site

The most current documentation is available in HTML and PDF formats on the Web site at URL:

http://www.ibm.com/software/websphere/httpservers/doc52.html
BookManager books
Web server documentation is available in BookManager format on the OS/390 CD-ROM Collection Kit and from the OS/390 Web site at URL:


For a summary of available OS/390 books and online information, see the OS/390 Information Road Map.

HTML books
An HTML version of HTTP Server Planning, Installing, and Using is installed with your Web server. This book and other documentation can be accessed from your server’s default Front Page at URL:

http://your.server.name/Frontpage.html

For your.server.name, enter your server’s fully qualified host name (for example, myhost.ibm.com).

From the default Front Page of the Web server, you can link to online versions of the following information:

• HTTP Server Planning, Installing, and Using
• Application Server documentation
• WebSphere Troubleshooter for OS/390
• IBM Web Traffic Express for Multiplatforms User’s Guide V1.0

The most current information is available on the HTTP Server Web site at URL:

http://www.ibm.com/software/websphere/httpservers/doc52.html

PDF books
Portable Document Format (PDF) versions of the books are available on the Web site for you to print or view with the Adobe Acrobat reader.

To access PDF books, go to URL:

http://www.ibm.com/software/websphere/httpservers/doc52.html

Hardcopy books
You can order a hardcopy version of the following books from IBM:

• HTTP Server Planning, Installing, and Using (SC31-8690-02). Updates of this book are available in softcopy formats only.
• For proxy server support, IBM Web Traffic Express for Multiplatforms User’s Guide V1.0 (GC31-8645-00) support

Support services
For information on support options and resources, refer to the Support section of the Web-based WebSphere Troubleshooter for OS/390.

To link to the Troubleshooter, go to URL:

Part 1. Planning
Chapter 1. Planning for installation

<table>
<thead>
<tr>
<th>Summary of Changes</th>
<th>Requirements and planning considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>For a description of new functions and information for this release, review the “Summary of Changes for IBM HTTP Server Version 5.2 for OS/390” on page xi. This section also includes information on new and changed directives and messages.</td>
<td></td>
</tr>
</tbody>
</table>

Requirements and planning considerations

**ABEND recovery performed by the Web server**

The Recovery directive enables you to customize how the Web server will handle error recovery when an ABEND occurs.

By default, the Web server will attempt to recover from an ABEND and will take the following actions in response to a recoverable situation:

- Issue messages to the MVS console, trace log, and error log
- Take a CEE dump of the environment at the time of the ABEND
- Continue processing

To properly obtain the CEE dump, you must have a CEEDUMP DD statement in your JCL if you are starting the Web server from a PROC. If you are starting the Web server from the OS/390 UNIX shell, ensure you have the _CEE_DMPTARG environment variable set to the path where you want the dump to be stored.

For information on error recovery options, see “Recovery — Customize ABEND recovery performed by the Web server” on page 292. For the most current information on dumps and the Web server error recovery process, see the Web-based WebSphere Troubleshooter for OS/390.

To link to the Troubleshooter, go to URL: [http://www.ibm.com/software/websphere/httpservers/troubleshooter.html](http://www.ibm.com/software/websphere/httpservers/troubleshooter.html)
Access control with RACF and other SAF-based security products

For controlling access to Web resources on MVS, the Web server provides extensive access control support, including system validation of user IDs and passwords and access control through surrogate user ID support.

Resource Access Control Facility (RACF), or an equivalent SAF-based security product, manages system and data security by verifying a user’s identity and access to a resource.

In general, SAF security checking with RACF, or another vendor’s SAF-based security product, is recommended because it can protect the installation from unauthorized access to MVS as well as Web server resources.

Users are identified by an OS/390 UNIX user ID (alphanumeric) kept in the RACF user profile, and an OS/390 UNIX group ID (GID) kept in the RACF group profile. For more information on setting up user and group IDs, see the OS/390 UNIX System Services Planning book.

RACF performance suggestions
If using RACF or another SAF-based security product, consider the following hints and tips:

- Use surrogate user IDs, if that is acceptable from a security point of view.
- If you use %SAF%% with the PUBLIC surrogate user ID, this provides protection of your server resources without a high CPU cost. OS/390 Release 3 or greater automatically enhances performance when you use surrogate user IDs and RACF.
- If you require Web clients to have a unique SAF-based user ID and password on the Web server system, specifying %SAF%% with %CLIENT%% will give you more robust security but at a higher CPU cost.
- Put your RACF data set on a control unit with caching and the DASD fast write feature. Instructions for enabling caching are in the documentation for DFSMS. Contact your DASD support programmer.
- Cache highly used pages.
- Clean up the RACF database periodically.
- If running Workload Management, make sure you permit Workload Management to RACF.

Related information:
- “Surrogate user IDs” on page 21
- “Step 7. Enable the Web server to support OS/390 Workload Management (optional)” on page 22

Browser requirements
To configure the Web server using the Configuration and Administration Forms, you need a browser that:

- Can display frames
- Supports Java Development Kit (JDK) 1.1.x
- Is enabled for both Javascript and Java
- Has color resolution set to at least 256 colors (operating system setting)
- Is set to cache documents and compare the cached document with the network document every time

Related information:
If you are using the HTTP Server Certificate Authority utility, see Note (1).

There are special browser requirements if you are using VeriSign’s Global Server ID. This is a special digital certificate available to financial and banking Web servers. For more information, see “Stronger encryption option for financial and banking Web servers” on page 53.

**Tested browsers**

*Note:* For the most current information on browser requirements, tested browsers, and browser hints and tips for the Web server, see the Browsers section of the Web-based WebSphere Troubleshooter for OS/390.

To link to the Troubleshooter, go to URL: [http://www.ibm.com/software/websphere/httpservers/troubleshooter.html](http://www.ibm.com/software/websphere/httpservers/troubleshooter.html)

**Web server Configuration and Administration Forms**

*Table 1. Browsers tested with the Web server Configuration and Administration Forms*

<table>
<thead>
<tr>
<th>Operating system</th>
<th>Browser level</th>
</tr>
</thead>
</table>
| Windows NT (2) and Windows 95 | • Netscape:  
  – Navigator 4.08 and 4.04  
  – Navigator Gold 3.01  
  – Communicator 4.51, 4.5, and 4.04  
  The JDK 1.1 SmartUpdate patch is required. (3)  
• Microsoft Internet Explorer 5, Version 5.00.2014.0216IC (4)  
To verify the version number, click Help, then About Internet Explorer.  
• Microsoft Internet Explorer Version 4.0, upgrades:  
  – V4.71.1712.6  
  – V4.72.2106.8  
To verify the upgrade number, click Help, then About Internet Explorer. |
| AIX | • Netscape Navigator V4.07 and V4.04 |
| OS/2 | • Netscape Navigator V4.04 with JDK 1.1.7 (5)  
• Netscape Navigator V2.02 with service level 7 (only the runtime code is required) (6) |

**Hints and tips:**

- Help is launched in its own separate window. At times, you might need to move the help window to see the original window under it.
- In Netscape Navigator, when you maximize the configuration browser window and then resize it, you might lose the help and restart icons. Simply maximize the window to make them reappear.

**Notes:**

(1): **CA Utility Requirements:** If you are using the CA utility and do not need access to the Configuration and Administration Forms:

- Netscape Navigator V3.x can be used on Windows NT, Windows 95, and AIX.
- Netscape Navigator V4.x can be used on Windows NT, Windows 95, and AIX without the JDK 1.1 SmartUpdate patch
Planning for installation

- Netscape V2.02 can be used on OS/2 without service level 7.

The CA utility has been tested with the following Microsoft Internet Explorer browser levels:
- 5.0, Version 5.00.2014.0216IC
- 4.01 with Service Pack 2
- 4.0 with Service Pack 1
- 4.0, upgrade 4.72.3110.8

To verify the version or upgrade number, click Help, then About Internet Explorer.

(2): Windows NT 4.0 should be configured to run with more than 256 colors to enable GIFs to display properly.

(3): The browser and JDK 1.1 SmartUpdate patch are two separate installations. The JDK 1.1 SmartUpdate patch updates the JDK level to 1.1.4.

To confirm that you have the correct level, look at the header in the Netscape Java Console (available from the Netscape Communicator menu). The header should say "Netscape Communications Corporation -- Java 1.1.4."

To download the JDK 1.1 SmartUpdate patch, go to URL [http://www.netscape.com/](http://www.netscape.com/)

(4): When using the Configuration and Administration Forms, note that Microsoft Internet Explorer 5, Version 5.00.2014.0216IC may prompt you for your administrative ID and password each time you perform a task. This is a browser problem.


(6): To confirm that you have Netscape Navigator service level 7, open the Installation utility in the Netscape folder. Select the Netscape 2.02.00 item and click Details, then Product Status. Select Netscape Navigator then click Service Level.

To confirm that you have Java 1.1.2 or higher, enter java -version on the command line.

If you have not installed Netscape Navigator, install Java first. When you install Navigator, you will be prompted for the Java level.

If you have already installed Navigator, install Java and then click Java Version Selection in the Netscape folder to select Java 1.1.

You can download Navigator and Java from the IBM Software Web site at URL [http://service.boulder.ibm.com/asd-bin/doc/](http://service.boulder.ibm.com/asd-bin/doc/) If you are installing Java 1.1.4, you need OS/2 Feature Installer Version 1.1 or higher. You can download the latest version of OS/2 Feature Installer from this site.

Fast Response Cache Accelerator

If you want to use the dynamic caching function, see "Chapter 9. Customizing cache management with the Fast Response Cache Accelerator" on page 69 for planning considerations.
**MVSDS DLL OS/390 data set considerations**

The Web server can use a DLL named MVSDS to preload OS/390 data sets. Preloading of OS/390 datasets is suggested for frequently accessed Web content. Note that MVS data sets can be accessed without preloading them.

Decide which, if any, data sets are to be preloaded when the server is started. Then add these datasets to the MVSDS DLL configuration file and specify the MVSDS DLL configuration file name on the ServInit directive in the server configuration file. The default MVSDS DLL configuration file is `/etc/mvsds.conf`.

**Related Information:**
- “Appendix F. GWAPI MVSDS DLL Service” on page 405
- “ServerInit - Customize the Server Initialization step” on page 312

**OS/390 UNIX System Services considerations**

**Authorization environment considerations**

Set up user ID authorizations and security authorizations before starting the Web server. For detailed information on setting up users and security considerations, see the OS/390 UNIX System Services Planning book.

**Considerations for controlling superusers under OS/390 UNIX**

In a UNIX system it is common for one person to have full administrative authority. In an OS/390 system, it is common for these administrative authorities to be divided among several people. OS/390 UNIX lets you separate some of the authorities normally granted to superusers by creating four System Authorization Facility (SAF) classes:

**BPX.SUPERUSER**

This facility is not used by the Web server. This facility enables non-superusers to gain superuser authority for OS/390 UNIX resources only (for example, HFS files).

**BPX.DAEMON**

The Web server usually uses this facility for daemon programs that need to validate user passwords and then change the MVS identity and OS/390 UNIXUID and GID of a spawned address space.

**BPX.SERVER**

The Web server usually uses this facility for its programs that use POSIX threads and need to associate a Surrogate MVS identity with each thread in their address space.

**BPX.SMF**

The Web server usually uses this facility to validate read access to the its user ID for writing SMF records.

**DLL considerations within the OS/390 UNIX environment**

A dynamic link library (DLL) is a file containing executable code and data bound to a program at load time or run time. Some DLL considerations:

- DLLs generated for a BPX.DAEMON environment must be loaded from a program control protected data set to ensure the DLL does not corrupt the the HTTP Server address space.

- The DLL must adhere to traditional OS/390 naming conventions.

- If you simulate the naming conventions used by the DLLs shipped with the product, external links from the HFS file to the PDS member name equivalent must be created.
Planning for installation

For example, to issue an external link for DLLs defined from the OS/390 UNIX shell, issue the following:

```
ln -e IMWDFTFM /usr/lpp/internet/bin/urdll.so
```

**Notes:**

1. The **ln** command is case sensitive.
2. For **urdll.so**, enter the DLL to be loaded from the server. Since **urdll.so** does not adhere to OS/390 naming conventions (namely, “.” is not a valid character in a PDS member name and not eight characters or less), an external link to the load module **IMWDFTFM** is placed in the **/usr/lpp/internet/bin/** directory.
3. The member **IMWDFTFM** is located in the traditional OS/390 search order when running from the shell.

For more information, see the OS/390 UNIX System Services Command Reference. Find additional information about DLLs in the OS/390 C/C++ Programming Guide book.

The data sets containing the DLLs should be included in the LPA, LNKLST, or STEPLIB. For performance, LPA is the preferred location, then LINKLST, then STEPLIB.

Alternately, a DLL file stored in an HFS can be used if it is marked program controlled with the **extattr +p efn** command. You need READ access to the BPX.FILEATTR.PROGCTL facility. For more information about the **extattr** command, see the OS/390 UNIX System Services Planning book.

**Performance considerations**

For the most current performance hints and tips, see the Tuning section of the Web-based WebSphere Troubleshooter for OS/390.

To link to the Troubleshooter, go to URL:

```
```

**Related Information:**

- "RACF performance suggestions" on page 4
- "Performance considerations when starting the Web server" on page 31
- "Support for specifying the encryption level to be used" on page 53
- "Chapter 9. Customizing cache management with the Fast Response Cache Accelerator" on page 69
- "Scalable Server mode" on page 106
- "Workload Management Enablement for the Web server" on page 107
- "Server Activity Monitor" on page 113
- "Logging information with System Management Facilities" on page 88
- "System Management Facilities" on page 139
- "Chapter 15. Running your server as a proxy" on page 163
- "System Management - Define system management settings" on page 375
- "Tuning - Define performance and scalability settings" on page 383

**Protection setup considerations**

Protection setups that use the same password file should use the same ServerId strings to avoid reprompting at the browser to cache the same information multiple times. Protection setups that use different password files should use different ServerId strings. This forces prompting once per realm and avoids thrashing the browser’s user ID or password cache each time realms are switched.
For more information on protection setups, see “Protection - Define a named protection setup within the configuration file” on page 28.

Reporting programs used by the Web server
You can use the default IBM reporting program, HTLOGREP, or specify a third-party reporting program, such as Analog. For more information, see “Tailoring the reports your server creates” on page 78.

Security considerations

Examples
You do not have a secure network connection until you have created a key for secure network communications and received a certificate from a certificate authority (CA) who is designated as a trusted CA on your server. For examples, see:

- “Setting up secure connections using an external CA” on page 63
  Use this example if you plan to have your operational server certificate signed by an external certificate authority (CA), such as VeriSign.
- “Setting up secure connections using self-signed certificates” on page 63
  Use this example if you plan to act as your own CA for a private Web network. This example shows you how to use IKEYMAN and HTTP Server CA to set up and administer your own certificate authority.

SSL and hardware encryption
V5.0 of the Web server added new SSL enhancements that allow the installation to designate a preference order for cipher specifications using the SSLCipherSpec directive. This enables you to configure the Web server to perform DES and Triple-DES encryption using hardware encryption. For information on enabling this support on the Web server, see “Hardware encryption” on page 57.

SSL performance and tuning information
For the most current SSL performance and tuning information, see the Tuning hints and tips in the Web-based WebSphere Troubleshooter for OS/390.

To link to the Troubleshooter, go to URL:

WebSphere Application Server Java-based servlet engine
For servlet support, install the WebSphere Application Server.

For installation and configuration information, refer to the Application Server documentation. You can access Application Server documentation from the default Front Page of the Web server or from the Application Server Web site at URL:

Workload Management considerations
In an OS/390 production environment, Workload Management (WLM) running in Goal mode can be used to balance workloads and distribute resources among competing workloads. To exploit the benefits of WLM, the Web server can be enabled for WLM support, that is, running in Scalable Server mode. This means that the Web server is configured for WLM support using the ApplEnv directive and is started using the -SN (subsystem name) parameter. The ApplEnv directive is used by the Web server to divide incoming requests into Application Environments and to route those requests.
Planning for installation

You should know about the following tasks when considering using workload management:

- Formatting coupled data sets
- Permitting workload management to RACF
- Using workload management panels
- Using operator console commands to verify workload management is working

Related Information:

- For information on permitting WLM to RACF, see "Step 7. Enable the Web server to support OS/390 Workload Management (optional)" on page 22.
- For overview and configuration information, see "Workload Management Enablement for the Web server" on page 107.
- For a description of WLM configuration directives, see "System Management - Define system management settings" on page 375.
- For additional information on WLM, see OS/390 MVS Planning: Workload Management.

APARs and service updates

For the most current information on APAR fixes and service updates, check the OS/390 Program Directory, the product Preventive Service Planning (PSP) documentation, and IBMLink.

You can access the PSP documentation by going to the Support or Documentation section of the Web-based WebSphere Troubleshooter for OS/390.

To link to the Troubleshooter, go to URL:

PSP documentation is also available on IBMLink. To access IBMLink on the Web, go to URL http://www.ibm.com/ibmlink/
Chapter 2. Migration considerations

Accessory script directives

Beginning in Version 2.2 of the Internet Connection Secure Server, the accessory script directives, DELETE-Script, POST-Script, PUT-Script, and Search, are no longer supported. You need to port your scripts to GWAPI applications. For more information, see “Chapter 18. Writing GWAPI programs” on page 197.

IMWHTTPD program

The following table lists commands (override parameters), the release each command can be used in, and the configuration directive (if one exists) associated with a specific command. For more information, see “IMWHTTPD program” on page 253. For information about configuration directives, see “Appendix C. Configuration directives” on page 267.

In the following table:
- 2.1 and 2.2 refer to the IBM Internet Connection Secure Server.
- 4.6.1 refers to Lotus Domino Go Webserver.
- 5.x refers to Lotus Domino Go Webserver (5.0) and the IBM HTTP Server (5.1 and later).

<table>
<thead>
<tr>
<th>Command</th>
<th>2.1</th>
<th>2.2</th>
<th>4.6.1/5.x</th>
<th>Directive</th>
</tr>
</thead>
<tbody>
<tr>
<td>-AE</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>ApplEnv</td>
</tr>
<tr>
<td>-cacheroot</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>CacheRoot</td>
</tr>
<tr>
<td>-db</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>DirReadme</td>
</tr>
<tr>
<td>-debug</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>N/A</td>
</tr>
<tr>
<td>-dn</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>DirAccess</td>
</tr>
<tr>
<td>-ds</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>DirAccess</td>
</tr>
<tr>
<td>-dy</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>DirAccess</td>
</tr>
<tr>
<td>-dt</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>DirReadme</td>
</tr>
<tr>
<td>-disable</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Disable</td>
</tr>
<tr>
<td>-enable</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Enable</td>
</tr>
<tr>
<td>-errlog</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>ErrorLog</td>
</tr>
<tr>
<td>-fscp</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>DefaultFsCp</td>
</tr>
<tr>
<td>-gc_only</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>N/A</td>
</tr>
<tr>
<td>-gmt</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>LogTime</td>
</tr>
<tr>
<td>-h</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>HostName</td>
</tr>
<tr>
<td>-l</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>AccessName</td>
</tr>
</tbody>
</table>
### Migration considerations

**Table 2. Command, Version, Associated Directive (continued)**

<table>
<thead>
<tr>
<th>Command</th>
<th>2.1</th>
<th>2.2</th>
<th>4.6.1/5.x</th>
<th>Directive</th>
</tr>
</thead>
<tbody>
<tr>
<td>-lb</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>N/A</td>
</tr>
<tr>
<td>-localtime</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>LogTime</td>
</tr>
<tr>
<td>-mtv</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>N/A</td>
</tr>
<tr>
<td>-netcp</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>DefaultNetCp</td>
</tr>
<tr>
<td>-newlog</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>N/A</td>
</tr>
<tr>
<td>-nobg</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>N/A</td>
</tr>
<tr>
<td>-nodebug</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>N/A</td>
</tr>
<tr>
<td>-nodns</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>DNS-Lookup</td>
</tr>
<tr>
<td>-nolog</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>NoLog</td>
</tr>
<tr>
<td>-normalmode</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>NormalMode</td>
</tr>
<tr>
<td>-nosec</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>N/A</td>
</tr>
<tr>
<td>-nosmf</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>SMF</td>
</tr>
<tr>
<td>-nosnmp</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>SNMP</td>
</tr>
<tr>
<td>-oldlog</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>LogFormat</td>
</tr>
<tr>
<td>-p</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Port</td>
</tr>
<tr>
<td>-r</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>N/A</td>
</tr>
<tr>
<td>-restart</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>wwwcmd command</td>
</tr>
<tr>
<td>-smf</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>SMF</td>
</tr>
<tr>
<td>-SN</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Subsys Name</td>
</tr>
<tr>
<td>-snmp</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>SNMP</td>
</tr>
<tr>
<td>-sslmode</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>SSLMode</td>
</tr>
<tr>
<td>-sslport</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>SSLPort</td>
</tr>
<tr>
<td>-v</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>N/A</td>
</tr>
<tr>
<td>-vc</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>N/A</td>
</tr>
<tr>
<td>-version</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>N/A</td>
</tr>
<tr>
<td>-vv</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>N/A</td>
</tr>
</tbody>
</table>

#### Java-based servlet engine

In previous releases, Java servlet support was provided by ServletExpress and was installed when you installed the Web server. Beginning in Version 5.1, servlet support is no longer included with the Web server; you must now install the WebSphere Application Server for servlet support.

For installation and configuration information, refer to the Application Server documentation. You can access Application Server documentation from the default Front Page of the Web server or from the Application Server Web site at URL: [http://www.ibm.com/software/websphere/appserv/library.html](http://www.ibm.com/software/websphere/appserv/library.html)

#### Proxy server and caching proxy server

Beginning in Version 5.0 of the Lotus Domino Go Webserver, the process for defining proxy and caching proxy settings changed.
IBM Web Traffic Express 1.0 provides proxy support for the server and is installed when you install the HTTP Server. For configuration information, see “Chapter 15. Running your server as a proxy” on page 163.

Security

Configuration and key management changes

Prior to Version 5.0 of the server, you could create keys and certificate requests using the Configuration and Administration Forms. This option was removed in Version 5.0 for security reasons. The Security Configuration and Administration Forms are now used to perform the following tasks:

- Enable or disable SSL transactions; by default, SSL transactions are enabled.
- Change the server’s default SSL port of 443
- Specify the type of SSL client authentication to use; by default, client authentication is not used.
- Change the server’s default timeout duration for SSL sessions
- Change the server’s default encryption settings; by default, the server will use the strongest encryption level supported by both the client and the server.
- Register key databases you have added using IKEYMAN
- Specify another default key database for the server
- Associate a certificate with a specific IP address, if using multiple IP addresses on the server

In Version 5.0, the IBM Key Management Utility (IKEYMAN) replaced the Make Key File Utility (MKKF) for creating keys and certificate requests. Also in Version 5.0, the IMWSendMail utility was removed and no longer used for sending certificate requests to a CA.

Related Information:

- For security configuration information and examples, see “Chapter 8. Setting up a secure server” on page 45.
- For information on migrating existing key files and using IKEYMAN, see “Appendix G. Key Management Utility” on page 409.

Certificate authority utility changes

Name

In Version 5.0, the certutil command was replaced by a new server certificate authority utility called Domino Go CA. Because of the server name change, this utility is now called HTTP Server CA.

Installation path

In Version 5.1, the installation path for the CA utility changed:

Install directory for V5.0
/usr/lpp/ServletExpress/web/resources/language_code/CAServlet/

Install directory for V5.1 and later releases
/usr/lpp/internet/server_root/CAServlet/language_code/

For language_code, enter C (English) or Ja_JP (Japanese).
If you used the CA utility in Version 5.0 and specified the default names for your files, the installation program could overwrite your existing files. We recommend that you copy the following files from the 5.0 directory to a backup directory before installing the current version of the HTTP Server:

- key database files (*.kdb)
- stash files (*.sth)
- CA key for browsers (*.der)
- CA key for servers (*.txt)
- all files in the CA utility /certs/ directory, which contains the signed certificates (*.cer), certificate data file (*.db), and other related files

After installation is complete, copy your 5.0 files from the backup directory to the current Web server directory.

**Servlet engine**

To use the CA utility, a Java-based servlet engine is required. In Version 5.0, this support was provided by ServletExpress which was installed as part of the Web server. Beginning in Version 5.1, servlet support is provided by the WebSphere Application Server.

**Related Information:**

- For more information on the WebSphere Application Server, see [Java-based servlet engine](#) on page 12.
- For information on using HTTP Server CA, see [Appendix A. Certificate authority utility](#) on page 233.

**Server IP Address**

Beginning in Version 2.2 of the Internet Connection Secure Server, a wildcard character can no longer be specified for a server's IP address. This applies to the following directives:

- Hostname
- Exec
- Fail
- Map
- Pass
- Redirect
- Enable
- NameTrans
- Service

For a description of these directives, see [Appendix C. Configuration directives](#) on page 269.
Part 2. Installing
Chapter 3. Installing your secure server

Before you install ........................................... 17
Review planning information .............................. 17
Check for information updates and hints/tips on the Web ........................................... 17
Perform OS/390 installation procedure .............. 17
Install the Java Development Kit ..................... 17
Completing and customizing your Web server installation ........................................... 18
Step 1. Create the OS/390 UNIX System Services group ID and the Web server user ID .......... 18
Step 2. Create the OS/390 UNIX System Services user ID ........................................... 19
Step 3. Permit SMF to RACF ................................ 19
Step 4. Turn on program control for RACF ......... 19
Step 5. Review access control user ID options and create surrogate user IDs .................. 20
  Special access control user IDs ....................... 20
  Surrogate user IDs ........................................ 20
  Example ..................................................... 21
Step 6. Activate the Lotus Notes Adapter function and HTTPD_local_security function (optional) ........................................... 22
Step 7. Enable the Web server to support OS/390 Workload Management (optional) ........ 22
Step 8. Make TCP/IP configuration adjustments as needed ........................................... 22
Step 9. Copy and customize the Web server PROC ........................................... 23
Step 10. Configure installed files by running setup.sh ........................................... 24
  Before running setup.sh ......................... 24
  Tasks performed by setup.sh ................... 24
  Hints and tips ........................................... 25
  Running setup.sh ....................................... 25
Step 11. Customize your Web server configuration file httpd.conf ................................ 27
  Using a Web server user ID other than WEBADM ........................................... 27
  Changing default directories for logs and reports ........................................... 27
Step 12. Set up security ..................................... 27
Step 13. Set up proxy server support (optional) ........................................... 27
Step 14. Install and configure WebSphere Application Server for servlet support (optional) ........................................... 28
What’s next? .................................................. 28

Before you install

Review planning information

Before you install Version 5.2 of the Web server, review "Chapter 1. Planning for installation" on page 3. If you are migrating from a previous release of the Web server, also review "Chapter 2. Migration considerations" on page 11.

Check for information updates and hints/tips on the Web

For the most current Web server documentation and updates, go to the Web site Library page at URL:

http://www.ibm.com/software/websphere/httpservers/doc52.html

Also, check the Installation hints and tips in the Web-based WebSphere Troubleshooter for OS/390.

To link to the Troubleshooter, go to URL:


Perform OS/390 installation procedure

After completing your OS/390 installation, use the instructions in this chapter to complete your installation and customize the Web server.

For information on OS/390 installation options and considerations, refer to OS/390 Planning for Installation.
Install the Java Development Kit

Before customizing the Web server, install the Java Development Kit (JDK) Version 1.1.6 or higher. You can download the JDK from the Java for OS/390 Web site at URL:

http://www.ibm.com/s390/java/

Completing and customizing your Web server installation

Install Path

In the following steps, install_path is the root directory of your Web server installation. This value is either the default install path (/usr/lpp/internet) or the install path you specified during installation. This is the path on the InstallPath directive in your httpd.conf file.

Migration Notes:

1. To avoid losing valuable configuration settings, copy all your previous configuration files. If your configuration files are located in the default server root directory, /usr/lpp/internet/server_root, copy them to a new directory.
2. Make sure you are pointing to copies of the httpd.conf and httpd.envvars files that are shipped with Version 5.2 of the Web server. It is a good idea to bring up your new Web server with copies of the sample httpd.conf and httpd.envvars files shipped with Version 5.2 before manually adding changes from your previous release files. Updating old levels of these files to use with the new Web server can cause problems and errors.
3. If you are migrating from a previous release, you have probably completed many of the steps in the following procedure. However, it’s a good idea to review all steps because of updates and changes to the Web server. For example, there are many RACF profiles that need to be defined during Web server customization. If they are not set up correctly, you will experience problems.

Step 1. Create the OS/390 UNIX System Services group ID and the Web server user ID

The OS/390 UNIX group ID and Web server user ID are used to configure and administer the Web server. Although you can use any names, this book and the sample files shipped in /install_path/samples assume you have chosen IMWEB for the group ID and WEBADM for the Web server user ID.

The following example shows how you might define IMWEB and WEBADM using RACF commands:

ADDGROUP IMWEB OMVS(GID(205))
ADDUSER WEBADM DFLTGRP(IMWEB) OMVS(UID(206) HOME('/usr/lpp/internet') PROGRAM('/bin/sh'))

Related information:

- Members of the IMWEB group have read/write/execute access to all of the files that control the Web server. The Web server sets up this access when you run the setup.sh command in "Step 10. Configure installed files by running setup.sh" on page 23.
- To use the Web server Configuration and Administration Forms user interface, the WEBADM user ID must have read permission to the PidFile, read/write permission to the Web server configuration file (httpd.conf) and execute...
permission to the wwwcmd command. The Web server sets up these permissions when you run the setup.sh command in "Step 10. Configure installed files by running setup.sh" on page 24. The PidFile is created when you start the Web server. The default PidFile is /install_path/server_root/httpd.pid.

Step 2. Create the OS/390 UNIX System Services user ID

The OS/390 UNIX user ID executes the Web server and validates incoming requests. Although you can use any name, this book and the sample files shipped in /install_path/samples assume you have chosen WEBSRV.

Notes:

1. Make WEBSRV a surrogate user ID for this to work correctly. For more information on surrogate user IDs, see "Step 5. Review access control user ID options and create surrogate user IDs" on page 20.

2. This user ID must have UID of 0 so that it always runs with superuser authority. The Web server always changes to either a surrogate user ID or the client’s local OS/390 user ID prior to accessing the requested resource.

3. If you have defined the BPX.DAEMON facility class, this user ID must be given READ access. Also, you must turn on program control and indicate that the server and Language Environment LOADLIBs are trusted programs.

4. If you have defined the BPX.SERVER facility class, this user ID must be given UPDATE access. You must also turn on program control and indicate that the server and Language Environment LOADLIBs are trusted programs.

The following example shows how you might define WEBSRV using RACF commands:

```
ADDUSER WEBSRV DFLTGRP(IMWEB) OMVS(UID(0) HOME('/usr/lpp/internet') PROGRAM('/bin/sh'))
PERMIT BPX.DAEMON CLASS(FACILITY) ID(WEBSRV) ACCESS(READ)
PERMIT BPX.SERVER CLASS(FACILITY) ID(WEBSRV) ACCESS(UPDATE)
SETROPTS RACLIST(FACILITY) REFRESH
```

Related information:

- "OS/390 UNIX System Services considerations" on page 7

Step 3. Permit SMF to RACF

To use the System Management Facilities (SMF), you first need to permit SMF to RACF, using the following command:

```
PERMIT BPX.SMF CLASS(FACILITY) ID(WEBSRV) ACCESS(READ)
```

For more information on SMF, see "System Management Facilities" on page 135

Step 4. Turn on program control for RACF

The following example shows how to turn on program control if it has been turned on previously. If turning on program control for the first time, you should use RDEFINE statements instead of RALTER statements.

```
RALTER PROGRAM * ADDMEM('SYS1.SCEERUN'//NOPADCHK) UACC(READ)
RALTER PROGRAM * ADDMEM('CBC.SCLBDLL'//NOPADCHK) UACC(READ)
RALTER PROGRAM * ADDMEM('SYS1.LINKLIB') UACC(READ)
SETROPTS WHEN(PROGRAM) REFRESH
```

Note: In the example above * is all programs in the data set or a specified program in the specified data set.
If you have not turned on program control for required datasets, you will probably get 0A2F (address bit dirty) errors. To help you troubleshoot 02AF problems, see the Debugging hints and tips section of the Web-based WebSphere Troubleshooter for OS/390.

To link to the Troubleshooter, go to URL:

Step 5. Review access control user ID options and create surrogate user IDs

Web servers frequently need to process requests from users that do not have user IDs on the system running the server. The server uses access control user IDs to access resources for these requests.

Special access control user IDs
The sample configuration provided with the Web server, protects the default Home Page and samples with the special user ID, %%CLIENT%%. This means that only users known to the system can use the Web server, until you explicitly grant access to other users.

%%CLIENT%%
The Web server requires that the requester have a local OS/390 user ID and password. The requester’s user ID is used to access the data. The user is prompted for a valid password.

%%SERVER%%
The Web server uses its own user ID to access data.

Note: Be extremely cautious when using %%SERVER%%. If your server is running as a superuser, this gives all users superuser authority.

%%CERTIF%%
The Web server treats SSL connection certificate data in a special way. When presented with an SSL session with client certificate data present, the Web server attempts to map the client certificate to a local MVS User ID and password. The request is treated as if %%CLIENT%% has been specified in the following situations:
• The session is not an SSL session.
• There is no certificate present or the certificate cannot be mapped.
• The underlying support is not available.

Note that SSLClientAuth must be set on in order to get client certificate data.

Related information:
• “Access control with RACF and other SAF-based security products” on page 4
• “UserID - Specify the Access Control user ID that the server should use” on page 286

Surrogate user IDs
You probably want to establish several surrogate user IDs with OS/390 UNIX access authority appropriate for a group of users or class of requests.

Examples:
In the following list, WEBADM and PUBLIC are the most commonly used surrogate IDs. INTERNAL and PRIVATE IDs may be used but are not required as part of your Web server setup. They are shown here as examples only.

**WEBADM**
If you want to use the Web server’s remote Configuration and Administration Forms with the supplied sample configuration file, the WEBSRV ID must be allowed to use WEBADM as a surrogate user ID. WEBSRV is the OS/390 UNIX user ID described in “Step 2. Create the OS/390 UNIX System Services user ID” on page 19.

**PUBLIC**
This ID has very limited access and is used to handle requests from unknown users on the public network. You might allow these users to view a few informational pages about your company and products. You would not allow these users to store data on your system or execute more than a few well controlled CGI programs.

**INTERNAL**
This ID has moderate access and is used to handle requests from anyone on your internal corporate network. You might allow these users to view company announcements and standards and perhaps provide a bulletin board application for posting items of general interest.

**PRIVATE**
This ID has access to a set of restricted information. You might require these users to know a Web administered user ID and password or even to have a valid user ID and password on this system. After validating their user ID and password, you would then access the data under the surrogate user ID PRIVATE.

**Example**
The following steps are provided as an example for a server called WEBSRV. This example uses the previously mentioned surrogate user IDs:

1. Activate the SURROGAT class support in RACF by issuing the following command:
   ```bash
   SETROPTS CLASSACT(SURROGAT)
   ```
   This has to be done only once on the system. The SURROGAT class may already have been set up on your system.

2. Define a SURROGAT class profile for each surrogate user ID:
   ```bash
   ADDGROUP EXTERNAL OMVS(GID(999))
   ADDGROUP EMPLOYEE OMVS(GID(500))
   ADDGROUP SPECIAL OMVS(GID(255))
   ADDUSER PUBLIC DFLTGRP(EXTERNAL) OMVS(UID(998) Home(''/') PROG('/bin/sh'))
   ADDUSER INTERNAL DFLTGRP(EMPLOYEE) OMVS(UID(537) Home(''/') PROG('/bin/sh'))
   ADDUSER PRIVATE DFLTGRP(SPECIAL) OMVS(UID(416) Home(''/') PROG('/bin/sh'))
   RDEFINE SURROGAT BPX.SRV.WEBADM UACC(NONE)
   RDEFINE SURROGAT BPX.SRV.PUBLIC UACC(NONE)
   RDEFINE SURROGAT BPX.SRV.INTERNAL UACC(NONE)
   RDEFINE SURROGAT BPX.SRV.PRIVATE UACC(NONE)
   ```

3. Permit the server, WEBSRV, to create security environments using these user IDs:
   ```bash
   PERMIT BPX.SRV.WEBADM CLASS(SURROGAT) ID(WEBSRV) ACCESS(READ)
   PERMIT BPX.SRV.PUBLIC CLASS(SURROGAT) ID(WEBSRV) ACCESS(READ)
   PERMIT BPX.SRV.INTERNAL CLASS(SURROGAT) ID(WEBSRV) ACCESS(READ)
   PERMIT BPX.SRV.PRIVATE CLASS(SURROGAT) ID(WEBSRV) ACCESS(READ)
   SETROPTS RACLASS(SURROGAT) REFRESH
   ```
Step 6. Activate the Lotus Notes Adapter function and HTTPD_local_security function (optional)

To activate the optional Lotus Notes Adapter function and the HTTPD_local_security function:

1. Provide READ access to the RUSERMAP facility defined in RACF:
   
   ```
   PERMIT IRR.RUSERMAP CLASS(FACILITY) ID(WEBSRV) ACCESS(READ)
   ```

2. Assign Lotus Notes IDs to the RACF user profile using either ALTUSER or ADDUSER:
   
   ```
   ALTUSER userid LNOTES(SNAME('jsmith'))
   ADDUSER userid LNOTES(SNAME('jsmith'))
   ```

3. Allow the server surrogate use of the RACF IDs:
   
   ```
   RDEFINE SURROGAT BPX.SRV.userid UACC(NONE)
   PERMIT BPX.SRV.userid CLASS(SURROGAT) ID(WEBSRV) ACCESS(READ)
   SETROPTS RACLIST(SURROGAT) REFRESH
   ```

   `userid` is the RACF user ID and `WEBSRV` is the RACF ID of the Web server.

Related Information:
- Description of the HTTPD_local_security function in "Predefined functions and macros" on page 206

Step 7. Enable the Web server to support OS/390 Workload Management (optional)

This section assumes you have already installed and set up Workload Management (WLM) on your OS/390 system.

If you plan to run the Web server in Scalable Server mode, use this procedure to enable the Web server to support WLM.

- Permit the Web server to WLM by entering:

   ```
   RDEFINE FACILITY MVSADMIN.WLM.POLICY UACC(NONE) NOTIFY(WEBSRV)
   RDEFINE BPX.WLMSERVER UACC(NONE) NOTIFY(WEBSRV)
   PERMIT MVSADMIN.WLM.POLICY CLASS(FACILITY) ID(WEBSRV) ACCESS(UPDATE)
   PERMIT BPX.WLMSERVER CLASS(FACILITY) ID(WEBSRV) ACCESS(READ)
   SETROPTS CLASSACT(FACILITY) REFRESH
   ```

- Define WLM resources

  For information on specifying application environments and WLM translation classes, see "ApplEnv - Specify application environment for workload management" on page 375.

Related Information:
- For information on Web server modes of operation and WLM, see "Chapter 12. Managing your Web server" on page 103
- For more detailed information on WLM terminology and configuration, see OS/390 MVS Planning: Workload Management.

Step 8. Make TCP/IP configuration adjustments as needed

1. Put TCP/IP data configuration in a location such as
   
   ```
   SYSL.TCPPARMS(TCPDATA)
   ```

   so OS/390 UNIX sockets can find the name server.

2. In your TCP/IP profile, reserve the port that the server uses to access OS/390 UNIX (default is TCP 80 for a base server, TCP 443 for a secure server).
PORT

80 TCP OMVS ; Base Server accesses OMVS
443 TCP OMVS ; Secure Server accesses OMVS

3. You may want to use TCP/IP to autostart the server PROC named IMWEBSRV from the TCP/IP profile. For autostart to be successful, you must have OS/390 UNIX System Services started before you start TCP/IP.

4. As part of TCP/IP, you can activate SNMP. Use the started task PROC, or activate SNMP from the console. For more information on SNMP, see “Simple Network Management Protocol” on page 117. For information on the data set search order, see “Appendix I. TCP/IP reference” on page 471.

5. Use the TCP/IP onsllookup command to verify that name resolution is working correctly, for example:
   a. From an OMVS command line, type:
      
      onsllookup hostname
   
   b. Check that the returned IP address is correct:
      
      server : your.server.name
      address : IP_address

   For more information on the onsllookup command, see the eNetwork CS IP User’s Guide.

   Note: For the most current information on setting up and debugging TCP/IP-related problems, see the TCP/IP hints and tips section of the Web-based WebSphere Troubleshooter for OS/390.

   To link to the Troubleshooter, go to URL:


Step 9. Copy and customize the Web server PROC

Copy the IMWEBSRV sample provided in SYS1.SAMPLIB to a site procedure library, for example, SYS1.PROCLIB, and make the following changes as needed:

• If you changed the default names of the target libraries in IMWJALLO JOB supplied in SYS1.SAMPLIB, make the corresponding changes in the startup PROC.

• If you modify any RUNTIME options or set any server parameters, store the file containing these changes where you have your startup PROC.

To modify parameters and for an example of the IMWEBSRV PROC, see “IMWHTTPD program” on page 253.

The server needs to be assigned a user ID. This documentation assumes that the user ID of the server is WEBSRV. For the IMWEBSRV cataloged procedure to obtain control with the desired user identity, you must add an entry to the RACF started procedures table, module ICHRIN03. This entry defines the user ID and group ID that the IMWEBSRV address space will be assigned.

To assign the user ID, WEBSRV, to the RACF started procedures table, module ICHRIN03, see the following example:
Installing

DC CL8'IMWEBSRV' PROCEDURE NAME
DC CL8'WEBSRV' USERID (ANY RACF-DEFINED USER ID)
DC CL8'IMWEB' GROUP NAME OR BLANKS FOR USER'S DEFAULT GROUP
DC XL1'00' NOT TRUSTED
DC XL7'00' RESERVED

Note: Remember to increment the count of defined started procedures.

Or, you can add the PROC to the started task table. The following example provides a way to add the server PROC to the started task table using RACF commands:

RALTER STARTED IMWEBSRV.* STDATA(USER(WEBSRV))
SETPROPTS RACLIST(STARTED) REFRESH

If running with workload management, repeat the procedures described in this step for the workload management PROC, IMWIWM. See "IMWIWM PROC (workload management)" on page 260 for a sample workload management PROC.

Step 10. Configure installed files by running setup.sh

You must run the supplied OS/390 UNIX System Services shell script, setup.sh, to change the ownership of the installed files to your Web administration user ID, set up permissions, copy default configuration files, and configure languages.

This section assumes you have installed one Web server. If installing multiple Web servers, see the Redbook IBM HTTP Server for OS/390 Customization and Usage (SG24-5603).

To access Redbooks on the Web, go to URL:
http://www.ibm.com/redbooks/

Before running setup.sh

To run setup.sh, you must be a superuser or the SMP/E installer. This means you must have read and execute permission for setup.sh. The permission file for setup.sh should be 700. If someone other than the owner or superuser runs setup.sh, it may not execute correctly.

To execute setup.sh, you must be in the OS/390 UNIX shell and utilities (by typing in OMVS in TSO ready). You cannot execute setup.sh in ISHELL. If you do not use the default Web administration user ID and group (WEBADM and IMWEB), then you must change these defaults coded in setup.sh to the specific names you used.

Tasks performed by setup.sh

Running setup.sh sets up your environment by performing the following tasks:

- Configures the Web server for languages by symbolically linking files to the parent directory.
- Backs up existing configuration files.
- If the Web server is installed in the default install path, /usr/lpp/internet/, your configuration files are copied to the /etc directory. Otherwise, they are copied to the $install_path/etc directory. Setupcfg.sh handles this for you.
- Copies the Web server configuration file. The configuration file shipped in $install_path/samples/config/httpd.conf gets copied to /etc and $install_path/etc. Setupcfg.sh handles this for you.
- Copies and customizes the Web server environment. The ENVVARS file shipped in $install_path/samples/config/httpd.envvars gets copied to /etc and $install_path/etc.
• Copies the PICS configuration file. The PICS configuration file shipped in /install_path/samples/config/ics_pics.conf gets copied to /etc/ and /install_path/etc/.

• Copies the sample OS/390 data set configuration file (mvsds.conf). The OS/390 data set configuration file shipped in /install_path/samples/config/mvsds.conf gets copied to /etc/ and /install_path/etc/.

• Copies FastCGI configuration file. The FastCGI configuration file shipped in /install_path/samples/config/lgw_fcgi.conf gets copied to /etc/ and /install_path/etc/.

• Copies Web Traffic Express configuration files. There are two configuration files that you need for Web Traffic Express support: socks.conf and javelin.conf. The IBM Web Traffic Express for Multiplatforms User’s Guide V1.0 explains how to configure proxy caching features (javelin.conf) and flexible socksification (socks.conf). Before using the guide, review the information in “Setting up your proxy server” on page 163.

Note: If the socksification configuration file (socks.conf) already exists on the system, the server does not write over that file; it creates a new file named socks.conf.exp.

Hints and tips

Files are identical message when processing symbolic links: If you are migrating from a previous release of the Web server or if you run setup.sh more than once for a first-time install, you may see a message indicating that the source and target files are identical, for example:

FSUM8977 In: source "C/go98.jpg and target "./go98.jpg" are identical

You can ignore this message.

No such file or directory message when setting file properties: If you receive message EDC5129I for the httpd.conf.base, httpd.conf.secure, or servlet.conf file, you can ignore the message, for example:

chown: FSUM6188 stat file "samples/config/httpd.conf.base": EDC5129I No such file or directory
chown: FSUM6188 stat file "samples/config/httpd.conf.secure": EDC5129I No such file or directory
chown: FSUM6188 stat file "samples/config/servlet.conf": EDC5129I No such file or directory

Running setup.sh

You may want to re-direct setup.sh output messages to a file rather than the screen because they are easier to read. Otherwise, messages may scroll off the screen before you can read them. To re-direct the error messages to a file, enter:

setup.sh 2> filename.filetype

Run setup.sh from the OS/390 UNIX services command line by entering the following as two separate commands.

To set your configuration to English, enter:

cd /install_path/sbin
./setup.sh

To set your configuration to Japanese, enter:

cd /install_path/sbin
./setup.sh Ja_JP

Example of output messages:
*** (5697-D43) IBM HTTP Server 5.2 for OS/390 ***
Processing: default server install path is /usr/lpp/internet
    actual server install path is /usr/lpp/internet
    JDK install path is /J1.1.6
Processing: install English version of server
Processing: symbolic link of translatable files
Processing: symbolic link of files in ../server_root/Admin
Processing: symbolic link of files in ../server_root/admin-bin/webexec
Processing: symbolic link of files in ../server_root/Docs
Processing: symbolic link of files in ../server_root/pub
Processing: symbolic link of files in ../server_root/pub/reports
Processing: symbolic link of files in ../server_root/pub/reports/javelin
Processing: symbolic link of files in ../server_root/Counters/Fonts
Processing: symbolic link of files in ../samples/config
Processing: set file properties
Processing: copying configuration and environment files
Processing: backup, copy and customize configuration file (httpd.conf)
Processing: backup, copy and customize configuration file (httpd.envvars)
Processing: backup, copy and customize configuration file (javelin.conf)
Processing: backup, copy and customize configuration file (lgw_fcgi.conf)
Processing: backup, copy and customize configuration file (mvsds.conf)
Processing: backup, copy and customize configuration file (socks.conf)
List of backup configuration files in: /etc
/etc/httpd.conf.9812281439
/etc/httpd.envvars.9812281439
/etc/ics_pics.conf.9812281439

List of backup configuration files in: /usr/lpp/internet/etc
/usr/lpp/internet/etc/httpd.conf.9812281439
/usr/lpp/internet/etc/httpd.envvars.9812281439
/usr/lpp/internet/etc/ics_pics.conf.9812281439
/usr/lpp/internet/etc/lgw_fcgi.conf.9812281439
/usr/lpp/internet/etc/mvsds.conf.9812281439
/usr/lpp/internet/etc/servlet.conf.9812281439

Install Process Information:
  - Language installed: English
  - Server install path is: /usr/lpp/internet
  - JDK install path is: /J1.1.6
  - Server configuration files copied to: /etc and /usr/lpp/internet/etc
  - Server configuration files default path is: /etc
*** setup.sh HAS COMPLETED ***

When running setup.sh from /usr/lpp/internet/sbin, you might receive the following error messages:
*** (5697-C58) IBM HTTP Server 5.2 for OS/390 ***
- setup.sh was not found in the current working directory.
- Execute setup.sh from the install path for this file.
- Change to the install path and execute setup.sh, again.

For install_path, enter the root directory of your Web server installation. This value is either the default install path (/usr/lpp/internet) or the install path you specified during installation.

If you receive these messages, change the directory path to install_path/sbin and execute setup.sh again.
Step 11. Customize your Web server configuration file

httpd.conf

The sample Web server configuration allows you to access pages in the following directories:

- `install_path/server_root/pub` (default Front Page, Frntpage.html and other Web server pages)
- `install_path/server_root/Docs` (online documentation)
- `install_path/server_root/Admin` (Configuration and Administration Forms)

You can customize your Web server configuration using the Configuration and Administration Forms user interface or by editing the configuration file directly. For more information about these options, see “How do I configure the server?” on page 42.

Using a Web server user ID other than WEBADM

After installation, your Web server has one authorized user ID that can be used to access the Configuration and Administration Forms. By default, the authorized user ID is WEBADM.

If you selected a different user ID during Web server installation, edit the httpd.conf file and change all occurrences of WEBADM and webadm to the user ID you selected.

Replace WEBADM in the following statement:

```bash
Protect /admin-bin/* IMWAdmin WEBADM webadm
```

Change the owner of files in the following directory to the user ID you specified:

```bash
/install_path/server_root/admin-bin
```

The `install_path` is the root directory of your Web server installation. This value is either the default install path (`/usr/lpp/internet`) or the install path you specified during installation.

Changing default directories for logs and reports

The sample configuration file assumes you want to use the following directories for logging and reporting:

```bash
/install_path/server_root/logs
/install_path/server_root/pub/reports
```

Create these directories, or change the configuration file to point to an existing directory, before starting the IMWEBSRV_PROC. Consider creating a new HFS to mount for logging and reporting. (See “Chapter 10. Customizing logs and reports” on page 71 for information about logging and reporting.)

Step 12. Set up security

For configuration information and examples, see “Chapter 8. Setting up a secure server” on page 45.

Step 13. Set up proxy server support (optional)

To set up a proxy server, see “Chapter 15. Running your server as a proxy” on page 163.
Step 14. Install and configure WebSphere Application Server for servlet support (optional)

For planning considerations, see “WebSphere Application Server Java-based servlet engine” on page 9.

What’s next?

Now that your Web server is installed and customized, you can start the server, view the server’s Front Page, and change your server’s default configuration. To get started, go to “Chapter 4. Starting the server” on page 31.
Part 3. Using
Chapter 4. Starting the server

Performance considerations when starting the Web server ........................................ 31
Starting the server from the start-up PROC .................................................. 31
Starting the server using the httpd command ............................................. 31
Starting the server from the OS/390 UNIX shell ..................................... 31
Starting multiple instances of the server ..................................................... 32
Starting the server with Workload Management ...................................... 32
Letting WLM start the queue server address space .................................. 32
Prestarting the queue server address space .............................................. 33
Restarting the server .................................................................................... 33

After you finish installing the server, it starts using the default configuration settings. To configure the server to your exact specifications, see “Chapter 7. Getting started” on page 41.

Note: You do not have a secure server until you have created a key for secure network communications and received a certificate from a certificate authority (CA) who is designated as a trusted CA on your server. Security is described in “Chapter 8. Setting up a secure server” on page 43.

Performance considerations when starting the Web server

It is recommended that production Web servers be started using the method described in “Starting the server from the start-up PROC”. It may be difficult to achieve desired performance results if you are running the Web server from the OS/390 UNIX shell.

If you choose to start the Web server from a UNIX shell, review the information and recommendations in the Tuning section of the WebSphere Troubleshooter for OS/390.

To link to the Troubleshooter, go to URL:

Starting the server from the start-up PROC

You can start the server using the start-up PROC, IMWEBSRV. Make sure it is a member in a PROCLIB data set. Enter the following command from the OS/390 operator console:
s IMWEBSRV

For a sample IMWEBSRV PROC and variables used in the PROC, see “IMWHTTPD program” on page 253.

Starting the server using the httpd command

You can start the server using the httpd command. For more information, see “httpd command” on page 249.

Starting the server from the OS/390 UNIX shell

Note: Before using this method to start your Web server, see “Performance considerations when starting the Web server”.

31
Starting the server

To start the server, you must be in /usr/lpp/internet/sbin or provide the fully-qualified path on the httpd command (The following example assumes you are in /usr/lpp/internet/sbin.)

You can start the server from the OS/390 UNIX shell environment using the default server configuration file /etc/httpd.conf by entering:

```
httpd
```

This starts the server with the current configuration settings.

Starting multiple instances of the server

You can start multiple instances of the server, but each instance must listen on a separate port or IP address. All instances can be started by entering the following command from a command prompt:

```
httpd -r other_rule_file
```

For `other_rule_file`, enter the configuration file that specifies the individual port.

To start the secure server on the same machine but on a different port, enter the following command from a command prompt:

```
httpd -p current_port -sslport new_port
```

For `current_port`, enter the port number the server is using. For `new_port`, enter the port number of the new port for the server.

If Workload Management is enabled, unique subsystem names (-SN) must be defined. It is also recommended that you run under a unique server root and a unique access report root.

You can also use a start-up PROC or batch JCL to start multiple instances of the server by making similar changes.

Starting the server with Workload Management running

When Workload Management is started, temporary files are created and stored in /tmp. These files do not get deleted if the cancel command is used to stop the server. Therefore, use the stop command for proper clean-up.

To minimize any initial start-up costs (when running WLM), you can prestart your server address space once you receive the message indicating the server is ready.

Letting WLM start the queue server address space

- Under the Application Environment section of the WLM panels, you have specified the PROC which is to be started when there is work on that queue. A sample workload management PROC, IMWIWM, is shipped with the product. See “Workload Management Enablement for the Web server” on page 107 for more information about Workload Management panels.
- Specify the start parameters on the Workload Management ApplEnv panel:

```
IWMSN=&IWMSSNM, IWMAE=ApplEnv_name
```

- Start the QueueManager address space like you usually do, using JCL or a console command. When work is put on a queue, WLM will automatically start the corresponding PROC and start the address space for the queue server.
Prestarting the queue server address space

You can prestart the server by doing one of the following:

- The Queue Manager address space may be started by submitting the following job, or using the line command under the OS/390 UNIX shell: (-SN should be specified in the PARM statement to indicate the subsystem name used.)

  ```
  //RUNHTTP jobcard
  //STEP1 IMWEBSRV,
  //     ICSPARM='-SN WEBSN1'
  ```

- Enter the following console command:

  ```
  S IMWEBSRV,ICSPARM='-SN WEBSN1'
  ```

After the Queue Manager address space has been started, you need to start the Queue Server address space that correlates with the subsystem instance.

You can do this by doing one of the following:

- The Queue Server address space must be started by submitting the following job: (-SN and -AE should be specified in the PARM statement. The subsystem name specified should correlate with the one specified by the Queue Manager.)

  ```
  //RUNSRVR jobcard
  //STEP1 IMWEBSRV,
  //     ICSPARM='-SN WEBSN1 -AE WEBHTML'
  ```

- Enter the following console command:

  ```
  S IMWEBSRV,ICSPARM='-SN WEBSN1 -AE WEBHTML'
  ```

Note that WEBHTML is the name of the appropriate queue to be started.

Restarting the server

You can restart the server in the following ways:

- If you are using Workload Management and need to restart the server with a new configuration file, the workload manager marks the server space and completes work that is currently running. Workload Management starts a new server, finds the new configuration file, and identifies a new set of server spaces.

- Restart the server using the OS/390 operator console MODIFY command, F IMWEBSRV,APPL=-restart

- Send SIGHUP to the server process ID from the OS/390 UNIX services shell environment.

  Use the OS/390 UNIX services shell command `kill` or the server `wwwcmd` command to send SIGHUP to the server process. Refer to the OS/390 UNIX System Services Command Reference for more information about the `kill` command. For information on the `wwwcmd` command, see “wwwcmd command” on page 261.

- Use the server command `httpd` with the `-restart` option. For more information, see “httpd command” on page 249.

- Use the MVS cataloged procedure, IMWEBSRV, with the `-restart` option.

- Use the Restart Server button on the Remote Configuration and Administration forms.
Starting the server
Chapter 5. Viewing the Web server's default Front Page

After you install and start the Web server, you can use any compatible Web browser to view the server's default Front Page at URL:

```
http://your.server.name/FrontPage.html
```

For your.server.name, enter your server's fully qualified host name (for example, myhost.raleigh.ibm.com).

Note: For information on browser requirements and tips, see "Browser requirements" on page 4.

The default Front Page provides the following links:

- **CONFIGURATION AND ADMINISTRATION FORMS** to set up, configure, and administer the IBM HTTP Server
- **IBM HTTP SERVER WEB SITE** to find the most current information
- **HOW DO I GET STARTED?** to get help and examples for basic configuration tasks
- **INFORMATION TO HELP YOU** to access the online documentation

From the default Front Page of the Web server, you can link to online versions of the following information:

- *HTTP Server Planning, Installing, and Using*
- Application Server documentation
- *WebSphere Troubleshooter for OS/390*
- *IBM Web Traffic Express for Multiplatforms User's Guide V1.0*

The most current information is available on the HTTP Server Web site at URL:

```
http://www.ibm.com/software/websphere/httpservers/doc52.html
```

You can access Application Server documentation from the default Front Page of the Web server or from the Application Server Web site at URL:

```
```
Chapter 6. Stopping the server

You can stop the server if you know the WEBSRV user ID or have root authority. The server can be stopped the same way you stop any other OS/390 UNIX process:

- Use the OS/390 operator console STOP command, p IMWEBSRV.
- Use the OS/390 operator console CANCEL command, c IMWEBSRV.
- Send a process termination signal to the server process ID from the OS/390 UNIX shell environment. Both the SIGTERM signal and the SIGKILL signal can be sent, but the effect differs slightly. SIGTERM causes the server to perform a shutdown which allows outstanding client requests more time to complete before stopping. SIGKILL causes the server to terminate immediately.

Use the OS/390 UNIX MVS shell command kill or the server command wwwcmd to send SIGTERM or SIGKILL to the server process.

Refer to the OS/390 UNIX System Services Command Reference for more information about the kill command. For more information about the wwwcmd command, see "wwwcmd command" on page 261.

Note: The cancel command does not perform clean up of server files. Extra files may remain in the /tmp directory.
Stopping the server
Part 4. Basic Configuration
Chapter 7. Getting started

How do I start serving pages? ........ 41
How do I serve directory listings? ...... 41
How do I configure the server? ........ 42
Using the Configuration and Administration forms .................. 42
Editing the configuration file ........... 43
Syntax of configuration file records .... 44

How do I control access to server administration functions? ........ 44
Controlling access to the Configuration and Administration forms .... 44
Copying files ................................ 44
What files should I back up? .......... 44

After you have the server installed and running, you can start serving HTML pages and Web sites. As you become more familiar with the server, you will probably want to modify it by changing its configuration to meet your own particular needs.

This chapter helps you get started serving pages. It also shows you how to change your server’s configuration using the Configuration and Administration forms or by editing the configuration file, and how to control access to your server’s administration functions.

How do I start serving pages?

The HTTP Server can serve files of many file types. The most common are HTML files, also called HTML pages or documents, that contain text. They often imbed GIF or JPEG files to include graphic images.

If you point your browser to the host where you installed your server and you can get to the Front Page, your server is ready to go. Try going to http://your.server.name/ and make sure this works. Now you can add your own HTML documents, including welcome or home pages, and serve them.

Detailed instructions and information for this task are available in the online help available from the default Front Page of your server or from the Configuration and Administration Forms user interface. To access the online help, use either of the following methods:

- From the default Front Page of your server, click HOW DO I GET STARTED?, then click How do I? to display the list of tasks.
- From the Configuration and Administration Forms, click the question mark in the upper right hand corner, then click How do I? to display the list of tasks.

How do I serve directory listings?

When the server receives a directory request, it can serve either a welcome page from that directory or a formatted list of links to the files in the directory. The server’s default setting always returns a welcome page. You can use the Configuration and Administration forms to change this and serve directory listings for all directory requests or only for selected ones. You can also modify the contents of the directory listings you serve.

Detailed instructions and information for this task are available in the online help available from the default Front Page of your server or from the Configuration and Administration Forms user interface. To access the online help, use either of the following methods:
Getting started

- From the default Front Page of your server, click **HOW DO I GET STARTED?**, then click **How do I?** to display the list of tasks.
- From the Configuration and Administration Forms, click the question mark in the upper right hand corner, then click **How do I?** to display the list of tasks.

How do I configure the server?

You can configure your server either by using the Configuration and Administration forms or by editing the server's configuration file.

After installation, your server has one authorized user name that can be used to access the Configuration and Administration forms. By default, the authorized user name is WEBADM.

Using the Configuration and Administration forms

The server comes with Configuration and Administration forms. These forms are a combination of CGI programs and HTML forms that provide an easy way for you to configure your server or to view your server's current configuration settings.

Once the server is running, you can use the Configuration and Administration forms from any Web browser to configure your server.

To use the Configuration and Administration forms:

1. Disable caching on your browser. Also, if you are configuring your server remotely from a browser that uses that specific server as its proxy server, you should disable the proxy server setting on your browser.
2. Using your browser, go to the server’s Front Page by typing the following URL:

   http://your.server.name/

   where *your.server.name* is the fully qualified name of your host. For example, http://www.ibm.com/
3. Click **Configuration and Administration Forms**.
4. If you have not used the Configuration and Administration forms since starting your browser, you will be prompted for a user name and password.
   After you enter an authorized user name and password, you go to the Configuration and Administration forms page.
5. From the Configuration and Administration forms page, you can link to each of the input forms by clicking on the form name.
   When you go to a form, it is displayed with the current configuration values in its input fields. (If you haven’t changed your configuration since installation, these are the values that have been set in the shipped configuration file.)
6. From any form, enter information about how you want to configure that particular part of your server.
   Each form provides instructions to assist you in deciding what changes to make. For further information, click the question mark (?) in the upper right-hand corner of the page and choose from three kinds of help:
   - **Field descriptions** for information about each field on the form
   - **How do I?** for information about configuration tasks
   - **Index** for an index of all the help topics
7. After you fill in the form, click **Submit** to update the server configuration with the changes you made. The **Submit** button is located below the input fields on each form.

If you decide you do not want to use the changes you made to the form, click **Reset**. This returns the fields on the form to the values they had when you first came to the form.

8. If you clicked **Submit**, the server re-displays the form and provides a status message at the top of the page. This message tells you if your requested configuration changes completed successfully or, if not, which errors occurred. While the form is displayed, you can fix the errors and resubmit it.

9. After a successful completion, click the restart button in the upper right hand corner of the page to restart your server. When it is restarted, the server completes in-process requests, stops accepting requests, and reloads the changed configuration file. After the changed configuration file has been reloaded, the server accepts requests again.

**Note:** After submitting changes on the **Basic** configuration form, you must stop your server and start it again in order for your changes to take effect. In this case, restarting the server from the form is not effective.

For instructions on stopping and starting your server, see [“Chapter 4. Starting the server” on page 31](#).

**Editing the configuration file**

The other way to configure your server is by editing the configuration file.

By default, the configuration file is named `/etc/httpd.conf`. See [“httpd command” on page 249](#) for information on using a different configuration file when starting the server.

The configuration file is made up of statements called **directives**. You change your configuration by editing the configuration file, updating the directives, and saving your changes.

When you restart the server, your changes take effect, unless you changed one of the following directives:

- Port
- ApplEnv
- UserId
- PidFile
- BindSpecific
- SSLClientAuth
- SSLMode
- SSLPort
- NormalMode
- KeyFile
- imbeds
- ServerRoot

[“Appendix C. Configuration directives” on page 267](#) describes each of the configuration file directives.

If you change one of the directives in the above list, you must stop the server and start it again. For instructions on stopping and starting your server, see [“Chapter 4. Starting the server” on page 31](#).
Syntax of configuration file records

Each non-blank, non-comment record in the configuration file must contain ONE corresponding value, optionally followed by a comment. These records are not column-dependent and values are separated by one or more blanks. A directive and its values must be contained on a single line, as continuation to the next line is not allowed. The beginning of a comment is indicated by a pound sign (#). All characters from the pound sign to the end of the line are ignored. If either a pound sign or a blank needs to be specified for a directive, use the backslash (\) as an escape character before it, to indicate that it is not to be interpreted as it normally would. For example, if \# is found on a line, the server interprets this as a pound sign character, not the beginning of a comment, and continues reading characters. Also, if \ is found on a line, the server interprets this as a blank, not a value delimiter, and continues reading characters to build the value.

How do I control access to server administration functions?

Controlling access to the Configuration and Administration forms

During installation, you created a user ID (recommended name WEBADM) and group ID (recommended name IMWEB) to configure and administer the server. This user ID and password are authorized to access the Configuration forms.

If you intend to run the HTTP Server as a secure server (for example, with SSLmode on), certificates and certificates requests are electronically mailed to a certificate authority (CA) from this user ID.

Copying files

Be careful when you copy files to ensure you give the correct permissions to the WEBADM user ID and the IMWEB group ID. For example, WEBADM needs write permission to modify ACL files and, for a secure server, write permission to modify key database files (*.kdb).

If the user ID or group ID changes, you need to switch any ACL files to the new user ID or group ID. For a secure server, in addition to switching ACL files, you also need to switch key database files (*.kdb) and database password files (*.sth). If you are using %SAF% for security, you do not need to switch ACL files.

What files should I back up?

It is recommended that you periodically back up the following files:

- All configuration files
- Environment variables file (httpd.envvars)
- Password files
- Key database files
- Signed certificates and, optionally, certificate requests
- Group files
- ACL files
- Content files
Chapter 8. Setting up a secure server

Before you begin ............................................. 45
Tips and frequently asked questions ................. 46
IBM Key Management Utility ............................... 46
  What is the IBM Key Management Utility (IKEYMAN)? .... 46
  Can I use IKEYMAN for all security configuration tasks? 46
HTTP Server CA utility ....................................... 46
Security configuration examples ......................... 47
Security concepts, options, and support .............. 47
Security concepts .......................................... 47
  What is a secure communication? ......................... 47
  What is encryption? ........................................ 47
  What is authentication? .................................... 48
  What is a Public Key Infrastructure? ..................... 49
  What is the Secure Sockets Layer protocol? ............ 50
Security options for the HTTP Server .................. 51
  Options for protecting server resources ............... 51
  Options for setting up secure communications .......... 52
  SSL support for multiple IP addresses .................. 52
  Support for specifying the encryption level to be used. 53
  Stronger encryption option for financial and banking Web servers .......................... 53
  Certificate and URL requirements ....................... 53
  Browser requirements ..................................... 53
Certificate authorities supported by the HTTP Server ......................................................... 54
  Buying a certificate from an external CA provider 54
  Acting as your own CA ..................................... 55
Encryption support for the HTTP Server ............... 55
  Supported public-private key sizes .................... 56
  North American edition (U.S. and Canada) ......... 56
  International and France export editions ............ 56
  Supported SSL cipher specifications ................ 56
  North American edition (U.S. and Canada) .......... 56
  International export edition .......................... 57
  France export edition .................................... 57
  Hardware encryption ...................................... 57
Setting up access control and secure connections ...... 57
  Accessing the online help examples .................... 58
  Setting up protection for server resources .......... 58
    Step 1. Accept the Web server default user ID %\%CLIENT\%\% ........................................ 58
    Step 2. Create a URL request template for the resource that needs to be protected and specify protection settings for that request ....... 58
    Step 3. Set up mapping rules to route the request to the correct resource ............... 59
    Step 4. Click Restart Server to use your new configuration settings ..................... 59
  Using group files ......................................... 60
  Using Access Control List files ......................... 60
Setting up secure connections using an external CA .... 61
  Step 1. Create a new key database and specify a key database password ..................... 61
  Step 2. Create a public-private key pair and certificate request ............................. 61
  Step 3. Receive your CA-signed server certificate into the key database .................. 61
  Step 4. Change server security defaults (optional) .................................................... 62
  Step 5. Register your key database with the server ................................................... 62
Setting up secure connections using self-signed certificates ..................................................... 62
  Step 1. Create your server key database and certificate request ................................ 63
  Step 2. Create your CA key database and certificate .................................................. 63
  Step 3. Create and sign your server certificate ......................................................... 63
  Step 4. Store your CA certificate in the server key database ........................................ 63
  Step 5. Receive your CA-signed server certificate .................................................................. 64
  Step 6. Change server security defaults (optional) ...................................................... 64
  Step 7. Register the server key database with the server ............................................. 64
  Setting up SSL support for multiple IP addresses .......... 65
Changing the server’s default encryption settings .......... 65

Before you begin

Before setting up your secure server, review the planning considerations in the following sections:

- "Access control with RACF and other SAF-based security products” on page 4
- "OS/390 UNIX System Services considerations” on page 7
- "Protection setup considerations” on page 8

If you are migrating from a previous release of the Web server, also review the following sections:

- "Configuration and key management changes” on page 13
- "Certificate authority utility changes” on page 13
IBM Key Management Utility

What is the IBM Key Management Utility (IKEYMAN)?
You do not have a secure network connection until you have created a key for
secure network communications and received a certificate from a certificate
authority (CA) who is designated as a trusted CA on your server. Use IKEYMAN
to create key databases, public-private key pairs, and certificate requests. If you are
acting as your own CA, you can use IKEYMAN to create self-signed certificates.

If you act as your own CA for a private Web network, you have the option to use
the server CA utility to generate and issue signed certificates to clients and servers
in your private network. For more information, see “HTTP Server CA utility”.

Can I use IKEYMAN for all security configuration tasks?
You can use IKEYMAN only for configuration tasks related to public-private key
creation and management. You cannot use IKEYMAN for configuration options
that update the server configuration file, httpd.conf. For options that update the
server configuration file, you must use either the Configuration and
Administration Forms or edit the configuration file directly.

Migration note: Prior to Release 5.0, you could create keys and certificate requests
using the Configuration and Administration Forms. This option was removed in
Release 5.0 for security reasons. You now use IKEYMAN to create keys and
certificate requests. The Security Configuration and Administration Forms are now
used to perform the following tasks:
• Enable or disable SSL transactions; by default, SSL transactions are enabled.
• Change the server’s default SSL port of 443
• Specify the type of SSL client authentication to use; by default, client
authentication is not used.
• Change the server’s default timeout duration for SSL sessions
• Change the server’s default encryption settings; by default, the server will use
the strongest encryption level supported by both the client and the server.
• Register key databases you have added using IKEYMAN
• Specify another default key database for the server
• Associate a certificate with a specific IP address, if using multiple IP addresses
on the server

HTTP Server CA utility
The HTTP Server provides a utility that simplifies administering a certificate
authority (CA) for a private network. This utility is called HTTP Server CA and
consists of a set of HTML forms and servlets.

Before you can use HTTP Server CA, you must create a CA key database file
named cakey.kdb using IKEYMAN, create a self-signed CA certificate, and
designate the CA certificate as trusted on your server.

Related Information:
• For migration considerations, see “Certificate authority utility changes” on
page 13.
• For an example, see “Setting up secure connections using self-signed certificates”
on page 62.
Security

For more information on CA utility requirements and tasks, see "Appendix A. Certificate authority utility" on page 233.

Security configuration examples

Before you use IKEYMAN or HTTP Server CA, we recommend that you review the following examples:

- "Setting up secure connections using an external CA" on page 61
- "Setting up secure connections using self-signed certificates" on page 62

Security concepts, options, and support

For an overview of security concepts, options, and support, see the following sections:

- "Security concepts"
- "Security options for the HTTP Server" on page 51
- "Certificate authorities supported by the HTTP Server" on page 54
- "Encryption support for the HTTP Server" on page 55

Security concepts

This section provides an overview of security concepts. If you are already familiar with basic security concepts and want to get started, go to "Setting up access control and secure connections" on page 57.

What is a secure communication?

The rapid growth of electronic commerce over the Internet has led to an increasing demand for secure network communications. In addition, intra-company communications over private networks often contain confidential information that needs to be protected.

A secure network communication has the following characteristics:

Access control

Resources can be protected and accessed only by authorized parties. Restricting access on the basis of passwords, IP address, host names, or SSL client authentication ensures access control.

Authenticity

You know who you are talking to and that you can trust that person. Authentication, using digital signature and digital certificates, ensures authenticity.

Information integrity

Messages are not altered while being transmitted. Without information integrity, you have no guarantee that the message you sent matches the message received. Digital signature ensures integrity.

Privacy and confidentiality

Information conveyed from party to party during a transaction remains private and cannot be read even if it gets into the wrong hands. Encryption ensures privacy and confidentiality.

What is encryption?

Encryption in its simplest form is scrambling a message so that it cannot be read until it is unscrambled later by the receiver. The sender uses an algorithmic pattern...
Security

(or key) to scramble (or encrypt) the message. The receiver has the decryption key. Encryption ensures privacy and confidentiality in transmissions sent over the Internet.

There are two kinds of keys that can be used for encryption:

Asymmetric keys

With asymmetric keys, you create a key pair. The key pair is made up of a public key and a private key, which are different from each other. The private key holds more of the secret encryption pattern than the public key. Your private key should not be shared with anyone.

The server uses its private key to sign messages to clients. The server sends its public key to clients so that they can encrypt messages to the server, which the server decrypts with its private key. Only you can decrypt a message that has been encrypted with your public key, because only you have the private key. Key pairs are stored in a key database which is protected by a password.

Symmetric keys

Symmetric keys follow an age-old model of the sender and receiver sharing some kind of pattern. This same pattern is then used by the sender to encrypt the message and by the receiver to decrypt the message.

The risk involved with symmetric keys is that you have to find a safe transportation method to use when sharing your secret key with the people you want to communicate with.

The Secure Sockets Layer (SSL) protocol uses both asymmetric and symmetric key exchange. Asymmetric keys are used for the SSL handshake. During the handshake the master key, encrypted with the receiver’s public key, is passed from the client to the server. The client and server make their own session keys using the master key. The session keys are used to encrypt and decrypt data for the remainder of the session. Symmetric key exchange is used during the exchange of the cipher specification (or encryption level) used.

To send its public key to clients, the server needs a digital certificate. This certificate is issued by a certificate authority (CA) who verifies the identity of the server.

Related Information:

- For more information about the SSL protocol, see "What is the Secure Sockets Layer protocol?" on page 50.
- For details on supported key sizes, see "Encryption support for the HTTP Server" on page 55.

What is authentication?

Authentication is the process used to verify identity, so that you can ensure that others are who they say they are. There are two ways that the server uses authentication:

Digital signature

A digital signature is a unique mathematically computed signature that ensures accountability. A digital signature is similar to a credit card with your picture on it. To verify the identity of the person sending you a message, you look at the sender’s digital certificate.
Digital certificate

A digital certificate (or digital ID), is like a credit card with a picture of the bank president with his arm around you. A merchant trusts you more because not only do you look like the picture on the credit card, the bank president trusts you, too.

You base your trust for the authenticity of the sender on whether you trust the third party (a person or agency) that certified the sender. The third party who issues digital certificates is called a certificate authority (CA) or certificate signer.

A digital certificate contains:

- The public key of the person being certified
- The name and address of the person or organization being certified, also known as the Distinguished Name
- The digital signature of the CA
- The issue date
- The expiration date

You enter your Distinguished Name as part of requesting a certificate. The digitally-signed certificate includes not only your own Distinguished Name but the Distinguished Name of the CA.

You can request one of the following certificates:

- A server certificate to do commercial business on the Internet from VeriSign or some other CA. For a list of supported CAs, see "Buying a certificate from an external CA provider" on page 53.
- A server certificate that you create for your own private Web network. For more information, see "Acting as your own CA" on page 53.

CAs broadcast their public key and Distinguished Name bundled together so that people will add them to their Web servers and browsers as a trusted CA certificate. When you designate the public key and certificate from a CA to be a trusted CA certificate, this means that your server trusts anyone who has a certificate from that CA. You can have many trusted CAs as part of your server. The HTTP Server includes several default trusted CA certificates. You can add or remove trusted CAs as needed using the IBM Key Management Utility included with your server.

In order to communicate securely, the receiver in a transmission must trust the CA who issued the sender’s certificate. This is true whether the receiver is a Web server or browser. When a sender signs a message, the receiver must have the corresponding CA-signed certificate and public key designated as a trusted CA certificate.

Related Information:

- For more information on CAs and a list of default trusted CAs, see "Certificate authorities supported by the HTTP Server" on page 54.

What is a Public Key Infrastructure?

A Public Key Infrastructure (PKI) is a system of digital certificates, certificate authorities, registration authorities, certificate management service, and X.500 directories that verify the identity and authority of each party involved in any transaction over the Internet. These transactions may be financial or may involve any operation where identity verification is required, such as confirming the origin of proposal bids or author of e-mail messages.
A PKI supports the use of certificate revocation lists (CRLs). A certificate revocation list is a list of certificates that have been revoked. CRLs provide a more global method for authenticating a client’s identity by certificate, and can also be used to verify the validity of trusted CA certificates.

CRLs and trusted CA certificates are stored and retrieved from an X.500 directory server. The protocols used for storing and retrieving information from an X.500 directory server are Directory Access Protocol (DAP) and Lightweight Directory Access Protocol (LDAP). The HTTP Server supports LDAP.

Information can be distributed on multiple directory servers over the Internet and intranets, thus allowing an organization to manage certificates, trust policy, and CRLs from either a central location or in a distributed manner. This makes the trust policy more dynamic because trusted CAs can be added or deleted from a network of secure servers without having to reconfigure each of the servers.

Related Information:
- For more information on LDAP support, see “Chapter 14. Retrieving LDAP information” on page 155.

What is the Secure Sockets Layer protocol?
The Secure Sockets Layer (SSL) protocol was developed by Netscape Communications Corporation. SSL ensures that data transferred between a client and a server remains private. It allows the client to authenticate the identity of the server. SSL Version 3 is required to authenticate the identity of a client.

Once your server has a digital certificate, SSL-enabled browsers like Netscape Navigator and Microsoft Internet Explorer can communicate securely with your server using SSL. With SSL, you can easily establish a security-enabled Web site on the Internet or on your private intranet. A browser that does not support HTTP over SSL will not be able to request URLs using HTTPS. The non-SSL browsers will not allow submission of forms that need to be submitted securely.

SSL uses a security handshake to initiate a secure connection between the client and the server. During the handshake, the client and server agree on the security keys they will use for the session and the algorithms they will use for encryption. The client authenticates the server; optionally, the server can request the client’s certificate. After the handshake, SSL is used to encrypt and decrypt all of the information in both the https request and the server response, including:
- The URL the client is requesting
- The contents of any form being submitted
- Access authorization information like user names and passwords
- All data sent between the client and the server

HTTPS is a unique protocol that combines SSL and HTTP. You need to specify https:// as an anchor in HTML documents that link to SSL-protected documents. A client user can also open a URL by specifying https:// to request an SSL-protected documents.

Because HTTPS (HTTP + SSL) and HTTP are different protocols and use different ports (443 and 80, respectively), you can run both SSL and non-SSL requests at the same time. As a result, you can choose to provide information to all users using no security, and specific information only to browsers who make secure requests. This
is how a retail company on the Internet can allow users to look through the merchandise without security, but then fill out order forms and send their credit card numbers using security.

Related Information:
- For more information on SSL client authentication and SSL Version 3, go to URL http://home.netscape.com/eng/ssl3/
- For more information about the SSL protocol and related security topics, go to URL http://home.netscape.com/newsref/std/.

Security options for the HTTP Server

The HTTP Server provides:
- “Options for protecting server resources” on page 52
- “Options for setting up secure communications” on page 52
- “SSL support for multiple IP addresses” on page 53
- “Support for specifying the encryption level to be used” on page 53
- “Stronger encryption option for financial and banking Web servers” on page 53

Options for protecting server resources

There are three protection options you can use to control access to the resources on your server:

- Password protection
  With this type of protection, you create user names for the requesters who are authorized to access your protected resources. When you define each user name, you assign a password for that user. You can assign the same password to multiple users. You can also set up user group files. For To classify users into groups, see “Using group files” on page 63.
  For OS/390, the server provides options to use an existing MVS user ID and password, and to allow a SAF controlled operating system (for example, RACF) to perform password verification.

- IP address or host name protection
  With this type of protection, you activate protection rules for a request based on the address the request comes in on or the host for which the request is made. For example, you might want to specify that a request for /schedules.html/ received on address 9.67.106.79 (or hostA) is protected by protection rules different from the rules used if the same request was received on address 9.83.100.45 (or hostB).

- Secure Sockets Layer (SSL) client authentication
  SSL Version 3 is required for client authentication. If you set up your server for SSL client authentication, the server will request a certificate from all clients making an https request.
  The server will establish a secure connection if the client has a valid certificate or if the client has no certificate at all. The server will deny the request if the client has a certificate that has expired or if the certificate is signed by a CA who is not designated as a trusted CA on the server.

You can use one protection option or a combination of options.

Related Information:
- For configuration instructions, see “Setting up protection for server resources” on page 58.
Options for setting up secure communications

The HTTP Server uses the following industry standards:

- Secure Sockets Layer (SSL) protocol for connection security
- Public key cryptography from RSA Data Security, Inc. for encryption and authentication
- X.500 and X.509 for certificate authentication and processing as part of an enterprise Public Key Infrastructure (PKI)

You do not have a secure network connection until you have created a key for secure network communications and received a certificate from a certificate authority (CA) who is designated as a trusted CA on your server. Use the IBM Key Management Utility (IKEYMAN), provided with your server, to create key databases, public-private key pairs, and certificate requests. If you are acting as your own CA, you can use IKEYMAN to create self-signed certificates.

Related Information:
- For configuration instructions, see the following examples:
  - “Setting up secure connections using an external CA” on page 61
  - “Setting up secure connections using self-signed certificates” on page 62
- For information on concepts and terminology, see
  - “What is the Secure Sockets Layer protocol?” on page 50
  - “What is encryption?” on page 47
  - “What is authentication?” on page 48
  - “What is a Public Key Infrastructure?” on page 49

SSL support for multiple IP addresses

A secure network connection between a server and a browser requires an SSL session between them. During an SSL session with the server, a digital certificate (or digital ID) is sent to the browser. The certificate contains the host name of the server.

If your server is configured with multiple IP addresses, you can send different certificates from each IP address. If you have a different host name assigned to each IP address, you can configure SSL to send a different certificate for each host name. For example, if your server is configured to have the two IP addresses 125.25.116.87 and 125.25.116.89 with DNS entries of www.Mall.com and www.SuperShopper.com, respectively, each host name/IP address combination could have a separate key and certificate. Because the IP session connections for www.Mall.com and www.SuperShoper.com are coming in on different IP addresses for the requests, the server can serve the corresponding certificate for each host name.

Where there are multiple host name/IP address combinations, we recommend a separate certificate for each combination. Separate certificates prevent the browser pop-up warning that the host name of the request does not match the host name presented in the certificate during the initialization of the SSL session.

Related Information:
- For configuration instructions, see “Setting up SSL support for multiple IP addresses” on page 65.
- For more information on digital certificates, see “What is authentication?” on page 48.
Support for specifying the encryption level to be used

By default, the HTTP Server uses the strongest level of encryption which is common to both the client and the server.

This configuration option enables the server administrator to specify the level of encryption to be used. The administrator can use a lower level of encryption, if appropriate. For example, you might want to use a lower encryption level to improve system performance.

Related Information:
- For configuration instructions, see “Changing the server’s default encryption settings” on page 65.
- For information on using S/390 cryptographic hardware, see “Hardware encryption” on page 57.

Stronger encryption option for financial and banking Web servers

When designated as a financial or banking Web server, the North American edition of the HTTP Server can exploit the strong encryption capabilities in the domestic and international versions of Netscape Navigator 4.x and Microsoft Internet Explorer 4.x. To use this function, you must purchase a special digital certificate from VeriSign called a Global Server ID.

For regular Web transactions, the Netscape and Microsoft export browsers can use 40-bit encryption only for Secure Sockets Layer (SSL) transactions. However, when the server uses a VeriSign Global Server ID for its SSL certificate, the export browsers can use stronger levels of encryption of 128-bits or greater. This enables a server with a Global Server ID to communicate at the highest SSL encryption level with both domestic and international versions of the Netscape and Microsoft browsers.

International financial and banking customers of the HTTP Server who want to use this function must contact IBM for an export license to obtain and use the North American edition of the HTTP Server.

Certificate and URL requirements

For an export browser to establish an SSL connection using 128-bit encryption:
- Use a Global Server ID from VeriSign so that the server’s certificate contains the required banking extensions. To obtain a Global Server ID, contact VeriSign at http://digitalid.verisign.com/server/global/index.html.
- Ensure that the client’s browser has designated the server’s certificate as a trusted CA.
- Ensure that the URL entered by the user includes the fully qualified server name that appears in the certificate, for example, bank2.raleigh.ibm.com.

Browser requirements

<table>
<thead>
<tr>
<th>Browser</th>
<th>Domestic</th>
<th>Export</th>
</tr>
</thead>
<tbody>
<tr>
<td>Netscape Navigator</td>
<td>All versions work when the requirements listed in “Certificate and URL requirements” are met.</td>
<td>Version 4.04 is the only version that supports 128-bit encryption.</td>
</tr>
</tbody>
</table>
Security

Table 3. Global server ID browser requirements (continued)

<table>
<thead>
<tr>
<th>Browser</th>
<th>Domestic work when the requirements listed in “Certificate and URL requirements” on page 53 are met.</th>
<th>Export Version 4.0, upgrade 4.72.3110.8, is the only version that supports 128-bit encryption. To verify the upgrade number, click Help, then About Internet Explorer.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Internet Explorer</td>
<td>All versions work when the requirements listed in “Certificate and URL requirements” on page 53 are met.</td>
<td>Version 4.0, upgrade 4.72.3110.8, is the only version that supports 128-bit encryption. To verify the upgrade number, click Help, then About Internet Explorer.</td>
</tr>
</tbody>
</table>

For the most current information on encryption support and browser requirements, see the HTTP Server Web site at URL:

http://www.ibm.com/software/websphere/httpservers/doc52.html

Certificate authorities supported by the HTTP Server

Your public key must be associated with a digitally signed certificate from a certificate authority (CA) who is designated as a trusted root CA on your server.

There are two ways to obtain a certificate:

- "Buying a certificate from an external CA provider"
- "Acting as your own CA" on page 55

Buying a certificate from an external CA provider

You can buy a signed certificate by submitting a certificate request to a CA provider. The HTTP Server supports the following external certificate authorities:

- VeriSign
  For more information, go to URL http://www.verisign.com/.
- Thawte
  For more information, go to URL http://www.thawte.com/.

By default, the following are designated as trusted CAs on the HTTP Server:

- VeriSign Class 1 Public Primary CA
- VeriSign Class 2 Public Primary CA
- VeriSign Class 3 Public Primary CA
- VeriSign Class 4 Public Primary CA
- VeriSign Test CA
- RSA Secure Server CA (from VeriSign)
- Thawte Personal Basic CA
- Thawte Personal Freemail CA
- Thawte Personal Premium CA
- Thawte Premium Server CA
- Thawte Server CA

Notes:

1. If you are using certificate revocation lists (CRLs) for client authentication, you must purchase CA software from the IBM Registry and issue your own certificates. For more information, go to URL http://www.ibm.com/software/commerce/registry/.

2. Financial and banking institutions have the option to purchase a special digital certificate that allows export editions of the server to use encryption levels of 128-bits or greater. For more information, see “Stronger encryption option for financial and banking Web servers” on page 53.
Acting as your own CA

If you act as your own CA, you can sign your own or anyone else’s certificate requests. This is a good choice if you only need the certificates within your private Web network and not for outside Internet commerce. Clients must have browsers, such as Netscape Navigator or Microsoft Internet Explorer, that can receive your CA certificate and designate you as a trusted CA.

To act as your own CA, you can use your server key management utilities (IKEYMAN and HTTP Server CA), or you can purchase certificate authority software from a CA provider.

Note: If you expect to administer over 250 certificates, you may want to consider purchasing software from a CA provider. This limit is based on the number of certificate requests that can be stored in the CA key database. After storing 250 certificate requests, performance of the utility is very slow.

- For an example using IKEYMAN and HTTP Server CA, see Setting up secure connections using self-signed certificates on page 62.
- If you purchase certificate authority software, the HTTP Server supports the following CA software:
  - IBM Registry
    If you are using certificate revocation lists (CRLs) for client authentication, you must use CA software from the IBM Registry. For more information, go to URL http://www.ibm.com/software/commerce/registry/.
  - Entrust Demo Certificates and Entrust WEBCA
    For more information, go to URL http://www.entrust.com/.
  - Netscape Certificate Server
    For more information, go to URL http://www.netscape.com/.
  - XCert
    For more information, go to URL http://www.xcert.com/.
  - Any X.509–compliant certificate authority

Encryption support for the HTTP Server

The U.S. Government and foreign import rules regulate products used for encryption and prohibit their export unless their key size is strictly limited. This section summarizes the key sizes and Secure Sockets Layer (SSL) cipher specifications for the North American and export editions of the HTTP Server. Customers in the United States and Canada can install the North American or export editions of the HTTP Server.

Note: Financial and banking institutions have the option to purchase a special digital certificate that allows export editions of the server to use encryption levels of 128-bits or greater. For more information, see Stronger encryption option for financial and banking Web servers on page 53.

As U.S. export laws and foreign import rules are updated, the supported key lengths and cipher specifications are subject to change. For the latest information, refer to the HTTP Server Web site at URL:

http://www.ibm.com/software/websphere/httpservers/doc52.html
Supported public-private key sizes

Key size refers to the key lengths used for the public and private keys in an asymmetric key exchange. For an explanation of asymmetric keys, see "What is encryption?" on page 47.

North American edition (U.S. and Canada)
The North American edition of the server can:
• Generate keys from 512-1024 bits
• Encrypt data with symmetric keys from 40-168 bits
• Sign data with keys from 512-1024 bits
• Check signatures with keys from 512-1024 bits

International and France export editions
The International and France export editions of the server can:
• Generate keys with a key size of 512 bits
• Encrypt data with symmetric keys from 40-56 bits (International edition); the France edition supports 40-bit keys only
• Sign data with 512-bit keys
• Check signatures with keys from 512-1024 bits

Supported SSL cipher specifications

The SSL cipher specification tells you which data encryption algorithm and key size are used; for SSL V3, the hashing algorithm is included. Key size refers to the key length used for the keys in a symmetric key exchange. For an explanation of symmetric keys, see "What is encryption?" on page 47.

For example, cipher specification Triple-DES SHA uses the Triple-DES encryption algorithm and the SHA hashing algorithm. For more information on cipher specifications, go to URL http://home.netscape.com/eng/ssl3/.

By default, the HTTP Server uses the strongest level of encryption which is common to both the client and the server. To change the default setting, see "Changing the server’s default encryption settings" on page 65.

The HTTP Server supports the following SSL cipher specifications (listed from strongest to weakest):

North American edition (U.S. and Canada)
• SSL V2
  – Triple-DES
  – RC4 (128 bit)
  – RC2 (128 bit)
  – DES
  – RC4 (40 bit)
  – RC2 (40 bit)
• SSL V3
  – Triple-DES SHA
  – RC4 SHA (128 bit)
  – RC4 MD5 (128 bit)
  – DES SHA
  – RC4 MD5 (40 bit)
  – RC2 MD5 (40 bit)
International export edition

- SSL V2
  - DES
  - RC4 (40 bit)
  - RC2 (40 bit)
- SSL V3
  - DES SHA
  - RC4 MD5 (40 bit)
  - RC2 MD5 (40 bit)

France export edition

- SSL V2
  - RC4 (40 bit)
  - RC2 (40 bit)
- SSL V3
  - RC4 MD5 (40 bit)
  - RC2 MD5 (40 bit)

Hardware encryption

IBM’s Common Cryptographic Architecture (CCA) and the RSA toolkit exploit S/390 hardware available on IBM S/390 Parallel Enterprise Servers (G3, G4, and G5), Muprise 2000, and IBM S/390 Application StarterPak. Triple-DES hardware encryption is available on G4 and G5 servers.

If the server is installed on a machine that supports CCA and the RSA toolkit, the HTTP Server will use the hardware to perform DES and Triple-DES encryption. This will improve the performance of SSL sessions between clients and servers that use DES or Triple-DES during the SSL handshake.

If you want the hardware to perform these cryptographic operations, configure the Web server to try DES or Triple-DES first during the SSL handshake with the client. For configuration instructions, see “Changing the server’s default encryption settings” on page 65.

You can optionally turn on the following message which indicates whether hardware encryption is being used for Web server encryption:

```
secinit.c: Setting up security, state=0
Crypto Hardware {is|is not} being used by SSL.
```

This message will be written to stderr and to the -vv trace, if tracing is turned on. To enable this message, add the GSK_SSL_HW_DETECT_MESSAGE environment variable using either of the following methods:

- Edit the Web server start-up PROC, IMWEBSRV, and add:
  ```
  LEPARM='ENVAR("GSK_SSL_HW_DETECT_MESSAGE=1")'
  ```
- Edit the Web server httpd.envvars file and add:
  ```
  GSK_SSL_HW_DETECT_MESSAGE=1
  ```

Setting up access control and secure connections

This section summarizes how to protect your server resources and set up secure connections to and from your server.
**Accessing the online help examples**

To access the online help examples, use either of the following methods:

- From the default Front Page of your server, click **HOW DO I GET STARTED?**, then click **How do I?** to display the list of tasks.
- From the Configuration and Administration Forms, click the question mark in the upper right hand corner, then click **How do I?** to display the list of tasks.

**Setting up protection for server resources**

This quick start example shows how to set up protection for server resources using the Configuration and Administration Forms.

**Step 1. Accept the Web server default user ID %%CLIENT%%**

The sample configuration provided with the Web server protects the default home page and samples with %%CLIENT%% so that only users known to the system can use the server until you explicitly grant access to other users. For more information on %%CLIENT%% and other ID options, see "Special access control user IDs” on page 20.

**Step 2. Create a URL request template for the resource that needs to be protected and specify protection settings for that request**

When a client sends a request for a resource to the Web server, the server uses the content of the request to activate protection for the resource. A request is the part of a URL that follows the server’s host name. For example, the host name of the IBM Web site is www.ibm.com. If a client enters URL http://www.ibm.com/Products, the request is /Products.

In this example, Company A wants to ensure that requests for the budgets.html file located in the financial directory are protected. To restrict access to the budgets file, Company A needs to create a URL request template and specify protection settings for the request, /financial/budgets.html, using these steps:

1. From the default Front Page of the Web server, click Configuration and Administration Forms.
2. Click Access Control, then click Document Protection.
3. Accept the default position in the list, Insert before.
4. For URL request template, enter /financial/*. This restricts access to all files in the financial directory, including budgets.html.
5. For Authentication options, click Password or user/group authentication. To use the SSL client authentication option, you must set up secure communications to and from your server. For instructions, see "Setting up secure connections using an external CA” on page 61 or "Setting up secure connections using self-signed certificates” on page 62.
6. Click Named protection setup and enter **BUDGETS** for the protection setup name. If you define the protection setup in-line, you can only use the settings for this one URL request template. A named protection setup is recommended because it can be used for other URL request templates.
7. Leave the Server IP Address or host name field blank. This field is used if your server has multiple IP addresses. For more information on using multiple IP addresses, see "Chapter 16. Running your server with multiple IP addresses or virtual hosts” on page 163.
8. Click Submit.
9. For performance reasons, we do not select the option to Allow ACL files to override protection settings. For more information on ACL files, see "Using Access Control List files" on page 60.

10. Enter your Password Authentication settings:
    • For Protection realm, enter restricted.
      Protection realm is a name that you select to identify this protection setup to requesters who will be entering a user ID and password. The name does not need to be a real machine name. When the server sends a request for user name and password, it also includes the name you specify for protection realm. Protection setups that use the same password should use the same name for protection realm.
    • Enter %%SAF%% for Password file.
      When %%SAF%% is specified, password verification is performed using your SAF-based security product, for example, RACF.
    • Leave the Group file field blank.
      This field is used only if you specify group names in the Permissions section.
      Group files cannot contain RACF user IDs. For more information on group files, see "Using group files" on page 60.
    • For Permissions, give Read permission only to those users who are authorized to access budget information. Enter each user name, separated by a comma, in the Users with Read permission (GET and POST) field.

11. Click Submit. The new URL request template appears in the list.

12. Click Configuration Page to configure additional options.

**Step 3. Set up mapping rules to route the request to the correct resource**

Before your Web server can route a request to a specific resource, you need to configure mapping rules for the server to use. To route requests for the budgets.html file to the correct directory (financial):

1. Click Request Processing, then click Request Routing.
2. Accept the default position in the list, Insert before.
3. For URL request template, enter /budgets.html/.
4. Click Submit. You will see a message that the requested configuration changes have been completed successfully.

**Step 4. Click Restart Server to use your new configuration settings**

**Related information:**

- For an overview of protection options, see "Options for protecting server resources" on page 51.
- For RACF considerations, see "Access control with RACF and other SAF-based security products" on page 4.

**Advanced options:**

- You can configure many protection options by editing the server configuration file directly. For instructions, see "Access control - Set up access control for the server" on page 273.
- For information on mapping directives, see "Resource mapping - Redirect URLs" on page 359.
- For information on using an LDAP server to authenticate user or group information, see "Using LDAP to protect files" on page 158.
Using group files

Use group files to classify users into groups. Protection setups can point to a Web server group file. The protection setup can then use the groups defined in the server group file on mask subdirectives. If a protected directory contains an Access Control List (ACL) file, the rules in the ACL file can also use the groups defined in the server group file.

**Note:** Group files cannot contain RACF user IDs.

You can create as many server group files as you need. Create each in a separate text file. Within the server group file, each line contains a group definition using the following format:

```
groupname : user1[,user2[,user3...]]
```

groupname

Any name you want to use to identify the group you are defining. This name can be used on:

- Mask subdirectives within protection setups that point to the server group file. (The mask subdirectives are DeleteMask, GetMask, Mask, PostMask, and PutMask.)
- Access rules within ACL files on directories that are protected with a protection setup that points to the server group file.
- Subsequent group definitions within the same server group file.

```
user1[,user2[,user3...]]
```

This can actually be any combination of user names, group names, and address templates. Separate each item with a comma.

For user names to be valid, they must be defined in the password file that the protection setup points to. You create and maintain password files using the htadm command. Note that the htadm command cannot be used to manage SAF user IDs and passwords.

Group names must be defined on previous group definition statements in the same group file.

**Example:**

```
ducks : (webfoot,billface)@96.96.3.1,swandude
geese : goosegg,bagel@(walden.pond.*,123.*.*.*)
flock : ducks,geese
webbed : All@water.fowl.*
```

In this example, notice that once the groups named ducks and geese are defined, they can be included as part of the group named flock.

Using Access Control List files

Use Access Control List (ACL) files to limit access to specific files on a protected directory. Note that using ACL files can impact performance.

Each protected directory can have only one ACL file. The ACL file must be named .www_acl and be present on the protected directory. Normally, the mask subdirectives in the protection setup define the first level of access control and then the ACL file further limits access to individual files. However, if you want all control to come from the ACL file, use the ACLOverride subdirective with a value of On in the protection setup. This causes the mask subdirectives in the protection setup to be ignored when a protected directory contains an ACL file.
To create ACL files using the Configuration and Administration Forms, click Access Control Lists.

**Setting up secure connections using an external CA**

The example in this section summarizes how to use the server’s Key Management Utility (IKEYMAN) and Configuration and Administration Forms to set up security when using an external trusted CA such as VeriSign. If you want to act as your own CA and use self-signed certificates, see the example in “Setting up secure connections using self-signed certificates” on page 63.

For detailed instructions and information on this task, go to the online help examples and select the task, **Set up secure communications to and from the server**.

To access the online help examples, use either of the following methods:

- From the default Front Page of your server, click **HOW DO I GET STARTED?**, then click **How do I?** to display the list of tasks.
- From the Configuration and Administration Forms, click the question mark in the upper right hand corner, then click **How do I?** to display the list of tasks.

**Step 1. Create a new key database and specify a key database password**

Use IKEYMAN for this task. Detailed steps are shown in “Creating a new key database” on page 412.

**Notes:**

1. If you use the default key database name of key.kdb, you can skip Step 5 in this example.
2. For OS/390, you must store the encrypted database password in a stash file. If this is not done, the server will start but the connection won’t be secure.

**Step 2. Create a public-private key pair and certificate request**

Use IKEYMAN for this task. Detailed steps are shown in “Creating a new key pair and certificate request” on page 414.

**Related Information:** After you create your server certificate request, you send that request to a CA to be signed. It usually takes two to three weeks to get a certificate from an external CA such as VeriSign. While you are waiting for the CA to process your certificate request, you can use IKEYMAN to create a self-signed certificate to enable SSL sessions between clients and the server. For an example, see “Setting up secure connections using self-signed certificates” on page 62.

**Step 3. Receive your CA-signed server certificate into the key database**

Use IKEYMAN for this task. Detailed steps are shown in “Receiving a certificate signed by a trusted CA” on page 417.

**Related Information:** If you request a server certificate from a CA who is not a trusted CA on your server, you cannot receive the CA-signed server certificate until you designate that CA as trusted. For a list of CAs designated as trusted on the server, see “Certificate authorities supported by the HTTP Server” on page 53. To designate a CA as trusted on your server, contact the CA to obtain their CA certificate. Then store the CA’s certificate using IKEYMAN. Detailed steps are shown in “Storing a CA’s certificate” on page 418.
Step 4. Change server security defaults (optional)
To perform this task, you must use either the Configuration and Administration Forms or edit the server configuration file directly. The online help example shows you how to specify the type of client authentication and encryption support using the Configuration and Administration Forms.

To change server security defaults using the Configuration and Administration Forms:
1. On the Front Page of your server, click CONFIGURATION AND ADMINISTRATION FORMS.
2. Click Security Configuration, then click SSL Security Options to access the options form.
3. After making your changes, you must stop and restart the server to update the server configuration file.

By default:
- Support for SSL connections is turned on.
- Port 443 is used for SSL connections.
- Client authentication is turned off.
- The server will use the strongest encryption level supported by both the client and the server.

For more information on security configuration settings, see the online help for the Configuration and Administration Forms or the descriptions of the Security directives in "Security - Set up secure connections for the server" on page 367. For an overview of secure communications options, see "Options for setting up secure communications" on page 52.

Step 5. Register your key database with the server
If you use only the default key database, key.kdb, you can skip this step.

To register other key databases with the server, you must use either the Configuration and Administration Forms or edit the server configuration file directly. The online help example shows you how to perform this step using the Configuration and Administration Forms. If you want to edit the configuration file, see "KeyFile - Set name for key database file" on page 367.

Setting up secure connections using self-signed certificates
This example summarizes how to use the server’s key management utilities (IKEYMAN and HTTP Server CA) to set up and administer your own certificate authority.

Notes:
1. In this example, you will create two key databases.
   - The server key database (or operational key database) is used during SSL operations.
   - The CA key database is necessary because you are acting as your own CA and signing your own server and client certificates. If you use an external CA, such as VeriSign, you do not need to build your own CA key database. For an example, see "Setting up secure connections using an external CA" on page 61.
2. In this example, key files will be stored in subdirectory /user1/ in the IKEYMAN program directory /usr/lpp/internet/bin.

CA Utility Note: You can use the HTTP Server CA utility to generate and issue signed certificates. This utility does not replace IKEYMAN. Before using the CA
utility, you must create your server and CA key databases, server certificate request, and self-signed CA certificate using IKEYMAN.

**Step 1. Create your server key database and certificate request**

Use IKEYMAN for this task.

To create your server key database and certificate request:

1. Create a new key database and specify a key database password. Detailed steps are shown in “Creating a new key database” on page 413.

   In this example, the new key database is /user1/server.kdb.

   **Note:** For OS/390, you must store the encrypted database password in a stash file. If this is not done, the server will start but the connection won’t be secure.

2. Create a public-private key pair and certificate request. Detailed steps are shown in “Creating a new key pair and certificate request” on page 414.

   In this example, the certificate request is /user1/certreq.arm.

**Step 2. Create your CA key database and certificate**

To create your CA key database and certificate:

1. Create a new key database and specify a key database password. Detailed steps are shown in “Creating a new key database” on page 413.

   In this example, the new key database is /user1/cakey.kdb.

   **IMPORTANT:**
   - To use the CA utility, your CA key database must be named cakey.kdb and copied into the following directory:
     
     /usr/lpp/internet/server_root/CAServlet/language_code/
     
     For language_code, enter C (English) or Ja_JP (Japanese).
   - You can optionally choose to store your encrypted CA key database password in a stash file. If the stash file is in the same directory as the CA key database file (cakey.kdb), then all certificates that are sent to be signed by the CA utility will be approved automatically. If you create a stash file (cakey.sth), copy that file into the same directory as the cakey.kdb file.
   - After creating your CA key database using IKEYMAN, you must export the key database into the HTTP Server CA utility. Detailed steps are shown in “Exporting the CA key database” on page 236.

2. Create a self-signed certificate. Detailed steps are shown in “Creating a self-signed certificate” on page 414.

   In this example, the certificate is /user1/cert.arm.

**Step 3. Create and sign your server certificate**

In this example, your server certificate is certserv.txt. To create and sign this certificate using IKEYMAN, issue the following command:

```
ikeyman -g -x 180 -cr /user1/certreq.arm -ct /user1/certserv.txt -k /user1/cakey.kdb
```

**Note:** If you do not create the server and CA certificates on the same day, the effective number of days starts to decrement. The default is 364 days. Use -x to specify the number of days.

**Step 4. Store your CA certificate in the server key database**

Detailed steps are shown in “Storing a CA’s certificate” on page 418.
Security configuration

In this example, the server key database is /user1/server.kdb and the CA certificate is /user1/cert.arm.

Step 5. Receive your CA-signed server certificate
Detailed steps are shown in "Receiving a certificate signed by a trusted CA" on page 417.

In this example, the CA-signed server certificate is /user1/certserv.txt. This certificate was created in "Step 3. Create and sign your server certificate" on page 63.

Step 6. Change server security defaults (optional)
To perform this task, you must use either the Configuration and Administration Forms or edit the server configuration file directly.

To change server security defaults using the Configuration and Administration Forms:
1. On the Front Page of your server, click **CONFIGURATION AND ADMINISTRATION FORMS**.
2. Click **Security Configuration**, then click **SSL Security Options** to access the options form.
3. After making your changes, you must stop and restart the server to update the server configuration file.

By default:
- Support for SSL connections is turned on.
- Port 443 is used for SSL connections.
- Client authentication is turned off.
- The server will use the strongest encryption level supported by both the client and the server.

For more information on security configuration settings, see the online help for the Configuration and Administration Forms or the descriptions of the Security directives in "Security - Set up secure connections for the server" on page 367. For an overview of secure communications options, see "Options for setting up secure communications" on page 52.

Step 7. Register the server key database with the server
To register the server.kdb key database with the server, you must use either the Configuration and Administration Forms or edit the server configuration file directly.

**Note:** You do not need to register the CA database, cakey.kdb, with the server since this database is not used during SSL operations.

To use the Configuration and Administration Forms:
1. On the Front Page of your server, click **CONFIGURATION AND ADMINISTRATION FORMS**.
2. Click **Security Configuration**, then click **Key Database Configuration**.
3. Click **Add key database** and enter the name of the database.
4. After adding the databases, you must stop and restart the server to update the server configuration file.

To edit the configuration file directly, see "KeyFile - Set name for key database file" on page 367.
Setting up SSL support for multiple IP addresses

To configure your server for SSL support of multiple IP addresses:
1. From the Front Page of your server, click **CONFIGURATION AND ADMINISTRATION FORMS**.
2. Click **Security Configuration**, then click **Key Database Configuration**.
3. Scroll down to **Multiple key entries** to enter the required information.

**Related information:**
- For a description of this option, see “SSL support for multiple IP addresses” on page 54.

**Advanced option:**
- Update the **SSLServerCert** directive in the server configuration file, httpd.conf.
  For instructions, see “**SSLServerCert - Associate a key database with an IP address**” on page 374.

Changing the server’s default encryption settings

To change the server’s default encryption settings:
1. From the Front Page of your server, click **CONFIGURATION AND ADMINISTRATION FORMS**.
2. Click **Security Configuration**, then click **SSL Security Options**.
3. Scroll down to **Specify SSL cipher specifications** to select the encryption settings for your server.

**Related information:**
- For a description of this option, see “Support for specifying the encryption level to be used” on page 53.

**Advanced option:**
- Update the **SSLCipherSpec** directive in the server configuration file, httpd.conf.
  For instructions, see “**SSLCipherSpec - Specify levels of encryption supported by the server**” on page 369.
Security configuration
Part 5. Advanced Configuration
Chapter 9. Customizing cache management with the Fast Response Cache Accelerator

Overview

The Fast Response Cache Accelerator can improve the performance of the HTTP Server when serving text and image files; dynamic content and protected pages are not cached.

Because the Cache Accelerator cache is automatically loaded during server operation, you are not required to list the files to be cached in your server configuration file. In addition, the server will automatically recache changed pages and remove outdated pages from the cache.

The Cache Accelerator provides support for caching on multiple Web servers and on servers with multiple IP addresses. Currently, support is not available for running the Cache Accelerator on a proxy server.

Planning considerations

Access to WLM

Before using the Cache Accelerator, ensure that the HTTP Server has access to OS/390 Workload Management (WLM) as outlined in "Step 7. Enable the Web server to support OS/390 Workload Management (optional)" on page 22.

Unique subsystem name for WLM

If you are not using the -SN parameter when starting the server, you must specify a unique subsystem name for the Cache Accelerator when you configure dynamic caching. For more information, see the description of the FRCAWLMParms directive in "Tuning - Define performance and scalability settings" on page 383.

TCP/IP stack name

If you are using common INET and multiple TCP/IP stacks, identify the name of the TCP/IP stack which supports the Cache Accelerator. This name must match the name on the SubFileSysType statement in the OS/390 UNIX BPXPRMxx parmlib member. For more information, see the description of the FRCAStackName directive in "Tuning - Define performance and scalability settings" on page 383.

Enabling and configuring the Cache Accelerator

By default, the Cache Accelerator is not enabled.

You can enable and configure the Cache Accelerator by using the Configuration and Administration Forms or by editing your server configuration file directly:

- To use the Configuration and Administration Forms:
1. From the Front Page of your server, click CONFIGURATION AND ADMINISTRATION FORMS.
2. Click Fast Response Cache Accelerator.
3. Click Enable Fast Response Cache Accelerator to turn dynamic caching on.
4. Configure the Cache Accelerator using the appropriate options. Refer to the online help for more information on each option.
5. Restart the server to use the new settings.

- To edit the configuration file:
  1. Open the httpd.conf configuration file.
  2. Set the EnableFRCA directive on.
  3. To configure the Cache Accelerator, use the FRCA directives. For information on FRCA directives, see “EnableFRCA — Turn dynamic caching on or off” on page 383.

Monitoring and managing the Cache Accelerator

Two resources for monitoring and managing the Cache Accelerator are Resource Measurement Facility (RMF) reports and output from the DISPLAY TCPIP command.

RMF reports

The WLM enclave created for the Cache Accelerator is a scheduled SRB. Therefore, to manage the Cache Accelerator workload, you need one service class period that has an execution velocity goal. The RMF report will show no transactions, and CPU information will be included under TCB CPU consumption.

For more information on RMF, see the RMF User’s Guide.

DISPLAY TCPIP command

To display information on the Cache Accelerator enter:

D TCPIP,FRCA_stack_name,NET,CACH

For FRCA_stack_name, specify the name of the OS/390 UNIX physical file system that supports the TCP/IP stack used by the Cache Accelerator, for example, TCPV34. This is the name that appears on the FRCAStackName directive in the server configuration file.

Sample output:

D TCPIP,TCPV34,NET,CACH
EZ25000I NETSTAT CS V2R7 TCPV34 324
CLIENT: WEBSRV3 LISTENING SOCKET: 0.0.0.0..8137
MAXCACHESIZE: 00000002500 CURRCACHESIZE: 0000000120
MAXNUMOBJECTS: 0000000200 CURRNUMOBJECTS: 0000000045
NUMCONNS: 00000002116 CONNSPROCESSED: 00000001576
CONNSDEFERRED: 0000000540 CONNSTIMEDOUT: 0000000000
REQUESTSPROCESSED: 00000002116 INCOMPLETEREQUESTS: 0000000000
NUMCACHEHITS: 00000001576 NUMCACHEMISSES: 00000000540
NUMUNPRODCACHEHITS: 00000001576
1 OF 1 RECORDS DISPLAYED

For more information on DISPLAY TCPIP output fields, see the TCP/IP OE User’s Guide.
Chapter 10. Customizing logs and reports

This chapter explains how to tailor the server’s access and error logs to meet your needs, and how to create customized reports from the information in the logs.

Tailoring the logs your server keeps

Overview of log types

The HTTP Server creates the following types of logs:
- Server Access log
- FRCA Access log, if you are using the Fast Response Cache Accelerator for dynamic caching
- Proxy and Cache Access logs, if your server is running as a proxy
- Agent log
- Referer log
- Server Error log
- CGI Error log

Server Access log

The server logs activity in the access log files and stores them each night. At midnight each night, the server closes the current access log and creates a new access log file for the following day. The access log contains entries for page request mode to the server.

For each access request your server receives, an entry is made in the access log showing:
- What was requested
- When it was requested
- Who requested it
- The method of the request
Customizing Logs and Reports

- The type of file that your server sent in response to the request
- The return code, which indicates whether the request was honored

**FRCA Access log**

The FRCA Access log is an optional log.

Set up the FRCA Access log if you want requests served by the Fast Response Cache Accelerator to be logged in a separate log file. If this log is used, no other log entries will be written for requests served from the Cache Accelerator cache. If this log is not used, requests that are served by the Cache Accelerator will be logged the same as other requests.

You may want to use this optional log if you are concerned about performance but still want access log information about requests served from the Cache Accelerator cache.

The server logs activity in the FRCA Access log file and stores it each night. At midnight each night, the server closes the current access log and creates a new access log file for the coming day.

For each access request your server receives, an entry is made in the access log showing:
- What was requested
- When it was requested
- Who requested it
- The method of the request
- The type of file that your server sent in response to the request
- The return code, which indicates whether the request was honored

**Related Information:**

- For information on the dynamic caching function, see "Chapter 9. Customizing cache management with the Fast Response Cache Accelerator" on page 69.

**Proxy server access logs**

If your server is running as a proxy, the server can create two different types of logs:
- A proxy access log, which contains access requests for files that come from the proxy server
- A cache access log, which contains access requests for files that come from the proxy server’s cache

**Agent and Referer logs**

The agent log indicates which Web browser was used to access a Web page. The referer log identifies the Web page that referred (or linked to) the requested Web page. By default the server writes an entry to the agent and referer logs each time a client sends the server a request. For every entry made in the access log:
- The agent log has a corresponding entry that indicates the browser used to display the page or file requested by the client
- The referer log has a corresponding entry that indicates the referring page

**Server Error log**

The server creates an error log that includes errors encountered by your server’s clients, such as timing out or not getting access.
Customizing Logs and Reports

CGI Error log
The server creates a CGI error log that logs standard error output (stderr) from CGI programs.

Overview of log setup options
To set up the logs to suit your particular needs, refer to the following sections:

- “Specifying global settings for all logs”
- “Specifying the path and file name for the Access, Agent, and Referer logs”
- “Specifying the path and file name for the FRCA Access log” on page 74
- “Specifying the path and file name for the proxy server’s Cache Access log” on page 74
- “Choosing log maintenance options for the Access, Agent, and Referer logs” on page 74
- “Setting filters for the Access, Agent, and Referer logs” on page 75
- “Specifying options for the error logs” on page 76

Note: You can change the default settings for the logs either by using the online Configuration and Administration forms or by manually editing the directives in the configuration file.

Specifying global settings for all logs
In most cases, you will want to accept the default global settings, which apply to all logs.

If you plan to use the reporting functions described under “Tailoring the reports your server creates” on page 78 you must accept the default file format, common.

If you want to have log information sent to the HTTP Server window in addition to sending it to the log files, you must change the default.

To change the global settings, we recommend that you specify them on the Global Log File Configuration form.

Defaults: Common file format, which is used by most Web servers, and local time format are used. Log files are written nightly.

Directives:
- For time stamp, edit the LogTime directive. For more information, see “LogTime - Specify GMT or local time stamps in log files” on page 340.
- For file format, edit the LogFormat directive. For more information, see “LogFormat - Specify common or old log file format” on page 338.

You cannot change whether the logs are written nightly.

Specifying the path and file name for the Access, Agent, and Referer logs

Note: If you are using a separate access log for the Fast Response Cache Accelerator, see the instructions in “Specifying the path and file name for the FRCA Access log” on page 74.

From the Access Log File Configuration form, you can specify the path and name of the directory where you want to place the access, agent, and referer log files.

Directives:
Customizing Logs and Reports

- For the access log path, edit the AccessLog directive. For more information, see “AccessLog - Name the path for the access log file” on page 324.
- For the agent log path, edit the AgentLog directive. For more information, see “AgentLog - Name the path for the agent log file” on page 333.
- For the referer log path, edit the RefererLog directive. For more information, see “RefererLog - Name the path for the referer log file” on page 342.

Specifying the path and file name for the FRCA Access log
From the Fast Response Cache Accelerator form, you can specify the path and file name for the FRCA Access log file.

Directive: For the FRCA access log path and name, edit the FRCAAccessLog directive. For more information, see “FRCAAccessLog — Specify a log file path and name for dynamic caching requests” on page 385.

Specifying the path and file name for the proxy server’s Cache Access log
If the server is running as a proxy, you can log requests to the cache separately from other requests.

From the Access Log File Configuration form, you can specify the path and file name where you want the server to put access requests that are satisfied from the proxy server’s cache. As an alternative, you can specify this information manually by editing the directive listed below.

Directive: For the Cache Access log path, edit the CacheAccessLog directive. For more information, see “CacheAccessLog - Specify the path for the cache access log files” on page 334.

Choosing log maintenance options for the Access, Agent, and Referer logs

Note: If you have set up a FRCA Access log for the Fast Response Cache Accelerator, your log maintenance options will apply to that access log.

With the log maintenance options, you can specify how to handle the accumulation of daily logs for days past.

You can choose whether you want to keep old logs, remove logs after they reach a certain age and/or a collective size, or run your own program at midnight each night to handle old logs. Note that the “collective size” is the collective size of all access logs only (not combined with agent and referer logs), or of all agent logs only (not combined with access and referer logs), or of all referer logs only (not combined with access and agent logs).

To reduce the space the access, agent, and referer logs require, you can specify that the logs be automatically removed, based on the age of the log and/or the collective size of the logs.

If you are interested in running your own backup program to store the logs, you can specify a user exit. In this case, you specify the path to your program and the parameters to pass to your program. The server appends to this information the path to the logs.
We recommend you define these options on the Access Log File Configuration form, but you can edit the configuration file to include the appropriate directives. The settings you specify on the Access Log File Configuration form apply to agent and referer logs, as well.

**Defaults:** By default, all access, agent, and referer log files are kept at the path location you specify on the Access Log File Configuration form (or the AccessLog, AgentLog, and RefererLog directives).

**Directives:** The directives you specify for access logs apply to agent and referer logs, as well.

- To keep access log files, no directive is required. AccessLogArchive none is the default.

- To remove access log files based on age, edit these directives:
  - AccessLogArchive purge
  - AccessLogExpire number-of-days

- To remove access log files based on collective size, edit these directives:
  - AccessLogArchive purge
  - AccessSizeLimit number-of-megabytes

- To run a user exit, edit the AccessLogArchive userexit directive.

For details on these directives, refer to "AccessLogArchive - Remove existing access, agent, or referer log files or run a user exit" on page 324.

You can use a binary program or a shell script for your user exit.

**Setting filters for the Access, Agent, and Referer logs**

**Note:** You cannot set filters for the FRCA Access log.

For the Server Access log, you can set filters so that the Access, Agent, and Referer logs include only the information you are interested in.

To improve your ability to use the information included in the Access, Agent, and Referer log files, you can filter out extraneous information so that the log includes only information that is meaningful to you. You filter out information by excluding entries that match a particular pattern. We recommend you define these options on the Access Log File Configuration form, but you can edit the configuration file to include the appropriate directives for the filters you want to set. You can specify filters based on any of the following:

- URL (directories and/or files)
- IP address or host name
- Method
- MIME type
- Return code

**Note:** Keep in mind that information filtered out from the access log will not show up in any access report and will not be available for future use.

Here are some reasons for controlling what gets logged:

**To reduce the size of the logs:** You might be interested in reducing the number of entries in an access log to include only meaningful access requests. Access log files can grow rapidly, since by default they contain entries for all access requests for GIF images, HTML pages, and so on. You might want to configure your access
Customizing Logs and Reports

logs so that they include log entries for access requests to HTML pages, but not for
the access requests for the GIF images that the HTML contains. For example, an
HTML page might include several GIF images, which can cause the size of the
access log to grow rapidly.

To collect information about external hits only: You might be interested only in
who is accessing your server from outside your company. In this case, you would
filter out access requests that originate from internal company IP addresses.

To gather information about who is accessing a particular Web site: To help you
determine the size of the audience for a particular Web site, you might want to
create an access log that shows only the hits to one URL.

Default: By default, everything is logged to the access log, unless you choose to
filter out (exclude) something. From the Access Log File Configuration form, you
can specify what you want to filter out from the access log. You do not need to fill
in the entire form.

Scroll to the Exclusions from the Access log section of the form. Choose which of
the following you want to base filtering on:
- Directories and files
- Host names or IP addresses
- Methods (GET, PUT, POST, DELETE)
- MIME types (images, text, applications, audio, video, multimedia, and other)
- Return code (success, redirection, client error, and server error)

If you want to filter based on directories and files or IP addresses and host names,
you need to update the index list on the Access Log File Configuration form. You
can insert or remove entries in the list to specify what you want filtered out. To
exclude entries based on methods, MIME types, or return codes, click the boxes
that describe what you want to filter out.

When you have finished specifying what you want to exclude on the Access Log
File Configuration form, click Submit to have the filters take effect.

Directives:
- To filter out files or directories that match a particular pattern, edit the
  AccessLogExcludeURL directive.
- To filter out entries of a particular method, edit the AccessLogExcludeMethod
directive.
- To filter out entries of a particular MIME type, edit the
  AccessLogExcludeMimeType directive.
- To filter out entries receiving a particular set of return codes, edit the
  AccessLogExcludeReturnCode directive.

Specifying options for the error logs
This section describes the following tasks:
- “Specifying the path for the Server Error and CGI Error logs”
- “Choosing log maintenance options for the Server Error and CGI Error logs” on
  page 77

Specifying the path for the Server Error and CGI Error logs: From the Error Log
File Configuration form, you can specify the path and name of the directory
where you want to place the Server Error and CGI Error log files. As an alternative, you can specify this information manually by editing the directive listed below.

**Directives:** For the Server Error log, edit the ErrorLog directive. For more information, see "ErrorLog - Name the file where you want to log internal server errors" on page 333.

For the CGI error log, edit the CgiErrorLog directive. For more information, see "CgiErrorLog - Name the path for the CGI error log file" on page 333.

**Choosing log maintenance options for the Server Error and CGI Error logs:** You can choose whether you want to keep old logs, remove logs after they reach a certain age and/or a collective size, or run your own program at midnight each night to handle old logs. Note that the “collective size” is the collective size of all error logs only (not combined with CGI Error logs) or all CGI Error logs only (not combined with Server Error logs).

To reduce the space error logs require, you can specify that the logs be automatically removed, based on the age of the log and/or the collective size of the logs.

If you are interested in running your own backup program to store the logs, you can specify a user exit. In this case, you specify the path to your program and the parameters to pass to your program. The server appends to this information the path to the logs.

You can use a binary program or a shell script for your user exit.

**Default:** By default, all error and CGI error log files are kept at the path location you specify on the Error Log File Configuration form (or the ErrorLog directive).

We recommend you define these options on the Error Log File Configuration form, but you can edit the configuration file to include the appropriate directives. The settings you specify on the Error Log File Configuration form apply to CGI error logs, as well.

**Directives:** The directives you specify for error logs apply to CGI error logs, as well.

- To keep error log files, no directive is required. ErrorLogArchive none is the default.
- To remove error log files based on age, edit these directives:
  - ErrorLogArchive purge
  - ErrorLogExpired number-of-days
- To remove error log files based on collective size, edit these directives:
  - ErrorLogArchive purge
  - ErrorSizeLimit number-of-megabytes
- To run a user exit, edit the ErrorLogArchive userexit directive.

For details on these directives, refer to "ErrorLogArchive - Remove existing error or CGI error log files or run a user exit" on page 333.
Sample scenario for configuring log files

In the following example, you have just purchased and installed the HTTP Server. You want to set up your server to log access and error information in the following ways:

- You want the access and error logs to use a local time stamp and a common log file format.
- You have enabled dynamic caching using the Fast Response Cache Accelerator and want to log those requests in a separate access log file.
- You want the Server Access log files to be purged when they are more than 30 days old and/or when they reach a collective size of 25 megabytes. You do not want the following requests to be logged to the Server Access log:
  - Requests for GIF images
  - Requests from hosts with IP addresses that match 9.67.*.*
  - Redirection requests (requests that yield a return code between 300 and 399)
- You do not want the Server Error log to be purged.

You can specify these criteria by using the Configuration and Administration forms, or by updating the configuration file directives.

Forms

- Use the Global Log File Configuration Settings form to set the time and file format
- Use the Fast Response Cache Accelerator form to specify the path and file name for the FRCA Access log.
- Use the Access Log File Configuration form to:
  - Set the interval for removing old access logs
  - Set the collective size of 25 megabytes
  - Exclude the MIME type of images/GIF
  - Exclude requests from hosts with IP addresses in the pattern 9.67.*.*
  - Exclude requests that yield a return code between 300 and 399
  - Specify the path for the Server Access log file
- Use the Error Log File Configuration form to indicate that you want to keep the error log files.

Directives

For the above scenario, update the configuration file as follows:

<table>
<thead>
<tr>
<th>Directive</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>FRCAAccessLog</td>
<td>/usr/lpp/internet/server_root/frca-log</td>
</tr>
<tr>
<td>LogFormat</td>
<td>Common</td>
</tr>
<tr>
<td>LogTime</td>
<td>LocalTime</td>
</tr>
<tr>
<td>NoLog</td>
<td>9.67.<em>.</em></td>
</tr>
<tr>
<td>AccessLogArchive</td>
<td>purge</td>
</tr>
<tr>
<td>ErrorLogArchive</td>
<td>none</td>
</tr>
<tr>
<td>AccessLogExpire</td>
<td>30</td>
</tr>
<tr>
<td>AccessLogSizeLimit</td>
<td>25</td>
</tr>
<tr>
<td>AccessLogExcludeURL</td>
<td>*.gif</td>
</tr>
<tr>
<td>AccessLogExcludeReturnCode</td>
<td>300</td>
</tr>
</tbody>
</table>

Tailoring the reports your server creates

Reporting options and considerations

IBM provides two reporting programs with the HTTP Server: HTLOGREP and Web Usage Mining. For information on these programs, see “Using the HTLOGREP or Web usage mining reporting programs” on page 79.
IBM also provides the option to use third-party reporting programs. See the following section for more information and instructions.

If you are using proxy server support, see "Proxy server considerations".

**Configuring the Web server to use a third-party reporting program**

You can specify a third-party reporting program using the Configuration and Administration Forms or by editing your server configuration file directly:

- To use the Configuration and Administration Forms:
  1. From the Front Page of your server, click **CONFIGURATION AND ADMINISTRATION FORMS**.
  2. Click **Logging and Reporting**.
  3. Click **Access Report File Configuration**. Refer to the online help for more information on configuration options.
  4. Restart the server to use the new settings.

- To edit the configuration file:
  1. Open the httpd.conf configuration file.
  2. Set the DoReporting directive on (default).
  3. Use the LoggingReportingProgram directive to specify the path and name of the third-party reporting program; use the LoggingReportingOptions directive to specify reporting program options.

For information on these directives, see:

- "DoReporting — Specify if reports are automatically generated" on page 335
- "LoggingReportingProgram — Specify the reporting program to be used" on page 339
- "LoggingReportingProgramOptions — Specify reporting program options" on page 339

**Proxy server considerations**

If you have specified the CacheAccessLog directive or if you have indicated on the **Access Log File Configuration** form a path and file name for the proxy server’s cache access log, your reports will not contain access requests for cached files. If you do not have a cache access log, access requests for a proxy server are logged in the access log and can be included in an access report.

**Using the HTLOGREP or Web usage mining reporting programs**

The following template-based reports are provided with your Web server:

- For HTLOGREP:
  - Host reports
  - Method reports
  - Code reports
  - URL reports
- For Web usage mining:
  - Daily mining statistics reports
  - Weekly mining statistics reports

You control what is included in reports by filtering out entries that match a particular pattern included in a report template. These options are defined using the Configuration and Administration Forms or by editing the configuration file. You can use the forms or the configuration file to specify filters based on any of the following:


Customizing Logs and Reports

- URL (directories and files)
- IP address or host name
- Method
- Return code

The contents of the report are governed by the following factors:
- The log file filters that were in effect when the log file was created
- The report filters that are in effect when the report is created

At report creation time, you control only the report filters that are currently in effect. You cannot include in the report entries that were filtered out from the log file.

You can specify report filters in two ways; you must decide which is easier in your situation.
- If you want to include in your report only a small percentage of the contents of the access log, it might be simplest to specify what to include, rather than everything you want to exclude. Think of specifying one or a few include filters as a shortcut to specifying many exclude filters.
- If you want to include in your report most of the contents of the access log, it might be simplest to specify what to exclude, rather than everything you want to include.

In some cases, you will find it simplest to specify both include and exclude filters. In this case, it is important to understand how include and exclude filters work together. The include filters are processed first. The report function searches the access log to find all entries that match any include filter patterns. If several include filters are specified, the filters act as OR Boolean expressions. In other words, entries that match at least one of the include filters are included.

The exclude filters are processed after all include filters have been processed. The exclude filters work only on the set of entries that have been already included by the include filters. For clarification, refer to the examples under “Sample scenarios for configuring reports” on page 82.

The include and exclude filters are specified on the Access Log Report Template Creation form or can be specified with the AccessReport directives.

Here are some reasons for controlling what gets reported:

To reduce the scope of the report: You might be interested in reducing the scope of the report so that it includes only a portion of what is contained in the log. You can even create several reports, each to gather different information from the same log. You might want to create your report template so that it includes log entries for access requests to HTML pages, but not for the access requests for the GIF images that the HTML contains.

To collect information about external hits only: You might be interested only in who is accessing your server from outside your company. In this case, you would filter out access requests that originate from internal company IP addresses.

To gather information about who is accessing a particular Web site: To help you determine the size of the audience for a particular Web site, you might want to create a report that shows only the hits to one URL.
To discover the top Web pages on your server: To help you determine the popularity of a particular Web site, you filter out everything in the report, except for the most visited Web pages.

Creating a report template

Before you create a report, you must modify or create a report template that outlines what you want the report to contain. To start configuring a report template, choose one of the following options from the Access Log Report Templates form:

- If you want to create a report that is very different from your existing reports, choose Create a new template.
- If you have never created a report, it might be easier to copy an existing template and edit the copy, rather than creating a new template. To use an existing report template as the basis of a new report, choose Copy existing template.
- To update an existing report template, choose Edit existing template.
- To delete an existing report template, choose Delete existing template.

When you choose Create a new template or Edit existing template, the Access Log Report Template Creation form appears.

On this form, you can specify some or all of the following:

- Basic settings, such as the report name, description, and the number of entries to include in the report (for example, the top 10)
- Entries from the access log that you want to exclude from the report (based on directories and file names, host names or IP addresses, methods, or return codes)
- Entries from the access log that you want to include (as a shortcut to many excludes)

Specifying entries to include is a shortcut to specifying many, many excludes. When you want to include only a few types of entries in the report, it is easier to specify what to include rather than excluding nearly everything. For example, if you want to include only access requests for a particular URL, you would include that URL, rather than excluding all the others.

The Access Log Report Template Creation form allows you to specify includes and excludes. It is important to understand how includes and excludes affect each other.

- In the index list, if nothing is listed with a Filter Action of “Include”, the entire log is included in the report, minus the entries that are excluded.
- If anything is listed with a Filter Action of “Include”, the report will contain only the information that is to be included, minus the entries that are excluded.

When you have finished filling in the form, click Submit.

The Web usage mining statistics are based on the report templates defined. To generate the report, you must either run webusage or wait until the current day’s log files are closed. This normally occurs at midnight when the current day’s logs are closed and the next day’s log files are started.

To initialize the report template files, you must either run htlogrep or wait until the current day’s log files are closed.
Customizing Logs and Reports

If you are using the Access Log Report Templates form, you see at the bottom of the form the field Report root directory. This field is filled in with a default directory. We recommend that you accept the default, rather than changing it. If you choose to change the default, you will need to create a new directory for the path you specify, give the directory the appropriate permissions and add a PASS statement to enable the server to honor requests to store reports in that directory.

Viewing reports

Reports are created dynamically through the use of a Java applet accessed from your browser. Configuration directives are provided for the compression of log data, archiving of reports, and inclusion of old log data in reports. Refer to the "Appendix C. Configuration directives" on page 267 for details of these directives.

To see a report, from the Configuration and Administration Form page, choose Access reports. From there, select the following options:

- Report template
- One of the following filters:
  - URL (directories and files)
  - IP address or host name
  - Method
  - Return code
- Date range

By loading a template and then clicking on any report, you can display the report or print it from your Web browser. The report is created and displayed after you select the options.

By default, your server creates a report template that shows:

- The top 50 most frequently visited URLs on your server
- The top 50 most frequent visitors to your Web site

In addition, for each report template, the server creates a daily and weekly report of Web usage mining statistics. For a description of the Web usage mining statistics, see "Using the Web usage mining statistics reports" on page 83.

Sample scenarios for configuring reports

You have just purchased and installed the HTTP Server and you want to set up your server to automatically generate four different access log reports.

Sample report: Top 100 page hits

You are interested in knowing which Web pages on your server get the most attention. You decide to create a report that meets the following criteria:

- The name of the report is “Top100”.
- The report description says “Top 100 page hits”.
- Requests for GIF images are not included.

You can specify these criteria by using the Configuration and Administration forms, or by updating specific directives in the configuration file.

Forms:

1. From the Access Log Report Templates form, choose Create a new template. Select before in the List it field.
2. Change nothing in the Root report directory field.
3. Click Submit.
5. For Report description, type Top 100 page hits.
6. For Report on top, type 100.

    Note: When you are using Web usage mining, the value for the AccessReportTopList directive must be an integer value up to 1000.

7. Scroll down the form.
8. Choose Add and Exclude Directories/Files listed below.
9. In the text box, type *.GIF.
10. Scroll down to the end of the form and click Submit.

Directives:

```
AccessReportTemplate Top100 {
    AccessReportDescription Top 100 page hits
    AccessReportTopList 100
    AccessReportExcludeURL *.GIF
}
```

Sample report: PUT requests to beta subdirectory

You are running a site that distributes beta-level software and are interested in knowing what is being written to the beta directory and who is requesting PUT access. You decide to create a report that meets the following criteria:

- The name of the report should be “BetaPuts”.
- The report description should say “PUT requests to beta subdirectory”.
- The report should include only requests for PUT access to the beta subdirectory, which is located at /www/beta.

You can specify these criteria by using the Configuration and Administration forms, or by updating specific directives in the configuration file.

Forms:

1. From the Access Log Report Templates form, choose Create a new template. Select before in the List it field.
2. Change nothing in the Root report directory field.
3. Click Submit.
5. For Report description, type PUT requests to beta subdirectory.
6. Scroll down the form.
7. Choose Add and Include Directories/Files listed below.
8. In the text box, type /www/beta/*.
9. Scroll down the form.
10. Under Exclude following Methods... choose GET, POST, and DELETE.
11. Scroll down to the end of the form and click Submit.

Directives:

```
AccessReportTemplate BetaPuts {
    AccessReportDescription PUT requests to beta subdirectory
    AccessReportIncludeURL /www/beta/*
}
```
Customizing Logs and Reports

```
AccessReportExcludeMethod  GET
AccessReportExcludeMethod  POST
AccessReportExcludeMethod  DELETE
}
```

**Sample report: Accesses, excluding beta subdirectory and alpha7 requests**

You are interested in knowing which files on your server are being accessed. However, you want to exclude beta programs, which have files located in the beta subdirectory. You also do not want to include any information on the “Alpha7” project, which has pages named Alpha7*.* in various subdirectories. You decide to create a report that meets the following criteria:

- The name of the report should be “NoBetaAlpha7”.
- The report description should say “Accesses, except beta subdirectory and alpha7”.
- The report should include all accesses, except those to the beta subdirectory at /www/beta or those files with the name alpha7 anywhere on the server.

You can specify these criteria by using the Configuration and Administration forms, or by updating specific directives in the configuration file.

**Forms:**

1. From the Access Log Report Templates form, choose Create a new template. Select before in the List it field.
2. Change nothing in the Root report directory field.
3. Click Submit.
5. For Report description, type Accesses, except beta subdirectory and alpha7.
6. Scroll down the form.
7. Choose Add and Exclude Directories/Files listed below.
8. In the text box, type:
   ```
   /www/beta/*
   alpha7*.*
   ```
9. Scroll down the form.
10. Scroll down to the end of the form and click Submit.

**Directives:**

```
AccessReportTemplate NoBetaAlpha7 {
    AccessReportDescription  Accesses, excluding beta and alpha7 requests
    AccessReportExcludeURL   /www/beta/*
    AccessReportExcludeURL   alpha7*.*
}
```

**Sample report: Accesses for department server and for the beta subdirectory except Alpha7*.* files**

Your server is a department server and you want to know the access requests for that server. You also want to know access requests for the beta subdirectory, but you are not interested in knowing access requests for any Alpha7*.* files. You decide to create a report that meets the following criteria:

- The name of the report should be “DeptServer_Beta-NotAlpha7”
- The report description should say “Accesses for Department Server”
- The report should include all accesses to the IP address that represents the department server.
Customizing Logs and Reports

- The report should include all access requests to /www/beta/*.
- The report should exclude all access requests to Alpha7*.* files.

You can specify these criteria by using the Configuration and Administration forms, or by updating specific directives in the configuration file.

Forms:
1. From the Access Log Report Templates form, choose Create a new template. Select before in the List it field.
2. Change nothing in the Root report directory field.
3. Click Submit.
5. For Report description, type Accesses for Department Server.
6. Scroll down the form.
7. Choose Add and Include Directories/Files listed below.
8. In the text box, type: 
   /www/beta/*

To exclude the Alpha7*.* files, you will have to save the settings on this form, then edit the template. You cannot specify both include and exclude on the same form.

9. Choose Add and Include host names listed below.
10. In the text box, type 9.67.*.*
11. Scroll down to the end of the form and click Submit.
12. To exclude the Alpha7*.* files, choose Edit existing template.
13. Click Submit.
14. Scroll down the form.
15. Choose Add and Exclude Directories/Files listed below.
16. In the text box, type: 
   Alpha7*.*
17. Click Submit.

Directives:

Using the Web usage mining statistics reports

If you want to understand how your users navigate through your Web site, you can look at the Web usage mining statistics. They tell you the sequence of Web pages a user clicked through during a visit.

These reports can tell you where people enter and exit from your Web site and which Web pages as a group are visited most. You can see the browsing patterns
and identify user behavior, which in turn allows you to better organize your Web pages. The reports are generated automatically and are not tailorable except through the standard report templates.

There are three types of Web usage mining statistics reports:
- User-based
- Path-based
- Group-based

**User-based** statistics help you to understand how users move through your Web site. Each user session is recorded as the sequence of HTML links followed by a specific user. If a user remains idle for some pre-specified period of time, the next sequence of links is considered to be a new user session. The user-based statistic reports shows:
  - Most frequently accessed pages organized by user count
  - Most frequent IP addresses from which users come to visit your site organized by user count
  - Distribution of user sessions both in duration and in number of pages accessed (bar charts are also supported when viewed with a Java-enabled browser)
  - Most frequent external link (referer) to your site
  - Most frequent page from which a user exits your site

**Path-based** statistics identify paths used to travel through your Web pages. Each user path is a sequence of HTML pages chosen to by a user and can reveal the user’s actual browsing behavior. Path-based statistics tell you how the HTTP links embedded in a Web presentation are actually followed by users.

**Group-based** statistics give you the groups of pages most frequently visited during a user session, helping you to see which groups of pages are most popular. A user session can contain multiple paths; and the group of pages frequently visited in a session may not lie on the same path. By examining the path-based and group-based statistics, you can obtain valuable information to improve the organization and linkage of the Web presentation.

Daily reports are provided for all three types of statistics. In addition, a week-to-date user-based report is also provided. To view these reports, from the Configuration and Administration Form page, choose **Access Reports**. A list of the existing reports by template is displayed. One link exists for each template-based report. The reports reside on the reports directory as `template_name_wumindex.MonYear.html`. For example, `Top50_wumindex.Sep1996.html` contains the hyperlinks to all the daily and week-to-date reports for the month of September 1996. Here, Top50 is the name of a report template specified by the Webmaster for basic reporting.

For each report template, there is a corresponding set of user-, path- and group-based reports. The number of items reported in each statistic is specified in each report template. For example, on the Access Report Template Creation page at the **Report on Top** prompt, you can specify the number of top report items (25, 100, or all) you want to view.

**Note:** When you are using Web usage mining, the value for the **AccessReportTopList** directive must be an integer value up to 1000.
Notes:

1. You must have set the AccessLog, AgentLog, and RefererLog directives to create the access, agent, and referer logs so the server can create the Web usage mining statistics reports. You also need to set the AccessReportTemplate directive.

2. The reports will be in .html format and will be stored in the path specified by the AccessReportRoot directive in the httpd.conf file.

3. Web usage mining reports can be viewed by selecting Access reports from the Configuration and Administration Form page and selecting the appropriate link.

Running Web usage mining from the command line

Before you can run the Web usage mining command, your sessions must be able to access the Web server’s environment settings. This means that your user ID .profile settings must match the settings in the server’s httpd.envvars file. To change environment variable settings in your .profile, use the export command, for example:

```bash
export NLSPATH=environment_variable_setting
```

Note: If you run the Web usage mining command and see the error message, Error using message catalog, this usually indicates that the NLSPATH setting in your .profile does not match the setting in the server’s httpd.envvars file.

To run the Web usage mining command, type:

```bash
webusage -c full_path_of_configuration_file [-f httpd_ip_filename]
```

where

- `full_path_of_configuration_file` is the full path of the active configuration file.
- `httpd_ip_filename` is the filename of a file containing the IP address of the machine that generated the log files (for example, 9.1.2.3) and the fully qualified hostname of that machine (for example, myhost.XYACorp.com).

The required -c option provides the full path of the active configuration file.

Use the -f option when you want to run the Web usage mining tool on log files that were generated on another system. When using the -f option:

- The agent, httpd, and referer log files from the other system must be copied to the LOGS directory of the system where the Web usage mining tool will run.
- The IP address must appear in the first line of the IP file and the host name must appear on the second line. These two lines must be the only two lines in the IP file.

Web usage mining creates the appropriate reports in .html format and stores them in the root report directory.

Logs generated on another system may have different names than the logs generated on the system where the Web usage mining tool is running. In this case, you must change the appropriate configuration file directives for the Web usage mining tool to find the logs.
Web usage mining and multiple servers

Multiple servers can run on a single machine. Each server runs with a unique configuration file and listens on a unique port. For more information about running your server with multiple IP addresses or virtual hosts, see “Chapter 16: Running your server with multiple IP addresses or virtual hosts” on page 165.

The Web usage mining tool generates reports specific to each server. Each invocation of the Web usage mining tool has unique -c parameters and generates unique reports.

Logging information with System Management Facilities

With the System Management Facilities (SMF), you can request that configuration and performance data be recorded to SMF datasets. With this recorded configuration and performance data, you can monitor Web server health, throughput, and activity.

Configuration record data is taken from the server configuration file, httpd.conf, and is written after the server daemon is fully initialized. Performance record data is accumulated continuously and written at intervals defined in the httpd.conf file by the SMFRecordingInterval directive.

When multiple servers are managed by a workload manager, the totals written to SMF are for all servers under the workload manager.

You can choose to have either configuration record data or performance record data, both configuration and performance record data, or no data written to SMF datasets. You can also define how often SMF writes the continuously accumulated performance information to SMF datasets.

To select the type of information to be written to the SMF dataset, you can use the Global Log File Configuration Settings form in the Configuration and Administration forms or the SMF directive. The default setting for the SMF directive is all, which records both configuration and performance record data. For more information about the SMF directive, see “SMF - Specify the type of information that SMF records” on page 346.

To specify how frequently performance record information is written to SMF datasets, use the Global Log File Configuration Settings form in the Configuration and Administration forms or the SMFRecordingInterval directive. The default SMFRecordingInterval is 00:15, which means performance record information is recorded every 15 minutes if the logging queue is full. If activity on the server is low, the logging queue fills slowly. In this case, the recording interval for performance record information may be longer than the specified interval. For more information about the SMFRecordingInterval directive, see “SMFRecordingInterval - Specify how often performance record information is recorded” on page 346.

For more information about the Configuration and Administration Forms see “Using the Configuration and Administration forms” on page 42.
Chapter 11. Customizing your Web site

Displaying page count, date, time, and text on a Web page ................................................................. 89
Configuration instructions ......................................................................................................................... 89
Samples ....................................................................................................................................................... 90
Options ....................................................................................................................................................... 90
Common options ......................................................................................................................................... 91
Counter option ........................................................................................................................................... 92
Date and time option ............................................................................................................................... 92
Text to gif option ...................................................................................................................................... 92
Server-side includes and support for text-based browsers ...................................................................... 93
Using server-side includes to insert information into CGI programs and HTML documents .............. 93

This chapter describes methods for customizing the appearance of your Web site. It includes information about displaying page count, date, time, and text on a Web page and using server-side includes to insert information into CGI programs and HTML documents.

Displaying page count, date, time, and text on a Web page

This section explains how to use the htcounter program to display the following information on a Web page:

Page count
The page counter is incremented each time the Web page is accessed, and the current value is displayed on the Web page.

Date and time
The current date and time are displayed on the Web page.

Text User-specified text is displayed on the Web page.

Configuration instructions

To use the HTCounter program:

1. Uncomment Service directives in the Web server configuration file for the functions you are using.

   In the following example, install_path is the root directory of your Web server installation. The default install path is /usr/lpp/internet.

   • Page count:
     # Service /cgi-bin/apicounter*
     install_path/bin/htcounter.so:HTCounter*
   
   • Date and time:
     # Service /cgi-bin/datetime*
     install_path/bin/htcounter.so:HTCounter*
   
   • Text:
     # Service /cgi-bin/text2gif*
     install_path/bin/htcounter.so:HTCounter*

2. Create the counter file and initialize the counter.
Customizing your Web site

The Web administrator must create the counter file and initialize the counter to some value, for example, 0. This enables the Web administrator to control access to the Web site and to Web site resources used to display the requested information.

The counter file must be located in the server_root/Counters directory and the server needs to have write access to the counter file. The default setting for server_root is /usr/lpp/internet/server_root.

For information on sample HTML files, see "Samples".

3. Insert lines in the Web page HTML file for the functions you are using.
   • Page count:
     
     `<img src="/cgi-bin/apicounter/counter_name?[common_options] 
     [counter_option]">`
   
   • Date and time:
     
     `<img src="/cgi-bin/datetime?[common_options] 
     [date_and_time_option]">`
   
   • Text:
     
     `<img src="/cgi-bin/text2gif?[common_options] 
     [text_to_gif_option]">`

   **Note:** Place the information in the above examples on one line, even though it is shown here on two lines.

   For *counter_name*, enter the name of the counter file you created, for example, *cntfile.cnt*.

Samples

Sample HTML files are located in the server_root /pub directory, or you can go to the following URLs:

   • To view the sample counter page, go to URL: http://your.server.name/counter1.html
     
     This page also includes an explanation of error messages that may be issued by the HTCounter program when you begin using the counter file.
   
   • To view the sample color page, go to URL: http://your.server.name/counter2.html

For *your.server.name*, enter the fully qualified name of your host, for example *http://myserver.raleigh.ibm.com*.

Options

This section describes the options you can use for displaying the page count, date, time, and text on your Web page.

**Notes:**

1. Defaults are shown in **bold** letters.
2. Option names and values are **not** case-sensitive.
3. Use an ampersand (&) to separate options.
   
   For example, to display a page counter with a foreground color of blue and a background color of white, use the following URL:
   
   `<img src="/cgi-bin/apicounter/counter_name?FG=blue&BG=white">`
   
4. The **RRGGBB** color option allows you to specify the color using a hexadecimal color code, where **RR**, **GG**, and **BB** are the hexadecimal digits that specify the Red, Green, and Blue values of the color. Examples of color values are:
For example, to display a page counter with a foreground color of yellow, you would use the following URL:

```
<img src="/cgi-bin/apicounter/counter_name?FG=FFFF00">
```

For colors demonstrated online, go to URL
http://your.server.name/counter2.html or Lem Apperson’s Color Index at http://www.infi.net/wwwimages/colorindex.html.

**Common options**

**FG=color**
where *color* specifies the foreground color, and can be:

- **Black**  
- **White**  
- **Red**  
- **Green**  
- **Blue**  
- **RRGGBB**

**BG=color**
where *color* specifies the background color, and can be:

- **Transparent**  
- **Black**  
- **White**  
- **Red**  
- **Green**  
- **Blue**  
- **RRGGBB**

**BorderColor=color**
where *color* specifies the border color, and can be:

- **Green**  
- **Black**  
- **White**  
- **Red**  
- **Blue**  
- **RRGGBB**

**BorderWidth=width**
where *width* specifies the width of the border around the image, and can be:

- **0** No border (the default)  
- **n** The number *n* determines the thickness of the border.

**BorderIndent=highlighting**
where *highlighting* specifies highlighting for upper and right border edges (3D beveled effect), and can be:

- **In** Upper and right border edges are shaded.  
- **Out** Lower and left border edges are lighter.
Customizing your Web site

**BorderIndentColor**=`color`
where `color` specifies the color for the border edge (3D beveled effect), and can be:

- **BorderColor**
  - Black
  - White
  - Red
  - Green
  - Blue
  - RRGGBB

**BorderIndentWidth**=`width`
where `width` specifies the width of the border edge (3D beveled effect), and can be:

- `0`   No border edge (the default)
- `n`   The number `n` determines the thickness of the border edge.

**FontName**=`font`
where `font` specifies the font used, and can be:

- **Block1**
- **LCD**

**FontSize**=`size`
where `size` specifies the font size (width x height), and can be:

- `8x12`
- `7x11`
- `9x13`
- `10x14`

**Counter option**

**Format**=`format`
where `format` specifies the format for displaying the `counter_value`, and can be:

- `%%d`   No padding
- `%%nd`   Pad with blanks; width=`n`
- `%%0nd`   Pad with zeros; width=`n`

**Date and time option**

**Format**=`strftime()-format`
Specifies the format for displaying the `date` and `time`:

- **Default**: http_time format
- `strftime()-format`
  - Use `%20` to represent a blank.
  - For all other options, see Table 4 on page 96.

**Timebase**=`time`
Specifies the time used:

- **Local**
- **GMT** (Greenwich mean time)

**Text to gif option**

**Text**=`string`
Specifies the text string that will be converted to a gif. Use `%20` to represent a blank.

**Example:**
Using server-side includes to insert information into CGI programs and HTML documents

Server-side includes allow you to insert information into CGI programs and HTML documents that the server sends to the client. This section describes the command format for using server-side includes and explains how to use the commands needed to make server-side includes work in your CGI programs and HTML documents.

Considerations for using server-side includes

Before using server-side includes on your server, note the following considerations:

- Performance
Performance can be significantly impacted when the server is processing files while sending them.

- **Security**
  Letting ordinary users execute commands can be a security risk. Be very careful when deciding which directories you use server-side includes in and which directories you use the exec command in. You can minimize the security risk if you do not enable the exec command.

- **File references**
  You cannot reference files recursively. For example, if you are running file sleepy.html and the program finds `<!--!#include file="sleepy.html" -->` the server doesn’t detect the error and the server loops until the server abends. However, you can reference files within files. For example, file sleepy.html references file smiley.html and file smiley.html references dopey.html.

- **Dynamic caching support**
  Files containing server-side includes cannot be dynamically cached by the Fast Response Cache Accelerator. For more information on dynamic caching, see “Chapter 9. Customizing cache management with the Fast Response Cache Accelerator” on page 69.

### Preparing to use server-side includes

To use server-side includes, you need to add the AddType directive to your configuration file. Two examples follow:

**Examples:**

```plaintext
AddType .shtml text/x-ssi-html ebcdic 1.0
AddType .htmls text/x-ssi-html ebcdic 1.0
```

**Note:** If you use file extensions other than .shtml or .htmls, you should check the AddType directive to see if that extension already exists. See the configuration file, appendix listing, or the MIME form for a list of existing AddType directives.

You can also use the imbeds directive to specify whether server-side includes can be used in HTML documents, CGI programs, or both. Examples of this directive follow:

**Example:**

```plaintext
imbeds on
```

**Default**

```plaintext
imbeds off
```

For more information about the imbeds directive, see “imbeds - Specify whether server-side includes will be dynamically imbedded” on page 290.

The server does not process your error files for imbeds, regardless of the file extensions or use of the imbeds directive.

### Format for server-side includes

The current date, the size of a file, the last change of a file are examples of the kind of information that can be sent to the client. There are commands that need to be included in the HTML document comments. The commands have the following format:
Syntax
The following is the syntax format for enabling server-side includes on the server:

```
<!--#directive tag=value ... -->
<!--#directive tag="value" ... -->
```

The quotes around value are optional. They are required when there are imbedded spaces.

Directives for server-side includes
This section explains the directives that are accepted by the server for server-side includes.

**config - controls file processing**
Use this directive to control certain aspects of file processing. Valid tags are cmntmsg, errmsg, sizefmt, and timefmt.

**cmntmsg - specify the message appended to the beginning of text**: Use this tag to specify the message that gets appended to the beginning of any text that follows a directive specification and comes before "-->".

Example:
```
<!--#config cmntmsg="[This a comment]" -->
<!--#echo var=" " extra text -->
```

Result: (Output from the echo) <!--This is a comment extra text -->

Default: [the following was extra in the directive]

**errmsg - specify the message sent to the client**: Use this directive to specify the message that gets sent to the client if an error occurs when a file is being processed. The message gets logged in the server's error log.

Example:
```
<!--#config errmsg="[An error occurred]" -->
```

Default: "[an error occurred while processing this directive]"

**sizefmt - specify file size format**: Use this directive to specify the format to be used when the file size is displayed. In the following examples, bytes is the value used for a formatted number of bytes. abbrev is used for displaying the number of kilobytes or megabytes.

Example 1:
```
<!--#config sizefmt = bytes -->
<!--#fsize file=foo.html -->
```

Result: 1024

Example 2:
```
<!--#config sizefmt=abbrev -->
<!--#fsize file=foo.html -->
```

Result: 1K

Default: "abbrev"
**Customizing your Web site**

**timefmt - specify date format:** Use this directive to specify the format to be used when providing dates.

**Example:**

```html
<!--#config timefmt="%T %D" -->
<!--#flastmod file=foo.html -->
```

**Result:** "12:05:33 10/18/95"

**Default:** "%a, %d %b-%Y %T %Z"

The following strftime() formats are valid with the timefmt tag:

<table>
<thead>
<tr>
<th>Specifier</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>%</td>
<td>Replace with %</td>
</tr>
<tr>
<td>%a</td>
<td>Replace with the abbreviated weekday name.</td>
</tr>
<tr>
<td>%A</td>
<td>Replace with the full weekday name.</td>
</tr>
<tr>
<td>%b</td>
<td>Replace with the abbreviated month name.</td>
</tr>
<tr>
<td>%B</td>
<td>Replace with the full month name.</td>
</tr>
<tr>
<td>%c</td>
<td>Replace with the date and time.</td>
</tr>
<tr>
<td>%C</td>
<td>Replace with the century number (year divided by 100 and truncated).</td>
</tr>
<tr>
<td>%d</td>
<td>Replace with the day of the month (01-31).</td>
</tr>
<tr>
<td>%D</td>
<td>Insert the date as %m/%d/%y.</td>
</tr>
<tr>
<td>%e</td>
<td>Insert the month of the year as a decimal number (01-12). Under C POSIX only, it is a 2-character, right-justified, blank-filled field.</td>
</tr>
<tr>
<td>%E[cCxyY]</td>
<td>If the alternative date/time format is not available, the %E descriptors are mapped to their unextended counterparts. For example, %EC is mapped to %C.</td>
</tr>
<tr>
<td>%Ec</td>
<td>Replace with the alternative date and time representation.</td>
</tr>
<tr>
<td>%EC</td>
<td>Replace with the name of the base year (period) in the alternative representation.</td>
</tr>
<tr>
<td>%Ex</td>
<td>Replace with the alternative date representation.</td>
</tr>
<tr>
<td>%EX</td>
<td>Replace with the alternative time representation.</td>
</tr>
<tr>
<td>%Ey</td>
<td>Replace with the offset from %EC (year only) in the alternative representation.</td>
</tr>
<tr>
<td>%EY</td>
<td>Replace with the full alternative year representation.</td>
</tr>
<tr>
<td>%h</td>
<td>Replace with the abbreviated month name. This is the same as %b.</td>
</tr>
<tr>
<td>%H</td>
<td>Replace with hour (23-hour clock) as a decimal number (00-23).</td>
</tr>
<tr>
<td>%I</td>
<td>Replace with hour (12-hour clock) as a decimal number (00-12).</td>
</tr>
<tr>
<td>%j</td>
<td>Replace with the day of the year (001-366).</td>
</tr>
<tr>
<td>%m</td>
<td>Replace with the month (01-12)</td>
</tr>
<tr>
<td>%M</td>
<td>Replace with minute (00-59).</td>
</tr>
<tr>
<td>%n</td>
<td>Replace with a new line.</td>
</tr>
<tr>
<td>%O[deHImMSUwWy]</td>
<td>If the alternative date/time format is not available, the %E descriptors are mapped to their unextended counterparts. For example, %Od is mapped to %d.</td>
</tr>
<tr>
<td>%Od</td>
<td>Replace with the day of the month, using the alternative numeric symbols, filled as needed with leading zeroes if there is any alternative symbol for zero, otherwise with leading spaces.</td>
</tr>
</tbody>
</table>
### Table 4. Conversion Specifiers Used by strftime() (continued)

<table>
<thead>
<tr>
<th>Specifier</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>%Oe</td>
<td>Replace with the day of the month, using the alternative numeric symbols, filled as needed with leading spaces.</td>
</tr>
<tr>
<td>%OH</td>
<td>Replace with the hour (24 hour clock) using the alternative numeric symbols.</td>
</tr>
<tr>
<td>%OI</td>
<td>Replace with the (12 hour clock) using the alternative numeric symbols.</td>
</tr>
<tr>
<td>%Om</td>
<td>Replace with the month using the alternative numeric symbols.</td>
</tr>
<tr>
<td>%OM</td>
<td>Replace with the minutes using the alternative numeric symbols.</td>
</tr>
<tr>
<td>%OS</td>
<td>Replace with the seconds using the alternative numeric symbols.</td>
</tr>
<tr>
<td>%OU</td>
<td>Replace with the week number of the year (Sunday as the first day of the week, rules corresponding to %U) using the alternative numeric symbols.</td>
</tr>
<tr>
<td>%OW</td>
<td>Replace with the weekday (Sunday=0) using the alternative numeric symbols.</td>
</tr>
<tr>
<td>%OV</td>
<td>Replace with the week number of the year (Monday as the first day of the week) using the alternative numeric symbols.</td>
</tr>
<tr>
<td>%Oy</td>
<td>Replace with the year (offset from %C) in the alternative representation and using the alternative numeric symbols.</td>
</tr>
<tr>
<td>%p</td>
<td>Replace with the local equivalent of AM or PM.</td>
</tr>
<tr>
<td>%r</td>
<td>Replace with the string equivalent to %I:%M:%S %p</td>
</tr>
<tr>
<td>%R</td>
<td>Replace with time in 24 hour notation (%H:%M)</td>
</tr>
<tr>
<td>%S</td>
<td>Replace with seconds (00-61).</td>
</tr>
<tr>
<td>%t</td>
<td>Replace with a tab.</td>
</tr>
<tr>
<td>%T</td>
<td>Replace with a string equivalent to %H:%M:%S.</td>
</tr>
<tr>
<td>%u</td>
<td>Replace with the weekday as a decimal number (1 to 7), with 1 representing Monday.</td>
</tr>
<tr>
<td>%U</td>
<td>Replace with the week number of the year (00-53) where Sunday is the first day of the week.</td>
</tr>
<tr>
<td>%V</td>
<td>Replace with the week number of the year (01-53) where Monday is the first day of the week.</td>
</tr>
<tr>
<td>%W</td>
<td>Replace with the week number of the year (00-53) where Monday is the first day of the week.</td>
</tr>
<tr>
<td>%x</td>
<td>Replace with the appropriate date representation.</td>
</tr>
<tr>
<td>%X</td>
<td>Replace with the appropriate time representation.</td>
</tr>
<tr>
<td>%y</td>
<td>Replace with the year with the century.</td>
</tr>
<tr>
<td>%Y</td>
<td>Replace with the year with the current century.</td>
</tr>
<tr>
<td>%Z</td>
<td>Replace with the name of the time zone or no characters if the time zone is unknown.</td>
</tr>
</tbody>
</table>

The operating system configuration determines the full and abbreviated month names and years.

**echo - specify environment variables**

Use this directive to display the value for specified environment variables using the `var` tag. If a variable is not found, a (None) is displayed. The following environment variables can be displayed:
Customizing your Web site

DATE_GMT
The current date and time in Greenwich Mean Time. The formatting of this variable is defined using the config timefmt directive.

DATE_LOCAL
The current date and local time. The formatting of this variable is defined using the config timefmt directive.

DOCUMENT_NAME
This is the name of the topmost document. If the HTML was generated by a CGI, this will contain the name of the CGI.

DOCUMENT_URI
The full URL the client entered, without the query string.

LAST_MODIFIED
The date and time that the current file was last modified. If the current file is a script, then this variable is set to the current date and time. The formatting of this variable is defined using the config timefmt directive.

QUERY_STRING_UNESCAPED
The search query sent by the client. This is undefined unless HTML was generated by a CGI.

SSI_DIR
The path of the current file, relative to SSI_ROOT. If the current file is in SSI_ROOT, this value is “/”.

SSI_FILE
The file name of the current file.

SSI_INCLUDE
The value used in the include command that retrieved this file. This is not defined for the topmost file.

SSI_PARENT
The path and file name of the includer, relative to SSI_ROOT.

SSI_ROOT
The path of the topmost file. All include requests must be in this directory or a child of this directory.

Example:
<!--#echo var=SSI_DIR -->

Also, echo can display a value set by the set or global directives.

exec - specify CGI programs
Use this directive to include the output of a CGI program. Exec discards any HTTP headers CGI outputs EXCEPT for:

content-type
determines whether to parse the body of the output for other Includes.

content-encoding
determines if translation needs to be done (ebcdic/ascii).

last-modified
replaces the current last modified header value if it is later.
cgi - specify CGI program URL
Use this directive to specify the URL of a virtual path to a CGI program.

Example 1:
<!--#exec cgi="/cgi-bin/program/path_info?query_string" -->

In the example, program is the cgi program to be executed, path_info are the parameters passed to the program as environment variables, and query_string are the parameters passed to the program as environment variables.

Example 2:
<!--#exec cgi="/path;cgiprog;pathinfo;querystring;" -->

Example 2 shows the use of variables.

flastmod - display time and date document was changed
Use this directive to display the last time and date the document was changed. The formatting of this variable is defined using the config timefmt directive. The file and virtual tags are valid with this directive, and the meaning is the same as it is for the include directive.

Directive Formats:
<!--#flastmod file="/path/file" -->
<!--#flastmod virtual="/path/file" -->

Example:
<!--#flastmod file="F00" extra text -->

Result: 12May96 <!-- This is extra text -->

fsize - display file size
Use this directive to display the size of the specified file. The formatting of this variable is defined using the config sizefmt directive. The file and virtual tags are valid with this directive, and the meaning is the same as it is for the include directive.

Examples:
<!--#fsize file="/path/file" -->
<!--#fsize virtual="/path/file" -->

Result: 1K

global - defines global variables
Use this directive to define global variables that can be echoed later by this file, or any included files.

Example:
<!--#global var=VariableName value="Some Value" -->

If you want to reference a parent document across the "virtual" boundary, you need to set a global variable DOCUMENT_URI. You also need to reference the global variable in the child document. The following is an example of the HTML coding you need to insert in the parent document:

Example:
<!--#global var="PARENT_URI" value=DOCUMENT_URI; -->
The following is an example of the HTML coding you need to insert in the child document:

Example:
<![--#lastmod virtual=&PARENT_URI; --]>

**include - includes a document in output**
Use this directive to include a document (the text from a document) in the output. The file and virtual tags are valid with this directive:

**file - specify file name**: Use this tag to specify the name of a file.

For **lastmod**, **fsize**, and **include**, **file** is assumed to be relative to SSI_ROOT if preceded by a ‘/’. Otherwise, it is relative to SSI_DIR. The file specified must exist either in SSI_ROOT or in one of its descendents.

Example:
<![--#include file="/path/file" -->

**virtual - specify a document URL**: Use this tag to specify the URL of a virtual path to a document.

For **lastmod**, **fsize**, and **include**, **virtual** is always passed through the server’s mapping directives.

Example:
<![--#include virtual="/path/file" -->

**set - sets variables to be echoed**
Use this directive to set a variable that can be echoed later by only this file.

Example:
<![--#set var="Variable 2" value="AnotherValue" -->

**Variables**: Server-side includes also allow you to echo a variable already set. While defining a directive, you can echo a string in the middle of “value”. For example:
<![--#include file="&filename;" -->

If an unrecognized variable is found, nothing gets displayed.

Server-side includes look for the variable, echo where the variable is found, and proceed with the function. You can have multiple variable references. When server-side includes encounter a variable reference INSIDE a server-side include directive, they attempt to resolve it on the SERVER side. The following example escapes the & so that server-side includes does not recognize it as a variable. In the second line, the variable "&index;" is a server-side variable and is used to construct the variable name "var1". The variable &ecirc; is a client side variable, so the & is escaped to create the value ":fr\&ecirc;d" or "fred" with a circumflex over the e.
<![--#set var="index" value="1" -->
<![--#set var="var&index;" value="fr\&ecirc;d" -->
<![--#echo var="var1" -->

The following characters that can be escaped. Note that escaped variables are preceded with a backslash (\).
Using server-side image maps to enable clickable images

Use the htimage service to process clickable image maps. It allows defined regions within an image map to be associated with specific URLs. When users click on a defined region, the htimage service redirects the server to the URL associated with the region. When the server returns the URL associated with the region, it issues a 302 return code (Moved Temporarily) and a Location header containing the URL.

Note: The URL specified in the Location header must be fully qualified, or absolute. The URL should include the protocol, host, and request path. For example:

http://hostnamefilename.html

The HTTP Server assumes, if the URL in the map file is not absolute, that the URL is local, and serves it directly. While this saves some network overhead, the browser thinks the original request was satisfied and continues to use the original request as the base for future requests.

When the URL is not fully qualified and the server finds the file locally, the HTTP Server issues a Content-Location header which some browsers use to make future requests.

Many browsers understand the HTML tag, `<BASE HREF= >`, which can be used to specify the base URL (for example, `http://hostname`). When this tag is used, URLs that are not fully qualified are evaluated relative to this base URL and are correctly found.

The service is used in conjunction with a map file. Map files are text files that define regions within a graphics file by their x,y coordinates and map them to the various URLs.

Note: Currently the universal image file accepted by all browsers and servers is the GIF format. GIF is an 8-bit 256 color image file.

You cannot use htimage from the command line. You include it as part of an anchor tag within an HTML document, and it is called when the server processes that document.

The htimage service is enabled by specifying the following in the server configuration file:

```
service /cgi-bin/htimage* INTERNAL:HTImage*
service /cgi-bin/imagemap* INTERNAL:HTImage*
```
Syntax

Since htimage can only be called from an anchor tag within an HTML document, the syntax is shown as HTML markup.

```html
<a href="/cgi-bin/htimage/mapfile.txt">
<img src="/icons/image.gif" ISMAP></a>
```

```html
<a href="/cgi-bin/htimage/mapfile.txt">
The href attribute of the anchor (a) tag specifies the URL of the htimage command followed by the URL of the map file.

This syntax description assumes that the sample configuration file is being used. The sample configuration file contains Service and Exec directive that maps requests beginning with /cgi-bin/ to the directory that contains the htimage service.

The server uses everything following /htimage/ as the URL of the map file. If the server is using the sample configuration and the URL contains only a file name, the server would look for the file in the document root directory.

```html
<img src="/icons/image.gif" ISMAP></A>
```

The src attribute of the image (img) tag specifies the URL of the file that contains the graphic you want to use as an image map.

The ISMAP attribute indicates that the graphic is an image map.

Examples

Following is a description of the map files that must be used with the htimage service. A map file is an EBCDIC text file.

**Note:** Each line of the file is in the following format:

```
region-identifier [region-definition] URL
```

**region-identifier**

A keyword that identifies the type of region being defined. Valid values are:

- rectangle
- circle
- polygon
- default

**region-definition**

A set of numbers that defines a particular region of the graphic. The format of the region definition is different for each type of region.

Coordinates within parentheses identify a point relative to the top left corner of the image. The first number is the number of pixels to the right of the top left corner. The second number is the number of pixels down from the top left corner. There are several shareware programs available that can help you easily identify the coordinates of particular points within a graphic file.

- The default keyword does not define a region. The keyword is followed by a URL to link to when the client clicks on a portion of the image map that is not covered by any of the other region definitions.
For rectangle, the first point is the upper left corner of the rectangle. The second point is the lower right corner. In other words, define any two diagonally opposite corners having coordinates \((x1,y1)\) and \((x2,y2)\).

For circle, the point is the center of the circle. The single number following the point is the radius of the circle as measured in pixels.

For polygon, up to 100 points can be defined. The shape is formed by connecting the points in the order they are given. The last point is connected to the first.

For example, you might use the following HTML in a document:

```html
<a href="/cgi-bin/htimage/mapit.txt">
<img src="/icons/mapimage.gif" ISMAP></a>
```

The above example calls the htimage command with a map file named mapit.txt. The mapit.txt file would define regions of the mapimage.gif graphic file. Because no path is specified for mapit.txt, the server would look for it on the document root directory. Following is an example of what mapit.txt might look like.

```
default http://brimstone/cgi-bin/go_home
rectangle (50, 100) (200,200) http://brimstone/cgi-bin/go_to_it
circle (100,300) 50 http://brimstone/pub/example.html
poly (450,350) (450,500) (150,500) http://brimstone/pub/triangle.html
```

**Note:** There is an HTML tag, BASE, which can be used to specify the base URL (for example, http://hostname) so that relative URLs are evaluated relative to this base URL and are correctly found.

One example of a shareware program for determining the \(x,y\) coordinates is mapedit. You can obtain mapedit from the following URL:

**http://www.boutell.com/mapedit/**

Please note that output from mapedit is in NCSA format. It is different from the CERN examples shown.

The same examples in NCSA format require anchor tags within an HTML document as follows:

```html
<a href="/cgi-bin/imagemap/mapit.txt">
<img src="/icons/mapimage.gif" ISMAP></a>
```

The map file will be an EBCDIC text file with the following format:

```
default http://brimstone/cgi-bin/go_home
rect http://brimstone/cgi-bin/go_to_it 50, 100 200,200
circle http://brimstone/pub/example.html 100,300 100,350
poly http://brimstone/pub/triangle.html 450,350 450,500 150,500
```

**URL**

The fully qualified or absolute URL, including the protocol, hostname and filename, is required. If the URL in the map file is not absolute, the HTTP Server assumes that the URL is local and serves it directly.
Customizing your Web site
Chapter 12. Managing your Web server

Modes of Operation for the Web server

There are three execution modes for the Web server: Standalone Server mode, Scalable Server mode, and running multiple servers. This section lists the characteristics of these execution modes.

It is important to understand these execution modes. These server execution modes are discussed here so you will have an overview of the characteristics of these modes and where workload management, simple network management protocol and system management facilities work.

Standalone Server mode

The following is a list of the characteristics of a standalone server:

- Single IMWHTTPD Web daemon job/process
- Console interface
- http: and https: accept loops
- Restart (SIGHUP) and shutdown (SIGTERM) handlers
- Proxy cache manager
- All requests are handled
- PID and log files
- SMF records, SNMP subagent
Scalable Server mode

When running in Scalable Server mode, the Web server works in conjunction with OS/390 Workload Management (WLM). This setup can improve server performance by dividing work between various work queues.

The following is a list of the characteristics of a scalable server subsystem:

Single IMWHTTPD Queue Manager job/process
- Console interface
- http and https: sockets
- Restart (SIGHUP) and shutdown (SIGTERM) handlers
- PID and log files
- Proxy cache manager
- SMF records, SNMP subagent
- Creates a unique subsystem instance (-SN parameter)
- Handles unrouted (non-ApplEnv) requests
- Creates shared memory configuration/parameters

(Zero to n) IMWHTTPD Queue Server jobs/processes
- Joins a unique subsystem instance (-SN parameter)
- Uses shared memory configuration/parameters
- No console interface
- No accept loops
- No restart handler
- PID and log file (per IMWHTTPD)
- No proxy cache manager
- No SMF records, and no SNMP subagent. However, counters are accumulated and can be accessed through the Web server Queue Manager.
- Handles requests routed to one ApplEnv (-AE parameter) and subsequent unrouted requests on persistent connections
- Auto-started (by WLM based on policies) or pre-started (after Queue Manager for -SN)

Running Multiple Servers

Running multiple servers allows you to have multiple instances of the Web server running simultaneously on different ports. To do this, you must have separate configuration files and ports.

Some resources cannot be shared across server instances:
- Port number
  - If your server is running without BindSpecific, port numbers MUST be unique.
  - If your server is running with BindSpecific, port numbers must be unique within the given IP address.
- PID file
  - The absolute PidFile must be unique. If the PidFile is a relative name, then ServerRoot and PidFile are combined to create an absolute name.
- Log files
  - Absolute LogFile names must be unique. If specified as relative names, then ServerRoot and LogFile are considered.
Managing the Web server

- Proxy cache manager (garbage collection)
  Servers can share a proxy cache, but only one server should do proxy cache management for a given proxy cache.

- Scalable server subsystem instance name
  Each Queue Manager must have a unique SubSystem Name (-SN). Each Queue Server must identify which SubSystem Instance it is joining (-SN).

- SNMP MIB
  If you are running multiple servers, you can either enable SNMP on only one server or install one SNMP agent for each SNMP-enabled server.

---

Workload Management Enablement for the Web server

Workload Management (WLM) is a component of the OS/390 operating system. WLM provides the functions to define, implement, and monitor system performance against business goals. WLM uses policies you define to match resources to workloads.

The Web server allows you to use workload management to establish policies to manage workload. The scalable server subsystem works in conjunction with workload management. OS/390 systems typically run a wide variety of applications with conflicting resource requirements and priorities. Workload management modifies the environment to balance workload and improve system performance.

Workload management overview

Workload management (WLM) allows you to establish policies for managing the server environment. Based on these policies, workload management modifies the environment to achieve the best overall results which match individual needs. You can divide incoming requests into Application Environments. Each application environment is treated as a workload management queue of transactions. HTTP requests are routed to workload management transaction queues based on the ApplEnv directives specified in your server configuration file.

Workload management may change scheduling priority of individual threads. WLM may also change the number of processes working on an application environment.

The description that follows tells you how to define application environments using ISPF panels. For more information about WLM panels, see OS/390 MVS Planning: Workload Management.

Configuring workload management to support the Web server

You use the WLM administrative application to set up the application environment and classification rules. The workload management application panels are automatically installed with your OS/390 system. To start workload management, from the TSO command line, enter:

```
ex 'sys1.sblscli0(iwmarn0)'
```

The following screen is the “Choose Service Definition” panel. This panel allows you to:
- Read an existing policy (option 1)
- Read the policy already installed on your coupled data set, if it applies (option 2)
Managing the Web server

• Create a new policy (option 3)

A sample policy has been included with the Web server. The data set name is called 'IMW.SIMWTBL1'.

To use the sample policy, select option 1. If you want to create a new policy, select option 3.

Note: The following information describes how to modify or use the sample that is provided with the Web server.

After selecting option 1, the following panel is displayed. Enter the data set name that contains WLM policy information in the appropriate field and press enter to continue. This causes WLM to read the sample and display the WLM configuration panels.
The following screen is the “Definition Menu” panel that allows you to set up WLM to your individual needs. From this screen you can select option 6 or 9 to change various WLM parameters to meet your needs.

By specifying option 6 on the Definition Menu, you can view a list of subsystem types defined to WLM. IWEB is reserved to WLM for the Web server.

The following screen is the Subsystem Type Selection List for Rules. One or more applications, in addition to the server, can be displayed. To modify the server (IWEB) policy, select option 3 and press enter to continue.
Use the following screen, "Modify Rules for the Subsystem Type" panel, to define classification rules. These classes are defined under option 4 of the definition panel.

When you are done with this screen, select PF12 to return to the previous screen.

After reading in the sample, you may want to add, delete, or modify some of the Application Environments based on your individual needs. To do this, specify 9 to display a list of the currently defined Application Environments.
The following screen is the “Application Environment Selection List”. The sample includes three application environments. Each environment is configured to manage a specific function. WEBCGI is a sample application environment to process CGI command requests; WEBCICS is a sample application environment for CICS processing; and WEBHTML is a sample application environment for HTML processing. To create new application environments, select option 1. To modify an existing application environment, select option 3.

You must define at least one application environment in order for WLM to start server address spaces. You may define more than one application environment if there are functions that have widely varying resource requirements, and the requests for those functions cannot be divided into separate service classes. Workload management maintains separate address spaces for each combination of application environment and service class that appears in the workload.
Managing the Web server

The following screen is the "Modify an Application Environment" panel and shows you how to create application environments. The variable, &IWMSSNM, is resolved to the subsystem name by the server PROC.

After modifying or creating new application environments, press PF12 to return to the main WLM panel. To complete the workload management configuration process, perform the following steps:

1. Select file, then save.
2. Install the WLM definition. Go to the first panel, put the cursor under UTILITIES, and choose INSTALL DEFINITION. This installs your changes to the coupled data set.
3. Activate the service policy. Go to the first panel, put the cursor under UTILITIES, and choose ACTIVE SERVICE POLICY.
4. Exit the WLM application.
5. Turn WLM on. Enter F WLM,MODE=GOAL on the operator console.
6. Start the server using the -SN parameter. The parameter should be added to the start-up PROC.

Information about configuring RACF to support WLM, starting the queue server with WLM, and common WLM problems can be found in "Chapter 3. Installing your secure server” on page 17.

How workload management panels are interpreted

You can start the queue manager by specifying a subsystem name as WLMUT1 (-SN WLMUT1). The substitution occurs as follows:

1. When WLM attempts to start the queue server, WLM sees &IWMSSNM in the start parameters and substitutes WLMUT1 for it before issuing the OS/390 start command. The command looks like this:

   S IMWIWM,IMWSN=WLMUT1,IMWAE=WEBHTML

2. The JCL for IMWIWM goes through the converter and interpreter (C/I) before you get control in the new address space. C/I processing sees all the &xxxxx symbols and tries to resolve them to values. &SN, &AE, and &QQ are easy to resolve because they are SET in the PROC. &IWMSN and &IWMAE must be resolved from the start command. Specifying IWMSN=&IWMSSNM, IWMAE=WEBHTML on the PROC
statement tells OS/390 that those symbols are inputs from the start command. OS/390 goes to the start command, gets the values, WLMUT1 and WEBHTML respectively, and substitutes them into the JCL.

3. After C/I is done and IMWEBSRV gets control, ICSPARM will have all the correct values substituted.
   ICSPARM=' -SN WLMUT1 -AE WEBHTML'

For more detailed explanations of how to use the WLM application panels, see OS/390 MVS Planning: Workload Management.

### Configuring your environment for workload management

To configure your environment for workload management, follow the directions in "Chapter 3. Installing your secure server" on page 17. This step instructs you on the following:

- Putting the IMWEBSRV and IMWIWM procedures in a site procedure library, such as SYS1.PROCLIB. IMWEBSRV is a procedure for the standalone server or the WLM Queue Manager. IMWIWM is the procedure that can be started automatically by WLM for the Queue Server.
- Defining the IMWIWM procedure to the started task table.
- Assigning the same user ID to IMWIWM as you do to the server.

**Notes:**

1. A RACF profile must be defined for the subsystem IWEB.
2. Only -SN and -AE can be passed on ICSPARM’s in the IMWIWM procedure. Any other parameters cause the procedure to fail.

### Configuring your application environment name for workload management

To configure your application environment name for workload management, follow the directions in "Appendix C. Configuration directives" on page 267. This section instructs you on how to set up the following workload management directives:

- ApplEnv
- ApplEnvConfig
- PluginDefault
- PluginExclude
- PluginInclude

You can use the Workload Management Configuration and Administration Forms to set up your application environment name.

### Server Activity Monitor

Use the Server Activity Monitor to monitor Web server performance and status. The Server Activity Monitor allows you to view Web server activity statistics, network statistics, and access log entries. You can access and display this information remotely without being on the same machine that is running the Web server. This option provides significantly more information than opening the console window.
Enabling the Server Activity Monitor

By default, the Server Activity Monitor is enabled by the following Service directive in the Web server configuration file:

```
Service /Usage* INTERNAL:UsageFn
```

Accessing statistics and usage information

To access Server Activity Monitor statistics from the Web server Configuration and Administration Forms:

1. Click **Configuration and Administration Forms** on the default Front page of the Web server.
2. In the **System Management** section, click **Server Activity Monitor**.

To update the statistics on a page, click **Refresh**.

To access usage reports directly, go to the following URLs:
- To monitor server activity: 
  http://your.server.name/Usage/Initial
- To monitor the status of proxy requests: 
  http://your.server.name/Usage/proxylog
- To monitor the network load: 
  http://your.server.name/Usage/Netstat
- To show a dynamic view of the access log: 
  http://your.server.name/Usage/Logs

Web server activity statistics

The following table shows the statistics included on the Activity Statistics page. Details follow the table.

<table>
<thead>
<tr>
<th>Thread counts</th>
<th>Request statistics</th>
<th>Throughput statistics</th>
<th>Connection counts</th>
<th>Response Times (in seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Threads running: 39</td>
<td>Requests processed: 80</td>
<td>Response time for local files: 1 second</td>
<td>Active inbound connections: 1</td>
<td>Service</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Plugins</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>--Maximum: 0.057782</td>
<td>--Minimum: 0.012920</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>--Average: 0.022376</td>
<td></td>
</tr>
<tr>
<td>Threads idle: 38</td>
<td>Request errors: 5</td>
<td>Response time for proxied requests: &lt;1 second</td>
<td>Active outbound connections: 0</td>
<td>CGI</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>--Maximum: 20.838786</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>--Minimum: 4.970356</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>--Average: 14.001646</td>
</tr>
<tr>
<td>Maximum allowed threads: 39</td>
<td>Requests discarded: 3</td>
<td>Bytes received: 83K</td>
<td>Connections since last SMF: 7</td>
<td>DNS lookup</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>--Maximum: 0.319905</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>--Minimum: 0.000004</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>--Average: 0.002186</td>
</tr>
<tr>
<td>Non-SSL Waiting Threads: 16</td>
<td>Requests proxied today: 9</td>
<td>Bytes sent: 790K</td>
<td></td>
<td>SSL</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Handshake</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>--Maximum: 53.886848</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>--Minimum: 0.000065</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>--Average: 0.002186</td>
</tr>
</tbody>
</table>
Thread counts

<table>
<thead>
<tr>
<th>SSL Waiting Threads: 16</th>
<th>Request statistics</th>
<th>Throughput statistics</th>
<th>Connection counts</th>
<th>Response Times (in seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSL</td>
<td>Proxy</td>
<td>Unknown</td>
<td>Proxy</td>
<td></td>
</tr>
<tr>
<td>Waiting</td>
<td>cache hit</td>
<td>Bytes</td>
<td>Response</td>
<td></td>
</tr>
<tr>
<td></td>
<td>rate: 0%</td>
<td>Received: 0 K</td>
<td>--Maximum: 0.821369</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>--Minimum: 0.209022</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>--Average: 0.421406</td>
<td></td>
</tr>
</tbody>
</table>

Async I/O Waiting Threads: 0

- Responses processed: 85

Msg Queue Waiting Threads: 0

- Responses discarded: 0

Thread counts

**Threads running**

Total number of threads available to do work. Threads running and Maximum allowed threads are usually the same number. However, if the Web server shuts down any threads, Threads running equals Maximum allowed threads minus the number of threads shut down since starting the Web server.

**Threads idle**

Number of threads not currently used

**Maximum allowed threads**

Maximum number of threads as specified in the Web server configuration file on the MaxActiveThreads directive; this value does not change unless the Web server is restarted with a new configuration file value for MaxActiveThreads.

**Non-SSL Waiting Threads**

Number of non-Secure Sockets Layer (SSL) threads available for use; if this value is 0, all non-SSL threads are allocated.

**SSL Waiting Threads**

Number of Secure Sockets Layer (SSL) threads available for use; if this value is 0, all SSL threads are allocated.

**Async I/O Waiting Threads**

If the Web server is running in Scalable Server mode, number of asynchronous I/O threads available for use; if this value is 0, all asynchronous I/O threads are allocated.

**Msg Queue Waiting Threads**

If the Web server is running in Scalable Server mode, number of message queue threads available for use; if this value is 0, all message queue threads are allocated.

**Request statistics**

**Requests processed**

Number of file requests that the Web server has successfully served;
Managing the Web server

Requests processed + Request errors = Responses processed. For example, if a request is for the Web server home page and 4 GIF files, the number of requests processed will be 5.

**Request errors**
Number of file requests that the Web server responded to with an error;
Request errors + Requests processed = Responses processed.

**Requests discarded**
Number of file requests sent to the Web server that are not valid

**Requests proxied today**
If configured as a proxy server, the number of file requests this server forwards to another server

**Proxy cache hit rate**
If configured as a proxy server, the percentage of proxy file requests cached

**Responses processed**
Total number of successful file responses sent; Responses processed = Requests processed + Request errors.

**Responses discarded**
Number of file responses the Web server was not able to send back to the client

**Throughput statistics**

**Response time for local files**
Average time to process a request for a file on the Web server

**Response time for proxied requests**
Average time to process a file that is forwarded to a proxy server

**Bytes received**
Total number of bytes of data sent to the Web server in requests

**Bytes sent**
Total number of bytes of data sent by the Web server in responses

**Unknown bytes received**
Total number of bytes of data that are not part of a request

**Connection counts**

**Active inbound connections**
Total number of connections from clients to the Web server; if the Web server is configured as a proxy server, the total number of connections from clients or other servers to this proxy server.

**Active outbound connections**
Total number of connections from the Web server to clients; if the Web server is configured as a proxy server, the total number of connections from this proxy server to clients or other servers.

**Connections since last SMF**
Total number of connections this Web server has provided since the last SMF record write; cumulative since the last SMF record was written
Response times (in seconds)

Service Plugins
Maximum, minimum, and average time it takes to complete customized application functions

CGI
Maximum, minimum, and average time it takes to complete Common Gateway Interface (CGI) programs

DNS lookup
Maximum, minimum, and average time it takes to complete the search for a domain name in the Domain Name Server (DNS)

SSL Handshake
Maximum, minimum, and average time it takes to complete the exchange of security information (IDs, passwords, certificates) between the Web server and browser

Proxy Response
If configured as a proxy server, the maximum, minimum, and average time it takes to complete a transaction between a browser, this proxy server, and another server

Network activity statistics
Figure 1 shows an example of the network statistics that are displayed on the Network Status page.

![Network Status page statistics](image)

Outgoing data: 1K bytes/second
Incoming data: 1K bytes/second

Refresh now

**Note:** Incoming and outgoing data values include only data received and sent by the server.

Access log entries
The access log page displays the 20 most recent entries in the access log. For more information on the access log, go to "Tailoring the logs your server keeps" on page 71.

Simple Network Management Protocol
A network management system is an application that runs continuously and is used to monitor, reflect status of, and control a network. Simple Network Management Protocol (SNMP) is the network management standard. It communicates management information with devices in a network. The network devices typically have an SNMP agent and one or more subagents. The SNMP agent talks to the network management station or responds to command line
SNMP requests. The SNMP subagent retrieves and updates data and gives that data to the SNMP agent to communicate back to the requester.

The HTTP Server provides an SNMP management information base (MIB) and SNMP subagent so you can use any SNMP-capable network management system, such as TME 10 NetView, TME 10 Distributed Monitoring, or HP OpenView, to monitor your server’s health, throughput, and activity. The MIB data describes the Web server being managed, reflects current and recent server status, and provides server statistics.

The network management system retrieves MIB values from other devices. It then can notify you if specified threshold values are exceeded. You can now proactively tune or fix server problems before they become server outages.

SNMP commands and protocol

Every device that is managed or that manages must have an SNMP agent. The user, management system, or programmer sends a GET command to the SNMP agent. In turn, this SNMP agent sends a GET command to retrieve the specified MIB variable values from a subagent responsible for those MIB variables.

The HTTP Server SNMP support includes an SNMP subagent that uses Distributed Protocol Interface (DPI) capability. DPI is an interface between an SNMP agent and its subagents.

The HTTP Server provides a subagent that updates and retrieves MIB data. The DPI subagent responds with the appropriate MIB data when the SNMP agent sends a GET command. The SNMP agent communicates the data to the network management station. The network management station can notify you if specified threshold values are exceeded.

Object IDs and variable names for the HTTP Server MIB

The HTTP Server MIB is modeled after the IETF WWW MIB RPC. MIB layout includes Variable Name, Object ID, Type, and Description.

The following Variable Names and Object IDs are provided for SNMP support with the HTTP Server:

**EntityDescription**

Description

Identifies a server in character string form. This read-only value is not customizable.
Managing the Web server

Object ID
1.3.6.1.4.1.2.6.120.1.1.1.1.1.1.1

Type          OCTET_STRING

Default value
An appropriate value for your server installation and platform.

EntityObjectID

Description
Identifies a particular server in machine-readable form, providing a globally unique name among other applications and versions. This read-only value is not customizable.

Object ID
1.3.6.1.4.1.2.6.120.1.1.1.1.2.1

Type          OCTET_STRING

Default value
1.3.6.1.4.1.2.6.120.10.1

EntityContact

Description
Indicates who to contact if a problem or question about this running server arises. It is a character string and frequently contains the e-mail address of the on-site system administrator responsible for server maintenance. The value for EntityContact may be customized with the WebMasterEmail directive in the httpd.conf file.

Object ID
1.3.6.1.4.1.2.6.120.1.1.1.3.1

Type          OCTET_STRING

Default value
webmaster

EntityProtocol

Description
Identifies the exact protocol and its version that a particular server supports. For a Web server, the protocol is HTTP. This read-only identifier is in machine-readable form and is not customizable.

Object ID
1.3.6.1.4.1.2.6.120.1.1.1.4.1

Type          OCTET_STRING

Default value
1.3.6.1.4.1.2.12.1

EntityProtocolVersion

Description
This character string identifies the protocol this server supports and the protocol version. Similar to EntityProtocol. This read-only value is not customizable.

Object ID
1.3.6.1.4.1.2.6.120.1.1.1.5.1
### Managing the Web server

<table>
<thead>
<tr>
<th>Type</th>
<th>OCTET_STRING</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default value</td>
<td>HTTP/1.1</td>
</tr>
</tbody>
</table>

**EntityName**

**Description**  
This character string provides the name of the host this Web server runs on. The value for EntityName may be customized with the HostName directive in the httpd.conf file. It is read-only and set by system-specific code at server initialization time.

<table>
<thead>
<tr>
<th>Object ID</th>
<th>1.3.6.1.4.1.2.6.120.1.1.1.1.1.6.1</th>
</tr>
</thead>
</table>

**EntityAddress**

**Description**  
This character string provides the IP address of the host this Web server runs on. It is read-only and set by system-specific code at server initialization time.

<table>
<thead>
<tr>
<th>Object ID</th>
<th>1.3.6.1.4.1.2.6.120.1.1.1.1.1.7.1</th>
</tr>
</thead>
</table>

**EntityPort**

**Description**  
This character string provides the port number this Web server listens to. It is read-only and set by system-specific code at server initialization time.

<table>
<thead>
<tr>
<th>Object ID</th>
<th>1.3.6.1.4.1.2.6.120.1.1.1.1.1.8.1</th>
</tr>
</thead>
</table>

**EntityType**

**Description**  
This integer differentiates between several server roles. Possible values are:

- **1** Simple or normal HTTP server
- **2** Proxy server
- **3** Caching server
- **4** Caching proxy

It is read-only and set by system-specific code at server initialization time. The information is taken from the httpd configuration file.
Object ID
1.3.6.1.4.1.2.6.120.1.1.1.1.1.9.1

CurrentThreads

Description
Indicates how many threads the server has currently. The total number of active threads is the sum of the MIB values, applInboundAssociations and applOutboundAssociations. This information is read-only.

Object ID
1.3.6.1.4.1.2.6.120.1.1.1.1.1.10.1

MaxThreads

Description
Indicates the maximum number of threads the server can have in the thread pool. This is read-only information.

Object ID
1.3.6.1.4.1.2.6.120.1.1.1.1.1.11.1

MinThreads

Description
Obsolete. Indicates the minimum number of threads the server can have. This is read-only information.

Object ID
1.3.6.1.4.1.2.6.120.1.1.1.1.1.12.1

SummaryTable

Description
Indicates the number of requests in each category issued by this server. These are cumulative values, counted over the life of the server process.

All server address spaces under workload management update one counter, which is accessible by the HTTP Server Queue Manager.

Table object ID
1.3.6.1.4.1.2.6.120.1.1.2.1.1

Type Table
Managing the Web server

SummaryRequests

Description
Indicates the total number of requests the server received plus the total number of requests the server generated (for example, as a proxy server). This read-only information is updated as the server runs.

Object ID
1.3.6.1.4.1.2.6.120.1.1.2.1.1.1.1

Type
COUNTER32

Default value
0

SummaryRequestErrors

Description
Indicates the total number of request errors detected by the server. This read-only information is updated as the server runs.

Object ID
1.3.6.1.4.1.2.6.120.1.1.2.1.1.2.1

Type
COUNTER32

Default value
0

SummaryRequestDiscards

Description
Indicates the total number of requests discarded by the server (for any reason). This read-only information is updated as the server runs.

Object ID
1.3.6.1.4.1.2.6.120.1.1.2.1.1.3.1

Type
COUNTER32

Default value
0

SummaryResponses

Description
Indicates the total number of responses generated or received by this server. This read-only information is updated as the server runs.

Object ID
1.3.6.1.4.1.2.6.120.1.1.2.1.1.4.1

Type
COUNTER32

Default value
0

SummaryResponseDiscards

Description
Indicates the total number of responses discarded by the server. This read-only information is updated as the server runs.
Managing the Web server

Object ID
1.3.6.1.4.1.2.6.120.1.1.2.1.1.5.1
Type  COUNTER32
Default value
0

SummaryInUnknowns
Description
Indicates the total number of unknown messages received by this server. This read-only information is updated as the server runs.

Object ID
1.3.6.1.4.1.2.6.120.1.1.2.1.1.6.1
Type  COUNTER32
Default value
0

SummaryInBytes
Description
Indicates the total number of bytes received by this server. This read-only information is updated as the server runs.

Object ID
1.3.6.1.4.1.2.6.120.1.1.2.1.1.7.1
Type  COUNTER32
Default value
0

SummaryOutBytes
Description
Indicates the total number of bytes sent out by this server. This read-only information is updated as the server runs.

Object ID
1.3.6.1.4.1.2.6.120.1.1.2.1.1.8.1
Type  COUNTER32
Default value
0

ResponseSummaryTable
Description
Indicates the number of responses in each category issued by this server. These are cumulative values, counted over the life of the server process. These values are not affected by any changes to the configuration of the HTTP Server.

All server address spaces under workload management update one counter, which is accessible by the HTTP Server Queue Manager.

Table object ID
1.3.6.1.4.1.2.6.120.1.1.2.2.1.2
Type  Table

ResponseError200Level
Managing the Web server

**ResponseError200Level**

*Description*
Indicates the number of Error 200 responses (Positive Completion responses) issued by this server. This is a cumulative value, counted over the life of the Webserver process.

*Object ID*
1.3.6.1.4.1.2.6.120.1.1.2.2.1.2.1

*Type*
Counter32

*Default value*
0

**ResponseError300Level**

*Description*
Indicates the number of Error 300 responses (Positive Intermediate Completion responses) issued by this server.

*Object ID*
1.3.6.1.4.1.2.6.120.1.1.2.2.1.2.2

*Type*
Counter32

*Default value*
0

**ResponseError400Level**

*Description*
Indicates the number of Error 400 responses (Transient Negative Completion responses) issued by this server.

*Object ID*
1.3.6.1.4.1.2.6.120.1.1.2.2.1.2.3

*Type*
Counter32

*Default value*
0

**ResponseError500Level**

*Description*
Indicates the number of Error 500 responses (Permanent Negative Completion responses) issued by this server.

*Object ID*
1.3.6.1.4.1.2.6.120.1.1.2.2.1.2.4

*Type*
Counter32

*Default value*
0

**CacheTable**

*Description*
Indicates information about the cache of this server. These values are affected by the cache settings in the configuration file.

All Webserver address spaces under workload management update one counter, which is accessible by the HTTP Server Queue Manager.

*Table object ID*
1.3.6.1.4.1.2.6.120.1.1.2.3.1
<table>
<thead>
<tr>
<th>Object ID</th>
<th>Type</th>
<th>Default value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.3.6.1.4.1.2.6.120.1.1.2.3.1.1.1</td>
<td>Counter32</td>
<td>0</td>
<td>Indicates the number of kilobytes read from the cache of this server. This is a cumulative value, counted over the life of the server process.</td>
</tr>
<tr>
<td>1.3.6.1.4.1.2.6.120.1.1.2.3.1.2.1</td>
<td>Counter32</td>
<td>0</td>
<td>Indicates the number of bytes read from the cache of this server. This is a cumulative value, counted over the life of the server process.</td>
</tr>
<tr>
<td>1.3.6.1.4.1.2.6.120.1.1.2.3.1.3.1</td>
<td>Counter32</td>
<td>0</td>
<td>Indicates the number of requests for files stored in the cache of this server. This is a cumulative value, counted over the life of the server process.</td>
</tr>
<tr>
<td>1.3.6.1.4.1.2.6.120.1.1.2.3.1.4.1</td>
<td>Counter32</td>
<td>0</td>
<td>Indicates the number of bytes of RAM used by the cache of this server. This is a snapshot value for the HTTP Server Queue Manager only.</td>
</tr>
</tbody>
</table>
Managing the Web server

**Description**
Indicates the number of files in the cache of this server. This is a cumulative value, counted over the life of the server process.

This is a snapshot value for the HTTP Server Queue Manager only.

**Object ID**
1.3.6.1.4.1.2.6.120.1.1.2.3.1.5.1

**Type**
Counter32

**Default value**
0

**TotalTimeouts**

**Description**
Indicates the total number of timeouts on the server. This read-only information is updated as the server runs. This is a cumulative value, counted over the life of the server process. This value is not affected by any changes to the configuration of the HTTP Server.

All server address spaces under workload management update one counter, which is accessible by the HTTP Server Queue Manager.

**Object ID**
1.3.6.1.4.1.2.6.120.1.1.2.4.0

**Type**
Counter32

**Default value**
0

**LastTimeoutEntityIndex**

**Description**
This value is for future extensibility and provides support for the Application Table. This read-only value is always 1 and is not customizable.

**Object ID**
1.3.6.1.4.1.2.6.120.1.1.2.5.0

**Type**
INTEGER

**Default value**
1

**LastTimeoutRemoteAddress**

**Description**
Provides the IP address of the machine that timed out last. This read-only value is updated by server code as the server runs. This value is not affected by any changes to the configuration of the HTTP Server.

**Object ID**
1.3.6.1.4.1.2.6.120.1.1.2.6.0

**Type**
IpAddress

**Default value**
0.0.0.0

**RequestTable**
Indicates the number of requests for each method received by this server. These are cumulative values, counted over the life of the server process. These values are not affected by any changes to the configuration of the HTTP Server.

All Webserver address spaces under workload management update one counter, which is accessible by the HTTP Server Queue Manager.

**Table object ID**
1.3.6.1.4.1.2.6.120.1.1.2.7.1.2

**Type**
Table

**Default value**
0

**RequestMethod**

Description
Indicates the number of GET requests received by this server.

**Object ID**
1.3.6.1.4.1.2.6.120.1.1.2.7.1.2.1.1

**Type**
Counter32

**Default value**
0

**RequestHeadMethod**

Description
Indicates the number of HEAD requests received by this server.

**Object ID**
1.3.6.1.4.1.2.6.120.1.1.2.7.1.2.1.2

**Type**
Counter32

**Default value**
0

**RequestPostMethod**

Description
Indicates the number of POST requests received by this server.

**Object ID**
1.3.6.1.4.1.2.6.120.1.1.2.7.1.2.1.3

**Type**
Counter32

**Default value**
0

**RequestCGIMethod**

Description
Indicates the number of CGI requests received by this server.

**Object ID**
1.3.6.1.4.1.2.6.120.1.1.2.7.1.2.1.4

**Type**
Counter32
## Managing the Web server

### RequestGWAPIMethod

**Description**
Indicates the number of the HTTP Server API (GWAPI) requests received by this server.

**Object ID**
1.3.6.1.4.1.2.6.120.1.1.2.7.1.2.1.5

**Type**
Counter32

**Default value**
0

### ResponseTable

**Description**
Indicates the number of responses with these codes sent by this server. This is a cumulative value, counted over the life of the server process. This value is not affected by any changes to the configuration of the HTTP Server.

All Webserver address spaces under workload management update one counter, which is accessible by the HTTP Server Queue Manager.

**Object ID**
1.3.6.1.4.1.2.6.120.1.1.2.8.1.2

**Type**
Table

**Default value**
0

### ResponseError200

**Description**
Indicates the number of Error 200 responses (OK responses) sent by this server.

**Object ID**
1.3.6.1.4.1.2.6.120.1.1.2.8.1.2.1.200

**Type**
Counter32

**Default value**
0

### ResponseError302

**Description**
Indicates the number of Error 302 responses (Moved Temporarily responses) sent by this server.

**Object ID**
1.3.6.1.4.1.2.6.120.1.1.2.8.1.2.1.302

**Type**
Counter32

**Default value**
0

### ResponseError401
Managing the Web server

Description
Indicates the number of Error 401 responses (Unauthorized responses) sent by this server.

Object ID
1.3.6.1.4.1.2.6.120.1.1.2.8.1.2.1.401

Type
Counter32

Default value
0

ResponseError403

Description
Indicates the number of Error 403 responses (Forbidden responses) sent by this server.

Object ID
1.3.6.1.4.1.2.6.120.1.1.2.8.1.2.1.403

Type
Counter32

Default value
0

ResponseError404

Description
Indicates the number of Error 404 responses (Not Found responses) sent by this server.

Object ID
1.3.6.1.4.1.2.6.120.1.1.2.8.1.2.1.404

Type
Counter32

Default value
0

ResponseError407

Description
Indicates the number of Error 407 responses (Proxy Unauthorized) sent by this server.

Object ID
1.3.6.1.4.1.2.6.120.1.1.2.8.1.2.1.407

Type
Counter32

Default value
0

ResponseError500

Description
Indicates the number of Error 500 responses (Internal Server Error responses) sent by this server.

Object ID
1.3.6.1.4.1.2.6.120.1.1.2.8.1.2.1.500

Type
Counter32

Default value
0
Managing the Web server

ApplicationData

Description
RFC 1565.

applName

Description
The name that the network service application is known by. This read-only value is not customizable.

Object ID
1.3.6.1.2.1.27.1.1.2.1

Type
OCTET_STRING

Default value
HTTP Server

applDirectoryName

Description
The X.500 name for Web server. This read-only value is not customizable and is currently not supported by the HTTP Server.

Object ID
1.3.6.1.2.1.27.1.1.3.1

Type
OCTET_STRING

Default value
Not available

applVersion

Description
The version of software the server is running. This character value is read-only and not customizable.

Object ID
1.3.6.1.2.1.27.1.1.4.1

Type
OCTET_STRING

Default value
5.0

applUptime

Description
This value is how long the server has been up. This is a read-only value.

For workload management, this value is for the HTTP Server Queue Manager.

Object ID
1.3.6.1.2.1.27.1.1.5.1

Type
TimeTicks

Default value
0

applOperStatus
Description
Indicates the operational status of the Web server. The HTTP Server sets this value to up at server startup. It is currently a read-only value.

For workload management, this value is for the HTTP Server Queue Manager.

Additional standardized values for this MIB variable include down, halted, congested, and restarting. These values may be used in the future.

Standardized values include:

1 Up - indicates that the server is operational and available.
2 Down - indicates that the Web server is not available.
3 Halted - indicates that the Web server is operational but not available.
4 Congested - indicates that the server is operational but no additional inbound associations can be accommodated.
5 Restarting - indicates that the server is currently unavailable but is in the process of restarting and will be available soon.

Object ID
1.3.6.1.2.1.27.1.1.6.1

Type INTEGER

Default value 1

applLastChange

Description
Indicates how long from when the server came up (applUptime) that the applOperStatus changed. Currently this will always be 0 because applOperStatus is only set to up at server startup. This is a read-only value.

For workload management, this value is for the HTTP Server Queue Manager.

Object ID
1.3.6.1.2.1.27.1.1.7.1

Type TimeTicks

Default value 0

applInboundAssociations

Description
Indicates the number of inbound connections currently running or how many threads are processing received requests. This is a read-only value. This value is not affected by any changes to the configuration of the HTTP Server.

For workload management, this value is for the HTTP Server Queue Manager.
Managing the Web server

**Object ID**
1.3.6.1.2.1.27.1.1.8.1

**Type**  
Gauge32

**Default value**
0

**applOutboundAssociations**

**Description**
Indicates the number of outbound connections that the server is currently handling or how many threads are processing outbound requests. This value is 0 if the server is not acting as a proxy server. This is a read-only value. This value is not affected by any changes to the configuration of the HTTP Server.

For workload management, this value is for the HTTP Server Queue Manager.

**Object ID**
1.3.6.1.2.1.27.1.1.9.1

**Type**  
Gauge32

**Default value**
0

**applAccumulatedInboundAssociations**

**Description**
Indicates the total number of server’s inbound connections until this time. This is a read-only value. This value is cumulative and is not affected by any changes to the configuration of the HTTP Server.

For workload management, this value is for the HTTP Server Queue Manager.

**Object ID**
1.3.6.1.2.1.27.1.1.10.1

**Type**  
Gauge32

**Default value**
0

**applAccumulatedOutboundAssociations**

**Description**
Indicates the total number of server’s outbound connections until this time. This value is 0 if the server is not acting as a proxy server. This is a read-only value. This value is cumulative and is not affected by any changes to the configuration of the Webserver.

For workload management, this value is for the HTTP Server Queue Manager.

**Object ID**
1.3.6.1.2.1.27.1.1.11.1

**Type**  
Counter32

**Default value**
0
Managing the Web server

applLastInboundActivity

**Description**
Indicates the time since applUptime that the last inbound connection was made. This is a read-only value. This value is not affected by any changes to the configuration of the HTTP Server.

For workload management, this value is for the HTTP Server Queue Manager.

**Object ID**
1.3.6.1.2.1.27.1.1.12.1

**Type**
TimeTicks

**Default value**
0

applLastOutboundActivity

**Description**
Indicates the time since applUptime that the last outbound connection was made. This is a read-only value. This value is not affected by any changes to the configuration of the HTTP Server.

For workload management, this value is for the HTTP Server Queue Manager.

**Object ID**
1.3.6.1.2.1.27.1.1.13.1

**Type**
TimeTicks

**Default value**
0

applRejectedInboundAssociations

**Description**
Indicates the total number of requests the server has rejected. This is a read-only value. This value is not affected by any changes to the configuration of the HTTP Server.

For workload management, this value is for the HTTP Server Queue Manager.

**Object ID**
1.3.6.1.2.1.27.1.1.14.1

**Type**
Counter32

**Default value**
0

applFailedOutboundAssociations

**Description**
Indicates the total number of the server’s outbound requests that failed. This is a read-only value. This value is not affected by any changes to the configuration of the Webserver.

For workload management, this value is for the HTTP Server Queue Manager.

**Object ID**
1.3.6.1.2.1.27.1.1.15.1
Managing the Web server

Type: Counter32

Default value:

0

Note: The timestamp values for the HTTP Server MIB variables, applLastChange, applLastInboundActivity, and applLastOutboundActivity, vary from RFC 1565. In RFC 1565, timestamps are relative to sysUpTime. These three timestamp values are relative to applUptime.

Creating an e-mail address to receive SNMP problem reports

The HTTP Server provides a default e-mail address, webmaster, for use by management applications that send e-mail as a result of threshold exceptions. The HTTP Server does not send such e-mail notices. Use the WebMasterEmail directive to customize the mail address. The typical format for this value is user@rootname. For more information about the WebMasterEmail directive, see “WebMasterEmail - Create an e-mail address to receive SNMP problem reports” on page 382.

Providing a security password for SNMP

You can create community names (passwords). The SNMP community name authorizes a user to view the performance variables monitored by SNMP for a particular community of servers. The system administrator defines which variables from which servers can be viewed when a password is entered. If you change the SNMP community name, be sure to also change the community name specified in the file named /etc/snmpd.conf.

The default SNMP community name is public. This is kept in a dataset named hlq.PW.SRC where hlq is the high level qualifier fo TCP/IP datasets. This dataset is accessible from OS/390 UNIX System Services. The default entry is:

public 0.0.0.0 0.0.0.0

The search order is:

• /etc/pw.src
• The dataset specified on the SYSPWSRC DD statement in the agent procedure
• jobname.PW.SRC, where jobname is the name of the job used to start the SNMP agent
• STS1.TCPPARMS(PWSRC)
• hlq.PW.SRC, where hlq either defaults to TCPIP or is specified on the DATASETPREFIX statement in the TCPIP.DATA file.

The default entry can be used when testing the default community name, public.

The hlq value can be determined by:

1. Looking at the JCL for the started procedure for the SNMP agent for MVS. This is usually ‘SYS1.PROCLIB(SNMPD)’.
2. Looking in the dataset referred to by the SYSTCPD DD statement. This is usually ‘SYS1.TCPPARMS(TCPDATA)’. In this dataset member is a keyword DATASETPREFIX. The value for DATASETPREFIX is what hlq equals. For example, if the statement is DATASETPREFIX TCP, then the community names are in TCPIP.PW.SRC.

Use the SNMPCommunityName directive to define the community name used between the HTTP Server DPI subagent and the SNMP agent. For more
Enabling and disabling SNMP support

Use the SNMP directive to enable or disable SNMP support. To enable SNMP support, change the SNMP value to on. The default SNMP value is off.

For more information about the SNMP directive, see "Turning the SNMP support on and off from the IMWHTTPD program" on page 38.

Turning the SNMP support on and off from the IMWHTTPD program

Use these flags to turn the SNMP support in the HTTP Server on and off.

The -snmp flag turns the SNMP support on. The -nosnmp flag turns the SNMP support off.

This overrides what is defined in the httpd.conf file.

For more information about the IMWHTTPD program, see "IMWHTTPD program" on page 253.

System Management Facilities

With the System Management Facilities (SMF), you can request that configuration and performance data be recorded to SMF datasets. With this recorded configuration and statistical data, you can monitor Web server health, throughput, and activity.

If you want other data to be recorded in an SMF record, you can use the sample SMF plug-in called smfplug.[ch].

Configuration record data is taken from the server configuration file, http.conf, and is written after the server daemon is fully initialized. Performance record data is accumulated continuously and written at intervals defined in the http.conf file by the SMFRecordingInterval directive.

Note: When the logging queue is full, the logging thread checks to see if it needs to write any SMF records, based on whether the SMF recording interval has expired. If activity on the server is minimal, the logging queue fills slowly. In this case, the SMF records are written less frequently. You may want to adjust the interval specified by the SMFRecordingInterval directive.

When running under workload management (WLM), the HTTP Server Queue Manager writes sum totals for the Queue Manager and Queue Servers to SMF.

To write SMF records, the SMF application must be enabled to accept 103 type records. If the SMF application is not enabled, the server turns off SMF recording. To use Console Modify to turn SMF recording back on, see "Turning SMF on and off with the OS/390 operator console MODIFY command" on page 136. To enable SMF to receive the records, refer to OS/390 MVS System Management Facilities.

Your systems programmer must also permit to the user ID running httpd READ permission for the RACF BPX.SMF facility class.
Managing the Web server

For information on how to interpret SMF record headers and layouts, using and configuring SMF, dumping the SMF datasets, summarizing the information in the datasets, and getting record details, refer to the OS/390 MVS System Management Facilities book.

Turning SMF support on and off from the IMWHTTPD program

Use these flags to turn the SMF support in the HTTP Server on and off.

The -smf flag turns the SMF support on. The -nosmf flag turns the SMF support off.

This overrides what is defined in the httpd.conf file.

For more information about the IMWHTTPD program, see "IMWHTTPD program" on page 253.

Turning SMF on and off with the OS/390 operator console MODIFY command

You can use the OS/390 operator console MODIFY command to turn SMF on or off.

Configuration and performance data is written to SMF records at the moment in which you turn SMF on with a command. After that point, it is written at time intervals specified by the SMFRecordingInterval directive. To turn SMF on with the operator console MODIFY command, enter:

F IMWEBSRV,APPL=-smf

To turn SMF off with the operator console MODIFY command, enter:

F IMWEBSRV,APPL=-nosmf

For more information about the OS/390 operator console MODIFY command, see "OS/390 MODIFY console command" on page 253.

Controlling the logging of information by SMF

You can choose to have either configuration record data or performance record data, both configuration and performance record data, or no data written to SMF datasets. You can also define how often SMF writes the continuously accumulated statistical information to SMF datasets.

To select the type of information to be written to the SMF dataset, you can use the Global Log File Configuration Settings form in the Configuration and Administration forms or the SMF directive. The default setting for the SMF directive is none.

To specify how frequently performance record information is written to SMF datasets, use the Global Log File Configuration Settings form in the Configuration and Administration forms or the SMFRecordingInterval directive. The default SMFRecordingInterval is 00:15, which means performance record information is recorded every 15 minutes if the logging queue is full. If activity on the server is minimal, the logging queue fills slowly. In this case, the recording interval for performance record information may be longer than the specified interval.
For more information about the Configuration and Administration Forms see “Using the Configuration and Administration forms” on page 42. For more information about the SMF directive, see “SMF - Specify the type of information that SMF records” on page 347. For more information about the SMFRecordingInterval directive, see “SMFRecordingInterval - Specify how often performance record information is recorded” on page 347.

SMF software requirements

To use SMF, you must have LE 1.7 or PN88377 and PE UW32003 on OS/390 Release 2.

SMF record formats

The SMF record formats are configuration and performance. Each SMF record format subtype has a different data content. The information in the SMF header is the same for both record formats.

- **Configuration - Type 103 Subtype 01**
  - SMF header
  - SMF data area

- **Performance - Type 103 Subtype 02**
  - SMF header
  - SMF data area

Self-defining section of the SMF record header

The SMF record header information is valid for both performance record data and configuration data.

<table>
<thead>
<tr>
<th>Decimal Offsets</th>
<th>Hex Offsets</th>
<th>Name</th>
<th>Length</th>
<th>Format</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>SMF103LEN</td>
<td>2</td>
<td>binary</td>
<td>Record Length</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>SMF103SEG</td>
<td>2</td>
<td>binary</td>
<td>Segment descriptor</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>SMF103FLG</td>
<td>1</td>
<td>binary</td>
<td>System Indicator</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>SMF103RTY</td>
<td>1</td>
<td>binary</td>
<td>Record Type 103 (X'67')</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>SMF103TME</td>
<td>4</td>
<td>binary</td>
<td>Time stamp</td>
</tr>
<tr>
<td>10</td>
<td>0A</td>
<td>SMF103DTE</td>
<td>4</td>
<td>packed</td>
<td>Date stamp</td>
</tr>
<tr>
<td>14</td>
<td>E</td>
<td>SMF103SID</td>
<td>4</td>
<td>EBCDIC</td>
<td>System Identifier</td>
</tr>
<tr>
<td>18</td>
<td>12</td>
<td>SMF103SSI</td>
<td>4</td>
<td>EBCDIC</td>
<td>Subsystem identifier</td>
</tr>
<tr>
<td>22</td>
<td>16</td>
<td>SMF103STY</td>
<td>2</td>
<td>binary</td>
<td>Record subtype (X’01’ or X’02)</td>
</tr>
</tbody>
</table>

SMF configuration record data area (record type 103, subtype 01)

The httpd.conf configuration file provides all this data. Use the field descriptions in that file for more detail.

<table>
<thead>
<tr>
<th>Decimal Offsets</th>
<th>Name</th>
<th>Length</th>
<th>Format</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>24</td>
<td>EntityNameLen</td>
<td>4</td>
<td>binary</td>
<td>Length of Entity Name</td>
</tr>
<tr>
<td>28</td>
<td>EntityName</td>
<td>var</td>
<td>EBCDIC</td>
<td>system name</td>
</tr>
</tbody>
</table>

To determine the value for ENE, add 28 and the value for EntityNameLen, for example, ENE = 28 + EntityNameLen.

<table>
<thead>
<tr>
<th>ENE</th>
<th>EntityAddressLen</th>
<th>4</th>
<th>binary</th>
<th>length next field</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENE+4</td>
<td>EntityAddress</td>
<td>4</td>
<td>binary</td>
<td>(long) IP address</td>
</tr>
</tbody>
</table>
### Managing the Web server

<table>
<thead>
<tr>
<th>Decimal Offsets</th>
<th>Name</th>
<th>Length</th>
<th>Format</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENE+8</td>
<td>EntityPort</td>
<td>4</td>
<td>binary</td>
<td>port number being used</td>
</tr>
<tr>
<td>ENE+12</td>
<td>serverType</td>
<td>4</td>
<td>binary</td>
<td>server type (0,1,2)</td>
</tr>
<tr>
<td>ENE+16</td>
<td>applVersionLen</td>
<td>4</td>
<td>binary</td>
<td>length of next field</td>
</tr>
<tr>
<td>ENE+20</td>
<td>applVersion</td>
<td>var</td>
<td>EBCDIC</td>
<td>version of server</td>
</tr>
</tbody>
</table>

To determine the value for AVE, add 20 and the value for applVersionLen, for example, AVE = ENE+20+applVersionLen.

| AVE+             | serverRootLen         | 4      | binary    | Length of server_root                            |
| AVE+             | serverRoot            | var    | EBCDIC    | directory for server_root                        |

To determine the value for AVE, add 4 and the value for serverRootLen, for example, SRE = AVE+4+serverRootLen.

| SRE+             | doDNSLookUp           | 4      | binary    | dns lookup flag                                  |
| SRE+             | max-contentBuf        | 4      | binary    | max size of content buffer                       |
| SRE+8            | ThreadsMin            | 4      | binary    | minimum number threads                           |
| SRE+12           | ThreadsMax            | 4      | binary    | maximum number of threads                        |
| SRE+16           | IdleThreadTO          | 4      | binary    | timeout for idle threads                         |
| SRE+20           | ACLSettings           | 4      | binary    | ACL settings                                     |
| SRE+24           | UseMetaFiles          | 4      | binary    | meta file flag                                   |
| SRE+28           | DirAccess             | 4      | binary    | directory access flag                           |
| SRE+32           | inputTO               | 4      | binary    | input timeout                                    |
| SRE+36           | outputTO              | 4      | binary    | output timeout                                   |
| SRE+40           | scriptTO              | 4      | binary    | script timeout                                   |
| SRE+44           | useGMT                | 4      | binary    | GMT flag                                         |
| SRE+48           | serverImbedsHtml      | 4      | binary    | server Imbeds HTML flag                         |
| SRE+52           | secureType            | 4      | binary    | security type                                    |
| SRE+56           | sslPort               | 4      | binary    | security (SSL) port                             |
| SRE+60           | normalMode            | 4      | binary    | normal mode flag                                 |
| SRE+64           | cacheOff              | 4      | binary    | cache flag                                       |
| SRE+68           | cache_max_k           | 4      | binary    | max k cache                                      |
| SRE+72           | cache_max_f           | 4      | binary    | max file to cache                                |
| SRE+76           | cache_limit_1         | 4      | binary    | cache limit 1                                   |
| SRE+80           | cache_limit_2         | 4      | binary    | cache limit 2                                   |
| SRE+84           | cacheTimeMarginLen    | 4      | binary    | cache time margin field length                   |
| SRE+88           | cacheTimeMargin       | 4      | binary    | (long) cache time margin                         |
| SRE+92           | cacheLockTOLen        | 4      | binary    | cache lock timeout field length                  |
| SRE+96           | cacheLockTO           | 4      | binary    | (long) cache lock timeout                        |
| SRE+100          | keepExpired           | 4      | binary    | keep expired flag                                |
| SRE+104          | cacheNoConnect        | 4      | binary    | cache connect flag                               |
| SRE+108          | gcDisabled            | 4      | binary    | garbage collection flag                         |
| SRE+112          | gcDailyGCLen          | 4      | binary    | garbage collection interval                      |
| SRE+116          | gcDailyGC             | 4      | binary    | (long) garbage collection interval               |
| SRE+120          | gcMemUsage            | 4      | binary    | garbage collection mem use                       |
### SMF performance record data area (record type 103, subtype 02)

The SNMP MIB provides all these performance numbers. For more details, use the descriptions in "Object IDs and variable names for the HTTP Server MIB" on page 118.

<table>
<thead>
<tr>
<th>Decimal Offsets</th>
<th>Name</th>
<th>Length</th>
<th>Format</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SRE+124</td>
<td>ProxySomething</td>
<td>4</td>
<td>binary</td>
<td>proxy flag</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Decimal Offsets</th>
<th>Name</th>
<th>Length</th>
<th>Format</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>24</td>
<td>EntityNameLen</td>
<td>4</td>
<td>binary</td>
<td>length of entity name</td>
</tr>
<tr>
<td>28</td>
<td>EntityName</td>
<td>var</td>
<td>EBCDIC</td>
<td>system name</td>
</tr>
</tbody>
</table>

To determine the value for ENE, add 28 and the value for EntityNameLen, for example, ENE = 28 + EntityNameLen.

<table>
<thead>
<tr>
<th>Decimal Offsets</th>
<th>Name</th>
<th>Length</th>
<th>Format</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENE</td>
<td>EntityAddressLen</td>
<td>4</td>
<td>binary</td>
<td>length of next field</td>
</tr>
<tr>
<td>ENE+4</td>
<td>EntityAddress</td>
<td>4</td>
<td>binary</td>
<td>IP address (long)</td>
</tr>
<tr>
<td>ENE+8</td>
<td>EntityPort</td>
<td>4</td>
<td>binary</td>
<td>port number being used</td>
</tr>
<tr>
<td>ENE+12</td>
<td>serverType</td>
<td>4</td>
<td>binary</td>
<td>server type (0,1,2)</td>
</tr>
<tr>
<td>ENE+16</td>
<td>applVersionLen</td>
<td>4</td>
<td>binary</td>
<td>length of next field</td>
</tr>
<tr>
<td>ENE+20</td>
<td>applVersion</td>
<td>var</td>
<td>EBCDIC</td>
<td>version of server</td>
</tr>
</tbody>
</table>

To determine the value for AVE, add 20 and the value for applVersionLen, for example, AVE = ENE+20 + applVersionLen.

<table>
<thead>
<tr>
<th>Decimal Offsets</th>
<th>Name</th>
<th>Length</th>
<th>Format</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AVE</td>
<td>TotalCurrentThreads</td>
<td>4</td>
<td>binary</td>
<td>number of threads currently used; represents the number of threads currently in use.</td>
</tr>
<tr>
<td>AVE+4</td>
<td>MaxThread</td>
<td>4</td>
<td>binary</td>
<td>maximum number of threads defined; static number</td>
</tr>
<tr>
<td>AVE+8</td>
<td>Request</td>
<td>4</td>
<td>binary</td>
<td>number of requests received; cumulative from Web server startup</td>
</tr>
<tr>
<td>AVE+12</td>
<td>RequestErrors</td>
<td>4</td>
<td>binary</td>
<td>number of request errors received; cumulative from Web server startup</td>
</tr>
<tr>
<td>AVE+16</td>
<td>RequestDiscards</td>
<td>4</td>
<td>binary</td>
<td>number of requests discarded; cumulative from Web server startup</td>
</tr>
<tr>
<td>AVE+20</td>
<td>Responses</td>
<td>4</td>
<td>binary</td>
<td>number of responses sent; cumulative from Web server startup</td>
</tr>
<tr>
<td>AVE+24</td>
<td>ResponseDiscard</td>
<td>4</td>
<td>binary</td>
<td>number of responses discarded; cumulative from Web server startup</td>
</tr>
<tr>
<td>AVE+28</td>
<td>InBytes</td>
<td>4</td>
<td>binary</td>
<td>number of bytes received; cumulative from Web server startup</td>
</tr>
<tr>
<td>AVE+32</td>
<td>OutBytes</td>
<td>4</td>
<td>binary</td>
<td>number of bytes sent; cumulative from Web server startup</td>
</tr>
<tr>
<td>AVE+36</td>
<td>InUnknowns</td>
<td>4</td>
<td>binary</td>
<td>number of unknown type bytes received; cumulative from Web server startup</td>
</tr>
<tr>
<td>AVE+40</td>
<td>TotalTimeOuts</td>
<td>4</td>
<td>binary</td>
<td>number of timeouts since startup; cumulative from Web server startup</td>
</tr>
</tbody>
</table>
Managing the Web server

<table>
<thead>
<tr>
<th>Decimal Offsets</th>
<th>Name</th>
<th>Length</th>
<th>Format</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AVE+44</td>
<td>KBytesReadFromCache</td>
<td>4</td>
<td>binary</td>
<td>number of kilobytes read from the proxy cache; cumulative from Web server startup; KbytesReadFromCache x 1024 + BytesReadFromCache = total bytes read from the proxy cache</td>
</tr>
<tr>
<td>AVE+48</td>
<td>BytesReadFromCache</td>
<td>4</td>
<td>binary</td>
<td>number of bytes read from the proxy cache; cumulative from Web server startup; KbytesReadFromCache x 1024 + BytesReadFromCache = total bytes read from the proxy cache</td>
</tr>
<tr>
<td>AVE+52</td>
<td>CacheHits</td>
<td>4</td>
<td>binary</td>
<td>number of proxy cache hits; cumulative from Web server startup</td>
</tr>
<tr>
<td>AVE+56</td>
<td>BytesCacheRamInUse</td>
<td>4</td>
<td>binary</td>
<td>number of bytes of proxy cache RAM in use; represents the size of the proxy cache currently in use</td>
</tr>
<tr>
<td>AVE+60</td>
<td>CachedFiles</td>
<td>4</td>
<td>binary</td>
<td>number of proxy cached files; represents the number of files currently in the proxy cache</td>
</tr>
<tr>
<td>AVE+64</td>
<td>GETrequests</td>
<td>4</td>
<td>binary</td>
<td>number of GET requests; cumulative from Web server startup</td>
</tr>
<tr>
<td>AVE+68</td>
<td>HEADrequests</td>
<td>4</td>
<td>binary</td>
<td>number of HEAD requests; cumulative from Web server startup</td>
</tr>
<tr>
<td>AVE+72</td>
<td>POSTrequests</td>
<td>4</td>
<td>binary</td>
<td>number of POST requests; cumulative from Web server startup</td>
</tr>
<tr>
<td>AVE+76</td>
<td>CGIrequests</td>
<td>4</td>
<td>binary</td>
<td>number of CGI requests; cumulative from Web server startup</td>
</tr>
<tr>
<td>AVE+80</td>
<td>GWAPIrequests</td>
<td>4</td>
<td>binary</td>
<td>number of GWAPI requests; cumulative from Web server startup</td>
</tr>
<tr>
<td>AVE+84</td>
<td>Level200responses</td>
<td>4</td>
<td>binary</td>
<td>number of Error Level 200 (Error 200-299) responses; cumulative from Web server startup</td>
</tr>
<tr>
<td>AVE+88</td>
<td>Level300responses</td>
<td>4</td>
<td>binary</td>
<td>number of Error Level 300 (Error 300-399) responses; cumulative from Web server startup</td>
</tr>
<tr>
<td>AVE+92</td>
<td>Level400responses</td>
<td>4</td>
<td>binary</td>
<td>number of Error Level 400 (Error 400-499) responses; cumulative from Web server startup</td>
</tr>
<tr>
<td>AVE+96</td>
<td>Level500responses</td>
<td>4</td>
<td>binary</td>
<td>number of Error Level 500 (Error 500-599) responses; cumulative from Web server startup</td>
</tr>
<tr>
<td>AVE+100</td>
<td>200responses</td>
<td>4</td>
<td>binary</td>
<td>number of Error 200 responses; cumulative from Web server startup</td>
</tr>
<tr>
<td>AVE+104</td>
<td>302responses</td>
<td>4</td>
<td>binary</td>
<td>number of Error 302 responses; cumulative from Web server startup</td>
</tr>
<tr>
<td>AVE+108</td>
<td>401responses</td>
<td>4</td>
<td>binary</td>
<td>number of Error 401 responses; cumulative from Web server startup</td>
</tr>
<tr>
<td>AVE+112</td>
<td>403responses</td>
<td>4</td>
<td>binary</td>
<td>number of Error 403 responses; cumulative from Web server startup</td>
</tr>
<tr>
<td>AVE+116</td>
<td>404responses</td>
<td>4</td>
<td>binary</td>
<td>number of Error 404 responses; cumulative from Web server startup</td>
</tr>
<tr>
<td>Decimal Offsets</td>
<td>Name</td>
<td>Length</td>
<td>Format</td>
<td>Description</td>
</tr>
<tr>
<td>----------------</td>
<td>--------------------</td>
<td>--------</td>
<td>--------</td>
<td>---------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>AVE+120</td>
<td>407responses</td>
<td>4</td>
<td>binary</td>
<td>number of Error 407 responses; cumulative from Web server startup</td>
</tr>
<tr>
<td>AVE+124</td>
<td>500responses</td>
<td>4</td>
<td>binary</td>
<td>number of Error 500 responses; cumulative from Web server startup</td>
</tr>
<tr>
<td>AVE+128</td>
<td>SMFRecordInterval</td>
<td>4</td>
<td>binary</td>
<td>interval since last SMF write in seconds</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Use the following fields only if the length of the SMF record type is greater than 192 bytes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AVE+132</td>
<td>Diagnostic 1</td>
<td>4(signed)</td>
<td>binary</td>
<td>IBM internal use only</td>
</tr>
<tr>
<td>AVE+136</td>
<td>Diagnostic 2</td>
<td>4(signed)</td>
<td>binary</td>
<td>IBM internal use only</td>
</tr>
<tr>
<td>AVE+140</td>
<td>Diagnostic 3</td>
<td>4(signed)</td>
<td>binary</td>
<td>IBM internal use only</td>
</tr>
<tr>
<td>AVE+144</td>
<td>Diagnostic 4</td>
<td>4(signed)</td>
<td>binary</td>
<td>IBM internal use only</td>
</tr>
<tr>
<td>AVE+148</td>
<td>SMFConnectCnt</td>
<td>4</td>
<td>binary</td>
<td>number of connections since the last SMF record write; cumulative since the last SMF record was written</td>
</tr>
<tr>
<td>AVE+152</td>
<td>Reserved</td>
<td>4</td>
<td>binary</td>
<td>Reserved for future use</td>
</tr>
<tr>
<td>AVE+156</td>
<td>DNSMax</td>
<td>8</td>
<td>binary</td>
<td>maximum DNS lookup response time in seconds; value in C floating point (double) format; represents the longest response time since Web server startup</td>
</tr>
<tr>
<td>AVE+164</td>
<td>DNSMin</td>
<td>8</td>
<td>binary</td>
<td>minimum DNS lookup response time in seconds; value in C floating point (double) format; represents the shortest response time since Web server startup</td>
</tr>
<tr>
<td>AVE+172</td>
<td>DNSAvg</td>
<td>8</td>
<td>binary</td>
<td>average DNS lookup response time in seconds; value in C floating point (double) format; represents the average response time since Web server startup</td>
</tr>
<tr>
<td>AVE+180</td>
<td>ServicePluginsMax</td>
<td>8</td>
<td>binary</td>
<td>maximum service plugins response time in seconds; value in C floating point (double) format; represents the longest response time since Web server startup</td>
</tr>
<tr>
<td>AVE+188</td>
<td>ServicePluginsMin</td>
<td>8</td>
<td>binary</td>
<td>minimum service plugins response time in seconds; value in C floating point (double) format; represents the shortest response time since Web server startup</td>
</tr>
<tr>
<td>AVE+196</td>
<td>ServicePluginsAvg</td>
<td>8</td>
<td>binary</td>
<td>average service plugins response time in seconds; value in C floating point (double) format; represents the average response time since Web server startup</td>
</tr>
<tr>
<td>AVE+204</td>
<td>CGIMax</td>
<td>8</td>
<td>binary</td>
<td>maximum CGI response time in seconds; value in C floating point (double) format; represents the longest response time since Web server startup</td>
</tr>
<tr>
<td>AVE+212</td>
<td>CGIMin</td>
<td>8</td>
<td>binary</td>
<td>minimum CGI response time in seconds; value in C floating point (double) format; represents the shortest response time since Web server startup</td>
</tr>
<tr>
<td>AVE+220</td>
<td>CGI Avg</td>
<td>8</td>
<td>binary</td>
<td>average CGI response time in seconds; value in C floating point (double) format; represents the average response time since Web server startup</td>
</tr>
</tbody>
</table>
### Managing the Web server

<table>
<thead>
<tr>
<th>Decimal Offsets</th>
<th>Name</th>
<th>Length</th>
<th>Format</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AVE+228</td>
<td>SSLHandshakeMax</td>
<td>8</td>
<td>binary</td>
<td>maximum SSL handshake response time in seconds; value in C floating point (double) format; represents the longest response time since Web server startup</td>
</tr>
<tr>
<td>AVE+236</td>
<td>SSLHandshakeMin</td>
<td>8</td>
<td>binary</td>
<td>minimum SSL handshake response time in seconds; value in C floating point (double) format; represents the shortest response time since Web server startup</td>
</tr>
<tr>
<td>AVE+244</td>
<td>SSLHandshakeAvg</td>
<td>8</td>
<td>binary</td>
<td>average SSL handshake response time in seconds; value in C floating point (double) format; represents the average response time since Web server startup</td>
</tr>
<tr>
<td>AVE+252</td>
<td>ProxyResponseMax</td>
<td>8</td>
<td>binary</td>
<td>maximum proxy response time in seconds; value in C floating point (double) format; represents the longest response time since Web server startup</td>
</tr>
<tr>
<td>AVE+260</td>
<td>ProxyResponseMin</td>
<td>8</td>
<td>binary</td>
<td>minimum proxy response time in seconds; value in C floating point (double) format; represents the shortest response time since Web server startup</td>
</tr>
<tr>
<td>AVE+268</td>
<td>ProxyResponseAvg</td>
<td>8</td>
<td>binary</td>
<td>average proxy response time in seconds; value in C floating point (double) format; represents the average response time since Web server startup</td>
</tr>
</tbody>
</table>

### OS/390 console commands

The HTTP Server allows you to use the OS/390 MODIFY and Workload Management console commands to display information on the Web server. For a description of command options, see [Appendix B. Commands on page 241](#).
Chapter 13. Rating Web sites and serving rated Web information

PICS overview ................................................. 143
Who can rate Web sites .................................. 143
How Web clients use PICS .............................. 144
How the HTTP Server helps you manage PICS labels ..................................................... 145
PICS for Web site administrators ..................... 145
Managing PICS labels for your Web site in each document ............................................. 146
Managing PICS labels for your Web site from a central file ............................................. 146
PICS for rating services and label bureaus .......... 146
How to manage PICS labels from a central file ................................................................. 147
Storing the PICS files on your server ............. 148
Managing PICS labels for your own Web site 148
Starting a PICS rating service and label bureau 148
How to create PICS labels .............................. 149
How to request PICS label information ............. 149
How to update the PICS configuration file ...... 150
Using the online Configuration and Administration forms ................................................. 151
Editing the PICS configuration file manually 151
PICS configuration file syntax .......................... 152
Using wildcards in the PICS configuration file 153

PICS overview

The Platform for Internet Content Selection (PICS) allows users of Internet applications, such as the World Wide Web, FTP, and Gopher, to filter the material they encounter, and accept or reject the material based on its ratings. This filtering allows parents, businesses, schools, or discerning individuals to block the access to inappropriate and objectionable material.

For the most up-to-date PICS information, see the World Wide Web Consortium’s PICS Web site at URL:

http://www.w3.org/PICS/

The specifications published at this Web site enable:

• Content providers (people who publish information on the Web) to rate and label their own documents. These can be HTML files, or other files that contain images, sound, or animations.

• Independent rating services to rate and label documents published by other Web sites and to distribute the labels to whomever requests them.

• Internet users (browsers and other clients) to request these labels and determine how to handle rated and unrated information.

The HTTP Server makes it easy for you to store and serve the rating labels for the documents you publish. It also allows you to act as a rating service or label bureau by providing a means for you to maintain and distribute rating labels for other Web sites.

Who can rate Web sites

Web sites can rate themselves or be rated by a third party, called a rating service. A rating service evaluates Web content according to their own published criteria and then distributes the labels through a label bureau. Often a rating service acts as its own label bureau and distributes its own labels.

Some rating services will also give you assistance in assessing and labeling your own site and documents. The World Wide Web Consortium publishes a list of PICS self-rating services on its Web site at URL:

http://www.w3.org/PICS/
Rating Web sites and information

The PICS specification does not determine who can or will act as a rating service. The World Wide Web Consortium publishes a list of PICS third-party rating services on its Web site. In addition, anyone who wants to can set up a rating service. You can set up such a service by:

- Deciding on a rating system
- Publishing the rating system
- Rating documents and creating the rating labels
- Establishing a Web site (URL) that clients can access to get your labels

A rating service can choose any criteria on which to rate Web sites. While some might rate Web sites for their violence or sexual content, others could choose to rate educational content, political correctness, or even how “cool” the site is. Also, a rating service can rate any and all Web sites that it wants to rate.

Having your Web site and pages rated is often desirable. In fact, it may even be necessary for your Web site to be rated in order to be viewed by a PICS-enabled client. Understanding how Web clients use the PICS labels and ratings will help make this clear.

How Web clients use PICS

PICS-enabled clients allow the users to determine which rating services they want to use and, for each rating service, which ratings are acceptable and which are unacceptable.

For example, a family might choose a rating service that rates documents according to their sexual content. The rating service might have a low rating for romance, a higher rating for passionate kissing, and yet higher ratings for more explicit sexual activity. The parents might decide that documents containing romance are the highest acceptable rating for their household. They would then configure their browser to reject all documents that are unrated or contain a higher rating from this rating service.

In another example, the Hi-Tek Systems Corporation could label its own documents with a "For Hi-Tek Use Only" and could equip all its employees with browsers configured to accept only documents with that rating.

There are several steps in this process:

The client sends a request

When a PICS-enabled client requests a document, it indicates in the request which rating services are of interest. For example, assume these parents had configured their browser to evaluate rating labels from The Best rating service. When their children click a link to an HTML document, the browser request would also ask for the rating labels that were assigned to the document by this rating service.

The server sends a response

Assume the PICS-enabled server has a copy of the labels the client is requesting. When the server receives the client’s request, it sends the labels along with the requested document. However, if the server does not support PICS or does not have copies of labels from that particular rating service, it sends the requested document anyway.

The client checks the server response first

The client first checks to see if the requested ratings labels are imbedded in
Rating Web sites and information

the document (in the meta information) or if they were sent along with the document. Some clients might accept rating information that is imbedded in the file. Others might require a separate label from a registered rating service and a guarantee that it was created by that service. If the client successfully finds the label information it wanted, it evaluates the rating and either displays the document or blocks it and displays a message.

The client contacts the rating service, if necessary
If the client does not receive the label information with the requested document from the server, it might send a subsequent request directly to the rating service asking for the label information for that document. This requires a second connection, which takes longer and can discourage future visits to that site. The browser waits until the label information is returned before it displays any data.

Faster response time is the main reason why rating labels for a site should reside at the site.

How the HTTP Server helps you manage PICS labels

Whether your Web server publishes Web documents or you are a rating service and want to provide the labels for other Web sites, the HTTP Server can help you manage PICS labels.

Note: If you are going to use your server to rate your own documents or to run a label bureau, we strongly suggest that you use the default server port (80).

PICS for Web site administrators

As more browsers are configured to block access to unrated documents, it behooves you to have your Web site rated. And because it saves time when a browser can get the ratings when it sends its initial request, it behooves you to store the ratings for your pages on your own server. With the HTTP Server’s PICS support, you can manage the labels from one central file and serve them with requested pages and documents. These labels can be:

- Self-assessed according to your own criteria
  If you are establishing your own rating service, you can rate your own site according to your published criteria.

- Self-assessed according to the published criteria of a voluntary rating service
  Voluntary rating services, such as SafeSurf (http://www.safesurf.com) trust Web administrators to be honest in the assessment of their own pages.

- Assessed by a third-party rating service according to the service’s criteria
  In this case, you might contact the rating service and request that they rate your Web site (if they have not already done so) and send you the label information. In fact, you might want to contact several rating services to have your site rated for different subject criteria. If the third-party rating services have the HTTP Server, this process can be simplified with an electronic request. See “How to request PICS label information” on page 150.

Once the ratings are established, Web administrators can do one of three things:

- Manually edit each of their HTML files, inserting the rating information in the headers. See “Managing PICS labels for your Web site in each document” on page 146.
Rating Web sites and information

- Use the label information to create PICS-compliant rating labels, store the labels in their file system, and use the PICS configuration file to manage and transmit them. See “Managing PICS labels for your Web site from a central file”.

- Let the system automatically store the transmitted rating labels and update your PICS configuration file for you. This can only be done when electronically requesting labels for a third-party rating service that has the HTTP Server. See “How to request PICS label information” on page 150.

Managing PICS labels for your Web site in each document
You can edit each of your HTML files and embed PICS ratings information in the meta element of the document header. This process is entirely manual and therefore time-consuming, error-prone, and difficult to maintain. It does not incorporate any of the security mechanisms (message digest, digital signature, etc.) that would guarantee the validity of the label, if this is important to the requesting client. The PICS specification explains how you can embed rating information in each document. It is not covered here.

To access the PICS specification, go to URL:
http://www.w3.org/PICS/

Managing PICS labels for your Web site from a central file
The HTTP Server’s PICS support allows you to store the rating labels for all the documents on your Web site and manage them from a central file. The labels are sent along with your Web pages when a client requests them.

In addition to the rating labels, you must also have a PICS-compliant rating system description file that describes the rating system used to rate your documents. These are called RAT files, and rating services will provide them along with their labels.

Once you have both the labels and the RAT file, you can use the PICS configuration file to manage these labels from a central point. See “How to manage PICS labels from a central file” on page 147.

PICS for rating services and label bureaus
Because many Webmasters will want their pages rated, you have an opportunity to provide a service to a large number of Web sites.

- Content providers will contact your organization to request that you rate their Web site and provide them with the labels so that they will be able to serve the labels along with their Web documents themselves.

- Clients will connect to your server electronically to request labels for pages they are attempting to view only when they cannot get the label information with the requested pages.

The PICS configuration file provides you with the means to manage the labels for other Web sites and transmit them when requested.

The PICS specifications allow anyone to set up a rating service, define the criteria by which they rate Web sites and documents, and then provide the ratings. With PICS support, you can establish your server as a rating service and maintain and distribute labels for other Web sites. You can rate documents at a Web site individually or use wildcard characters to quickly assign the same rating to all or part of a Web site’s offerings. You will need to create these labels and your own RAT file. The RAT file is a PICS-compliant rating file that describes the rating
system used to rate documents. Once you have both the labels and the RAT file, you can use the PICS configuration file to manage these labels from a central point. Your server will then be able to automatically send the rating labels you have assigned when a client requests them. See “How to manage PICS labels from a central file”.

If a Web site that you have rated requests the labels for their pages, you can also provide them with all their current ratings. Unfortunately, the World Wide Web Consortium has not yet defined a standard for the label bureaus or rating services to send a Web site all their label information. This means that the method for this exchange will have to be determined by the rating services and the Web sites that ask for them.

If the Web sites and the rating service (or label bureau) both have the HTTP Server, they can electronically exchange rating labels and label entries for their PICS configuration file. In this case, the rating labels will be automatically stored on the server and the PICS configuration file will be updated so that it can transmit the labels with the requested documents.

If not, we are assuming that the rating services will send a file of all the required label information to the Web site administrators. Once the administrators receive this information, they will use whatever method is available on their server to create PICS rating labels and enable their server to transmit them with the requested documents.

How to manage PICS labels from a central file

Managing PICS labels requires three things:

1. A rating (RAT) file that describes the ratings
   If you are starting your own rating service or label bureau, you will need to create a file that describes your rating system. This file must be in the machine-readable format detailed in the PICS technical specifications and it should have the .rat extension. If you are getting your labels from a third-party rating service, you must also get a copy of their RAT file.

2. The rating labels themselves
   Whether you are maintaining labels for your own Web site or, as a rating service, maintaining labels for other sites, you will need to store the labels in your server’s file system, one label per file. Rating services will rate documents on the Internet and create the rating label files themselves. Web sites will either rate their own site and create the label files or they will request the rating labels from third-party rating services.
   If you are getting your rating labels from a third-party rating service that also has the HTTP Server, you can request the labels electronically. They will be sent and stored directly on your system for you. Otherwise, you may need to do some editing of the information you receive before creating rating labels to store in your file system.
   When creating PICS rating labels, be sure to follow the PICS specification. See “How to create PICS labels” on page 149. We recommend you use .lbl for the extension on your label files. We have included a predefined AddType directive in the configuration file for this extension.

3. The PICS configuration file
   This file provides a mapping between the actual rating labels and the documents they rate. It enables the server to quickly respond to HTTP, FTP, and Gopher requests. If you are getting your rating labels from a third-party...
Rating Web sites and information

rating service that also has the HTTP Server, your PICS configuration file will automatically be updated with entries for the labels you receive. If you are a rating service or if you receive rating labels from third-party rating services that have a different server, you will need to maintain the PICS configuration file yourself. You can use the online Configuration and Administration Forms to update and maintain this file or you can edit it manually. See “How to update the PICS configuration file” on page 151.

Storing the PICS files on your server

You will need to store both the RAT file and rating labels in files on your server.

The RAT file should be available from a rating service’s Web site. The rating labels must be stored one label per file.

You can use any directories, subdirectories, and file names that make sense at your site and for your implementation. We recommend that Web sites have a separate directory or subdirectory for each third-party rating service that they use. This is required for automatic updates when requesting labels from rating services that have the HTTP Server.

Our examples use a file extension of .lbl on each rating label file. This is also the extension for any label files the server transmits electronically.

Managing PICS labels for your own Web site

Follow these steps to store rating labels in your file system and configure your server so it sends these labels when clients request them.

1. Obtain a copy of the RAT file from the rating services you want to use and store it in your file system on your server.
2. If you are getting rating labels from a third-party rating service that has the HTTP Server:
   • Use the online Configuration and Administration Forms to request the labels and the entries for your PICS configuration file electronically. When you receive these files, your server will automatically be updated for you. See “How to request PICS label information” on page 150.

If you are not getting rating labels from a third-party rating service that has the HTTP Server:
   • Obtain the ratings from the third-party rating service or rate your own documents.
   • Create labels according to the format published in the PICS specification. See “How to create PICS labels” on page 149.
   • Store the labels in separate files, one label per file, in your server’s file system.
   • Tell your server which documents are rated, where the actual rating labels can be found, and which rating service provided the labels. You do this by adding entries to the PICS configuration file to associate the rated documents with their label files. You can use the online Configuration and Administration Forms to update and maintain this file or you can edit it manually. See “How to update the PICS configuration file” on page 151.

Starting a PICS rating service and label bureau

Follow these steps to configure your server as a PICS rating service, store rating labels for other Web sites, and serve them in response to client requests.
1. Define a rating system and create your own RAT file. Check the World Wide Web Consortium’s PICS specification for instructions on how to do this. It includes the syntax for the machine-readable format of the RAT file.

You can access the PICS specification at URL:

http://www.w3.org/PICS/

2. Establish two URLs for your rating service. One URL identifies your service by name. Include this URL in your RAT file. The other URL is for label requests. You must direct all the label requests that come to your server to this specific URL.

The PICS specification has no requirements regarding these URLs; you may choose any URL that you like.

Add the Service directive to the configuration file to inform the server that you are a PICS service and specify where to direct the PICS rating label requests. To add the Service directive, use the Request Routing form in the Configuration and Administration forms. For example:

```
Service /Ratings INTERNAL:PICS-Ratings
```

Replace /Ratings with the path and file name portion of the URL you will use for label requests. For example, if you publish the URL http://www.coolratings.com/CoolSite, you would only include /CoolSite in the Service directive.

**Note:** It is advisable to use the Configuration and Administration form, Request Routing, to ensure correct placement of the Service directive in the configuration file. If the Service directive is inserted manually, place it with the default Service directives defined in the configuration file.

3. Rate documents and Web sites according to your established rating system.

4. Create rating labels for these documents and sites and store them in your server’s file system, one label per file. See “How to create PICS labels.”

5. Tell your server which documents you have rated, what host serves them, and where the labels can be found in your file system. You do this by putting entries in the PICS configuration file that associates the rated documents and their specific label files. You can use the online Configuration and Administration Forms to update and maintain this file or you can edit it manually. See “How to update the PICS configuration file” on page 151.

6. Make the URL you will use for label requests known to the public.

   Notify all your subscribers and users to send their requests for rating labels to this URL. PICS-enabled clients and servers will use this URL to contact your server for labels.

---

## How to create PICS labels

In general, a label file is a text file containing a label. Carefully review the format of labels given by the PICS Rating Services and Rating Systems specification.

You can access the PICS specification at URL:

http://www.w3.org/PICS/

## PICS label extensions

The HTTP Server has added extensions to this format to save you repetitious data entry and to allow you to add comments.
Rating Web sites and information

Comments for your own use
You can insert comments for your own use into label files. Begin these comment lines with '#'. Lines beginning with '#' are not sent to clients. This type of comment is an addition to the “comment” statements used inside labels. “Comment” statements in labels are sent to clients.

Additional variables
You can insert some variables in label files:

%%URL%%
The current URL will be substituted for this variable. When the server receives a request for a rating label that contains for %%URL%%, it replaces this variable with the correct for statement before sending the label.

Note: Do not use this variable on generic labels (those that apply to multiple files).

%%SERVICENAME%%
The service name requested will be substituted for this variable. When the server receives a request for a rating label that contains for %%SERVICENAME%%, it replaces this variable with the correct service statement before sending the label.

How to request PICS label information

If a third-party rating service has the HTTP Server, you can electronically request rating labels for all the documents on your Web site that the third-party service has rated. As a response to that request, you will receive both rating labels and PICS configuration file label entries. Both types of information will automatically be stored on your server.

To electronically request rating label and entries for automatic update:

1. From the default home page (Frntpage.html), select Configuration and Administration Forms. When prompted, enter the administration user ID and password you have set up.
2. Select PICS Services Configuration. This displays the PICS Services Configuration main page.

Note: The third-party rating service must have the HTTP Server for you to use this feature. If not, the request fails.

If the third-party rating service has rated your Web site, it will return both the rating labels and label entries for your PICS configuration file. The rating labels will be stored in the directory you specified on the form. The label entries will automatically be added to your PICS configuration file.

If the third-party rating service has not rated your Web site, it will return a response indicating that it does not have the information you requested.
How to update the PICS configuration file

The HTTP Server provides the PICS configuration file for you to manage PICS labels from a central point and serve them when clients request them. You can use the online Configuration and Administration forms to add, modify, and delete the label entries in the PICS configuration file, or you can edit the file and maintain the data manually.

Using the online Configuration and Administration forms

1. From the default home page (Frntpage.html), select Configuration and Administration Forms. When prompted, enter the administration user ID and password you have set up.
2. Select PICS Services Configuration. This displays the PICS Services Configuration main page.
3. If you are maintaining labels for your own Web site:
   a. Select Register Third-Party Rating Services to register the services that have sent you labels and identify their RAT files. With the PICS example files, initially you will have one entry for the The Best rating service, http://www.coolness.raleigh.ibm.com/ratings/V1.html, along with its RAT file, coolness.rat.
   b. Select Maintain PICS Label Entries for Your Web Site to view, add, modify, or delete the entries that associate specific documents or pages with your rating labels.

If you are maintaining labels for other Web sites:
   a. Ensure that you have your RAT file stored in your file system.
   b. Select Register Your Own Rating Service to register the location of your RAT file on your server.
   c. Select Maintain PICS Label Entries for Other Web Sites to view, add, modify, or delete the entries that associate specific documents or pages with your rating labels.

Editing the PICS configuration file manually

The exact name and location of the PICS configuration file is etc/ics_pics.conf.

The configuration file consists of a list of paragraphs. There are three types of paragraphs.

- **LabelsFor**
  Specifies the ratings given by a particular rating service for documents on a given Web server. For example, one LabelsFor paragraph could cover ratings according to the RSAC rating system for documents on the local server, while another paragraph could cover ratings according to the The Best rating system for documents on the local server.

- **DefineService**
  Lists local label files associated with a third-party rating service.

- **DefineLBService**
  Lists local label files associated with your own label bureau or rating service.

**Note:** The PICS configuration file associates Web documents with files containing labels. The labels themselves are stored in separate files, not in the PICS configuration file.
Rating Web sites and information

PICS configuration file syntax

LabelsFor:

The first line of the paragraph consists of the keyword LabelsFor, the name of the server on which the rated documents are found, the name of the rating service, and an opening brace. The body of the paragraph specifies labels for sets of documents. Each paragraph ends with a closing brace.

LabelsFor servername servicename {
    /WebPath1/document1 /path/LabelFile1
    /WebPath2/document2 /path/LabelFile2
    ...and so on...
}

servername
This can be the keyword LOCAL to indicate documents on this server, or it can be a full URL if documents on remote servers are being rated. Only servers acting as label bureaus (rating services) will need to use a hostname other than LOCAL. When your server is providing labels for the documents it hosts, you should always use the keyword LOCAL for the hostname. Note that you must specify the protocol and hostname without a trailing slash; thus, http://www.xyz.com is acceptable as a hostname on a LabelsFor line, but http://www.xyz.com/ is not.

servicename
The full URL where clients will send their label requests.

/Path/Path/document
The Web path and name of the document being rated. This is the path a Web client would use when requesting the document. For example, if the Naughty/Image1.gif was on the server www.rated.xyz.com, then a Web client would request http://www.rated.xyz.com/Naughty/Image1.gif.

Note: You can use wildcard characters (*) to rate multiple documents at once. See "Using wildcards in the PICS configuration file" on page 153.

/Path/LabelFile
The fully qualified name of the label file in your file system.

You cannot use wildcard characters in file names.

A special keyword, NOTLABELED, can be used in place of a label file name. This indicates that the given file(s) cannot be labeled; it serves as a shorthand way of creating a label file that contains a "not-rated" label. In the example above, a not-rated error message will be returned to any clients who request a rating for the file /Unknown.html.

For example, an actual LabelsFor paragraph might look like this:

LabelsFor LOCAL http://www.rsac.org/ratingsv01.html {
    /Naughty/Image1.gif /usr/lpp/internet/server_root/labels/AdultsOnly.lbl
    /Clean/*.html /usr/lpp/internet/server_root/labels/AllAges.lbl
    /Unknown.html NOTLABELED
}

DefineService:

The first line of the paragraph consists of the keyword DefineService, the rating service URL, the quoted name of the rating service, the location and name of the service’s RAT file, and an opening brace. The body of the paragraph lists the label
Rating Web sites and information

files associated with this service, specifying each one with the keyword LABELFILE. Each paragraph ends with a closing brace.

DefineService servicename "name-of-service" ratingfile {
   LABELFILE /path/LabelFile1 "description"
   LABELFILE /path/LabelFile2 "description"
   ...and so on...
}

servicename
The name (URL) of the rating service.

name-of-service
The name (text) of the rating service, in quotes.

ratingfile
The fully qualified name of the service’s RAT file in your file system.

/path/LabelFile
The fully qualified name of the label file in your file system.

description
A text description of the label, in quotes.

For example, an actual DefineService paragraph might look like this:

DefineService http://www.abc.org/rate.html "The ABC's of Ratings" d:\www\pics\rat\abc.rat {
   LABELFILE /usr/lpp/internet/server_root/AdultsOnly.lbl "rated XXX"
   LABELFILE /usr/lpp/internet/server_root/AllAges.lbl "rated GGG"
}

DefineLBService:

This paragraph has the same syntax and format as the DefineService paragraph. The only difference is that it uses the DefineLBService keyword. The RAT file and labels that it lists are for your own label bureau and rating service.

Using wildcards in the PICS configuration file

You can use an asterisk (*) as a wildcard only in the LabelsFor paragraphs of the PICS configuration file. When using wildcards, remember that the order of entries within a paragraph is important. For each paragraph, the HTTP Server breaks the list of rated documents into two parts: those that contain wildcards, and those that do not contain wildcards.

- When the server looks for labels for a document, it will first try to find the document in the "no-wildcards" list. Order is unimportant here. Without wildcards, each entry in the list refers to exactly one document.

- If the server cannot find a match in the "no-wildcards" list, it will try to match the document name against the entries that contain wildcards. Order is important here. The server tries to match the requested document against the wildcard entries in the order in which they appear in the configuration file and will use the first entry that matches.

For example, if you want an entry that gives /* as the WebPath/document, serving as a catchall for documents that don’t have another rating, then make this the last entry in the paragraph.
Chapter 14. Retrieving LDAP information

Introduction and concepts

The HTTP Server supports the use of the Lightweight Directory Access Protocol (LDAP). LDAP is a protocol that provides access to an X.500 directory over a TCP or SSL connection.

Notes:
1. For LDAP servers that support LDAP over SSL, all SSL function available in the HTTP Server is supported. LDAP over SSL has been tested only with the Telstra server.
2. For LDAP servers that conform to the LDAP V2 or V3 standard, all non-SSL function available in the HTTP Server is supported.

LDAP enables you to store information in a directory service and query it in a database fashion. When you use X.500 directories and LDAP, any LDAP-enabled application can store information such as user authentication information once, and other applications using the LDAP server will recognize it. Using LDAP support allows multiple HTTP Servers to share configuration information.

LDAP reduces required system resources by including only a functional subset of the original X.500 Directory Access Protocol (DAP). See the LDAP client toolkit documentation for a complete listing. The LDAP client toolkit is included when you install OS/390.

The HTTP Server LDAP support is extremely scalable. You have a choice of LDAP configurations including:
- A single LDAP server
- High availability configurations using multiple LDAP servers, for example, primary, secondary, and so on
- Different LDAP servers to be accessed for different requests. For example, requests may come in from two different IP addresses, and the Web server could contact a different LDAP server for each one.

Plug-ins can use the LDAP API directly. For information on writing LDAP APIs, see “Chapter 19. Accessing LDAP information with the LDAP API” on page 227.

This book assumes you have an existing X.500 directory service available, for example the IBM eNetwork X.500 directory. For more information, go to the URL: http://www.ibm.com/software/network/directory/
X.500 overview

X.500 is a directory service with components that provide more efficient retrieval. LDAP uses two of these components: the information model, which determines the form and character, and the namespace, which allows information to be indexed and referenced.

The X.500 directory structure differs from others in the way the information is stored and retrieved. Information is associated with attributes. A query based on attributes is generated and sent to the LDAP server, and the server returns the respective values. LDAP uses a simple, string-based approach for representing directory entries.

An X.500 directory consists of entries, which are typed depending on the ObjectClass attribute. Each entry is composed of attributes. The ObjectClass attribute identifies the type of entry (for example, person or organization), which determines which attributes are required and which are optional.

Entries, which are arranged in a tree structure, may be divided among servers in geographical and organizational distribution. They are named according to their position within the distribution hierarchy by a distinguished name (DN).

Using the example directory tree structure, you can see how the DN and relative DN differ, as well as how they are used for identifying entries.

LDAP overview

Accessing an X.500 directory requires a certain protocol, for example Directory Access Protocol (DAP). However, DAP requires large amounts of system resources and support mechanisms to handle the complexity of the protocol. To allow desktop workstations to access the X.500, LDAP was introduced.

LDAP is client/server based, and some of the heavy resources required by DAP clients are handled by the LDAP server. An LDAP server can only return results or errors to the client, requiring little from the client. If an X.500 server is unable to
answer a client request, it must chain the request to another X.500 server. The server must complete the request or return an error to the LDAP server, which in turn passes the information to the client.

LDAP

Querying the LDAP server

LDAP search filters

LDAP accesses the X.500 directory through the use of human readable strings. When these query strings are passed to the LDAP server, the server returns the distinguished name of the entry.

LDAP entries are typed, or classified, by an ObjectClass attribute to simplify searches. For example, you could search an LDAP directory whose objectclass=acl to locate all entries that are access control lists.

A search filter for an LDAP entry has the following structure:

- Filters must begin and end with parentheses. See the following examples which show the placement of parentheses in complex queries.
- Filters may contain the following Boolean comparisons:
  - & - Boolean AND
  - | - Boolean OR
  - ! - Boolean NOT
- Depending on the objectclass, an attribute name may be required.
- Filters may contain the following equality expressions
  - = - equal to
  - ~= - approximately equal to
  - >= - greater
  - <= - less
- Filters must contain the value of the attribute to search on. This may contain wildcards.

For more information on LDAP search filters, see RFC 1960.
Examples of LDAP search filters

(cn=Joe Smith)
searches the directory service for the common name of Joe Smith. Possible matches are:

Joe Smith

(!cn=Jane Doe)
queries the directory service for entries whose common name is not Jane Doe. Possible matches are:

Joe Schmoe
Adam Fosset
any name other than Jane Doe

(&objectClass=acl)((sn=Johnson)
queries all acl entries matching a surname of Johnson. Possible matches are:

Peter Johnson
Davey Johnson

(o=univ*of*carolin*)
queries the organization attribute. Possible matches are:

University of North Carolina Chapel Hill
University of South Carolina

Note: LDAP can return more than one entry. However, the Web server will not authenticate when multiple entries are returned. If the University of North Carolina and the University of South Carolina were included in the directory queried by this example, both would be returned, and the authentication would fail. The search filter must be altered.

Configuring LDAP on the HTTP Server

Current limitations:

• If the LDAP server does not allow writing via the LDAP client API (as in the case of the Lotus Notes server), the HTTP Server cannot store configuration information using the ldapadd command in the LDAP client toolkit.
• There is a known problem with storing and retrieving configuration information on the Telstra server.

Using LDAP to protect files

This section will show you how to protect files or directories using user or group information on an LDAP server.

1. From the server Front Page, click Configuration and Administration Forms.
2. Click Access Control, then Document Protection.
3. Fill in the following fields:
   • URL request template - URL template of files or directories to protect. Wildcard is accepted.
   • Select the position in the list.
   • Select whether you want the protection made in-line or as a named protection.
   • To share access control on an external shared LDAP server, select the Named LDAP setup option.
Select the existing LDAP setup from the list or select **Create a new LDAP setup**...

- Optionally, enter the IP address or host name of the requesting machine.

4. Click **Submit** to continue, or **Reset** to clear the form.

5. To create a new LDAP connection, you must provide information about the LDAP server being used. This creates an LDAPInfo directive. See [LDAPInfo - Define an external LDAP server](#) on page 318 for more information on this directive.

Enter the following general LDAP settings:

- **LDAP Setup Name** - name of the LDAP server label associated with a group of LDAP parameters. For example, `PrimaryLdapServer`
- **Protection Realm** - name of the protected area as seen by the requesting client. For example, `Administrator Access`
- **Permissions** - enter the names of the users or groups which have read, write, or delete permissions. The name or group must be a valid distinguished name. For example, `cn=group1, o=IBM, c=US`
- **Host Name** - hostname of the LDAP server. For example, `ldap.ibm.com`
- **Connect via** - transport method used. Values include TCP or SSL
- **Port Number** - optional port number on which the LDAP server is listening. The default for TCP connections is 389 and 636 for SSL
- **Key file name** - file name of the key file database. This is required if you are using SSL.
- **Key Label** - name of the certificate label the Webserver uses to authenticate with the LDAP server. This label is required if using SSL as the transport. For example, `My Server's Certificate`

6. Enter the Web server connection information:

- **Server Authentication Type** - specify the method for authenticating the Web server to the LDAP server. Possibilities are:
  - None - if the LDAP server does not require the Web server to authenticate.
  - Basic - the HTTP Server must provide a userid and password to access the information on the LDAP server. The Web server’s distinguished name is used as the userid, and the password stored in the stash file is the password.
- **Server’s Distinguished Name** - distinguished name of the Web server. This name is used as the username when accessing an LDAP server using Basic authentication.
- **Server Password** - password for the server to log on to the LDAP server. This is required only if Server Authentication type is Basic. The password will be encrypted and stored in a stash file named in the `ServerPasswordStashFile` subdirective.

7. Enter client connection information:

- **User Authentication Type** - authentication type for connections made by the client
- **User Search Base** - starting point for the LDAP server to search user names
- **User Name Filter** - filter used to convert the username as input by the user to a search filter for an LDAP entry. The default is `"(&(objectclass=person)(cn=%v1* %v2*))"` where `%v1` and `%v2` are the words typed by the user.
For example, if the user types "Pa Kel", the resulting search filter would be 
"(cn=Pa* Ke1*)". Search filter syntax is described in "LDAP search filters" on page 157.

However, because the Web server cannot differentiate between multiple returned entries, authentication fails when the LDAP server returns more than one entry. For example, if there are entries (cn=Paul Kelsey) AND (cn=Paula Kelly) using the search filter above, the authentication will fail. You must modify your search filter.

- User Name field separator - specify the set of characters used to delimit the users data. The default characters are the space, comma, and the tab (\t) character.

- User Certificate Filter - converts the information in the client certificate passed over SSL to a search filter for an LDAP entry. The default is "(|&(objectclass=person)(cn=%v1)(ou=%v2)(o=%v3)(c=%v4))". SSL certificates include the following fields, all of which can be converted to a search filter:

<table>
<thead>
<tr>
<th>Certificate field</th>
<th>Variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>common name</td>
<td>%v1</td>
</tr>
<tr>
<td>organizational unit</td>
<td>%v2</td>
</tr>
<tr>
<td>organization</td>
<td>%v3</td>
</tr>
<tr>
<td>country</td>
<td>%v4</td>
</tr>
<tr>
<td>locality</td>
<td>%v5</td>
</tr>
<tr>
<td>state or country</td>
<td>%v6</td>
</tr>
<tr>
<td>serial number</td>
<td>%v7</td>
</tr>
</tbody>
</table>

Note: When the search filter is generated, the values in the fields are placed into the matching variable fields (%v1, %v2). The following table shows the conversion:

| Certificate:          | cn=Road Runner  
|                       | o=Acme Inc      
|                       | c=US             |
| Filter                | (cn=%v1, o=%v3, c=%v4) |
| Resulting Query       | (cn=Road Runner, o=Acme, Inc, c=US) |

- Group Search Base - starting point for the LDAP server to search for group entries
- Group Name Filter - filter LDAP uses to search for group names. The default is (|(objectclass=groupOfNames)(objectclass=groupOfUniqueNames))
- Group Member Attributes - specify the names of the attributes which contain group member information. More than one attribute may be used to contain member information. The default attributes are member and uniqueMember.

8. Enter timeout settings:
- Search Time-out - time limit an LDAP server is given to complete a search
- Cache Time-out - amount of time a response returned from the LDAP server remains valid
- Idle Connection Time-out - time limit before an idle LDAP server connection is closed due to inactivity

160 HTTP Server Planning, Installing, and Using
Using LDAP to share configuration information

To share configuration information, you must store it on an LDAP server. Storing information on an LDAP server differs from storing information on other systems.

Storing information on an LDAP server is usually done by an administrator at the LDAP server (see your LDAP server documentation for more information). The LDAP client toolkit, included in S/390, is used to access the X.500 directory service. Refer to the toolkit documentation for syntax.

To view LDAP documentation, go to the OS/390 Internet Library at URL: http://www.ibm.com/s390/os390/bkserv/

Other LDAP functions are described in “Chapter 19. Accessing LDAP information with the LDAP API” on page 229.

Note: The following examples use the applicationProcess objectclass to store configuration information. Although using the applicationProcess objectclass is not required, it is recommended.

To manually search a directory server
Use the following command to search an LDAP server for a list of applicationProcess objects:

```
ldapsearch -b "o=Ace Industry, c=US" -h <hostname> "(objectclass=applicationProcess)"
```

Change "o=Ace Industry, c=US" to a search base which is valid for your LDAP server.

To add an entry
This example creates a new applicationProcess entry on the LDAP server. It assumes "o=Ace Industry, c=US" is a valid search base for your LDAP server, and that "Directory Manager" has the permission to add a new entry.

1. Create a text file named "new" containing:

```
dn: cn=Go Webserver, o=Ace Industry, c=US
objectclass: applicationProcess
cn: HTTP Server
```

2. Enter the following command:

```
ldapadd -f new -D "cn=Directory Manager, o=Ace Industry, c=US" -w <password> -h <host>
```

To modify an existing entry
This example adds configuration information stored in a local file to the LDAP server directory. You can modify an entry as often as you need.

1. Modify the "new" file to contain:

```
dn: cn=HTTP Server, o=Ace Industry, c=US
objectclass: applicationProcess
description: <path>
```

where <path> is the full path name of a file containing configuration information to store in the LDAP directory.
Note: Your original httpd.conf configuration file must contain an LDAPInfo and LDAPInclude directives. All other directives can be shared on the LDAP server.

2. Enter the following command:

```bash
ldapmodify -b -f new \
-D "cn=Directory Manager, o=Ace Industry, c=US"\n-w <password> -h <host>
```
Chapter 15. Running your server as a proxy

Web Traffic Express proxy features ........................................ 163
Setting up your proxy server .............................................. 163

Web Traffic Express proxy features

The HTTP Server includes the proxy, caching, and filtering features of IBM’s Web Traffic Express.

With these features, your server can act as a proxy and retrieve Internet data for multiple browsers. And, with the optional caching features, you can manage its caching functions to optimize server performance and minimize user response time. You will also be able to filter the content you serve using Platform for Internet Content Selection (PICS) labels.

Web Traffic Express is also a socksified proxy. It includes a flexible-client SOCKS feature that allows requests for specific IP addresses to go directly to the destination server instead of being routed through a SOCKS server.

Setting up your proxy server

Refer to the IBM Web Traffic Express for Multiplatforms User’s Guide for instructions to set up these functions and run your server as a caching proxy server. The following guidelines will help you apply the information in that book to the HTTP Server:

• In general, use the examples for the AIX operating system.
• Skip the installation instructions. Web Traffic Express features are automatically installed when you install the HTTP Server.
• Use the instructions in the Configuration Quick Start chapter to enable basic proxy and caching functions and to set up a secure connection.

Note: The proxy functions will not be enabled by default. You must set the following directives in the httpd.conf configuration file:

- ProxyAccessLog server_root/logs/httpd_proxy (where server_root is the root directory of the proxy server)
- Proxy http:*
- Proxy ftp:*
- Proxy gopher:*
- Proxy *:443

• Use the instructions in the remainder of the book to:
  - Manage the cache functions
  - Complete the automatic cache refresh configuration and start the cache agent
  - Configure PICS-based filtering
  - Complete the flexible-client SOCKS configuration
  - Set up SSL-tunneling
  - Customize header information
  - Use the Server Activity Monitor to review server activity and tune the server
Proxy server

After you’ve configured your proxy server, tell your users so they can configure their browsers to direct their requests to the proxy.
Chapter 16. Running your server with multiple IP addresses or virtual hosts

Requirements for using multiple IP addresses or virtual hosts

- Multiple IP addresses: 165
- Virtual hosts: 165

Setting up your Web server to use multiple IP addresses or virtual hosts: 166

Requirements for using multiple IP addresses or virtual hosts

You may want to use one Web server to provide Web sites for multiple customers. For example, you might have two customers (customerA and customerB), both of whom want to make information about their companies available on the Web. You might want to put both Web sites on the same machine if the expected number of requests for the information is not great enough to justify a separate machine for each customer.

With the HTTP Server, you can use multiple IP addresses, virtual hosts, or both to provide multiple Web sites on one server.

Multiple IP addresses

To use multiple IP addresses, your server must be installed on a machine with multiple network connections.

If you have only one network connection and run two instances of the server on the same machine, then only one server has the benefit of using the default port number. Requests to the other server would have to include a port number.

If your machine has two network connections, you can run just one instance of the server and assign each customer to a different IP address. For each IP address you would define a different host name. So customerA could be www.customerA.com on IP address 9.67.106.79 and customerB could be www.customerB.org on IP address 9.83.100.45. You could then configure the server to serve a different set of information depending on the IP address the request comes in on. Because the server can accept requests from the default port of each network connection, requests to either host name would not require a port number.

If your server has multiple IP addresses, you can associate a specific Secure Sockets Layer (SSL) key database label with each individual IP address. For more information, see "SSL support for multiple IP addresses" on page 53.

Virtual hosts

With virtual hosts, no additional hardware is required, and you can save IP addresses. However, clients must support HTTP 1.1 or HTTP 1.0 with 1.1 Extensions.

With virtual hosts, you can run just one instance of the server and assign each customer to a different host. In the domain name server, you define your hosts and associate them with the IP address of your server. You can then configure the server to serve a different set of information depending on the host for which the request is made. Requests do not require a port number.
Multiple IP addresses or virtual hosts

To take advantage of SSL support for multiple IP addresses (multiple virtual hosts), you must assign each host name a different IP address and then define a specific key database label for each IP address. For more information, see “SSL support for multiple IP addresses” on page 53.

Setting up your Web server to use multiple IP addresses or virtual hosts

Setting up your Web server to use multiple IP addresses or virtual hosts is very similar. For multiple IP addresses, you’ll need to specify the IP address a request comes in on and for virtual hosts, you’ll need to specify the host name for which a request is made.

You configure the server to serve different information for each customer by indicating that certain parts of your configuration apply only to requests coming in on certain addresses or for certain hosts. You can configure three key parts of your server so that requests are processed differently based on the IP address they come in on or the host name in the URL.

- Welcome pages
- Mapping rules
- Access control

Welcome pages

You can specify a different set of file names to use as welcome pages depending on the address a request comes in on or the host name in the URL. The file names you define as welcome pages determine how the server responds to requests that do not contain a file name.

For example, you might want to specify that homeA.html is a welcome page only for requests received on 9.67.106.79 or for hostA, and homeB.html is a welcome page only for requests received on an address 9.83.100.45 or for hostB.

From the Configuration and Administration forms page, you can configure your list of welcome pages by clicking on Initial Page. From the Initial Page form, click the help icon for information on defining welcome pages and how to associate a welcome page file name with an IP address or a host name.

Alternatively, if you are editing the configuration file, you can add an IP-address or hostname at the end of a Welcome directive to associate a welcome page file name with an IP address or a host name. For details, see the description of the Welcome directive in “Directories and Welcome Page - Set viewing options” on page 293.

Mapping rules

You can specify a different set of mapping rules for the server to use depending on the address a request comes in on or the host name in a URL. Mapping rules map a request to a physical file on the server and determine whether the server processes a request.

For example, you might want to specify that a request beginning /cgi-bin/ received on address 9.67.106.79 or for hostA is mapped to the /customerA/cgi/ directory, and the same request received on 9.83.100.45 or for hostB is mapped to the /customerB/cgi/ directory.
Multiple IP addresses or virtual hosts

From the Configuration and Administration forms page, you can configure your mapping rules by clicking on Request Routing. From the Request Routing form, click the help icon for information on how to use mapping rules and how to associate a mapping rule with an IP address or a host name.

Alternatively, if you are editing the configuration file, you can add an IP-address or hostname at the end of Exec, Fail, Map, Pass, and Redirect directives to associate the directive with an IP address or a host name. For details, see the description of these directives in "Resource mapping - Redirect URLs" on page 359.

Access Control

You can activate different protection rules for a request based on the address the request comes in on or the host name in a URL. Protection rules are defined in protection setups and determine how your server controls access to files and programs.

For example, you might want to specify that a request beginning /cgi-bin/ received on address 9.67.106.79 or for hostA is protected by the rules in a protection setup named PROT-A and the same request received on 9.83.100.45 or for hostB is protected by the rules in a protection setup named PROT-B.

From the Configuration and Administration forms page, you can configure how protection is activated by clicking on Document Protection. From the Document Protection form, click the help icon for information on protecting documents and how to associate protection with an IP address or a host name.

Alternatively, if you are editing the configuration file, you can add an IP-address or hostname at the end of DefProt and Protect directives to associate the directive with an IP address or a host name. For details, see the description of these directives in "Access control - Set up access control for the server" on page 275.
Part 6. Programming
Chapter 17. Writing Common Gateway Interface programs

Overview of the CGI ................................................. 171

CGI and dynamic documents ................................. 172
Uses for CGI ...................................................... 172
FastCGI support .................................................. 173

Overview ......................................................... 173
Updates to the Web server configuration file for FastCGI ................................. 173
Updates to the FastCGI configuration file .............. 173
Local directive and subdirectories .......................... 174
External directive and subdirectories ..................... 175
Installing the FastCGI Developer’s Kit .................. 175
Step 1. Download the FastCGI Developer’s Kit to your workstation .................. 175
Step 2. Transfer the fcgi-devkit-2_1.tar.gz file to OS/390 ............................. 176
Step 3. Uncompress and reload the fcgi-devkit-2_1.tar.gz file ..................... 176
Step 4. Rebuild the FastCGI Developer’s Kit on OS/390 ............................. 176
Hints and tips .................................................... 178

Forms and data processing .................................. 178
Creating HTML documents that reference CGI programs .................................. 179
Example ......................................................... 180

Sending information to the server ................................................. 181
Processing the information ................................................. 181
Parsing .......................................................... 181
Data manipulation ............................................... 182
Response generation ............................................. 182
Returning output .................................................. 184
Protecting your programs .......................................... 184

Environment variables ............................................. 184
Processing standard search (ISINDEX) documents ............................................. 184
Passing SSL environment variables to a CGI program ............................................. 185
Creating CGI source .............................................. 185
Parsing Routines .................................................. 185
String Compare Reminders .......................................... 185
Running CGI programs written in Java ................................................. 186
Response generation ............................................. 187

CGI Examples .................................................... 188
CGI C program .................................................. 188
Example (cgixmp) ............................................... 188
CGI REXX program .............................................. 193
Example ......................................................... 193
CGI shell script .................................................. 195
Example ......................................................... 195

Overview of the CGI

CGI is a standard, supported by almost all Web servers, that defines how information is exchanged between a Web server and an external program (CGI program). CGI programs are stored in the cgi-bin subdirectory.

CGI programs can be written in any language supported by the operating system on which the server is run. The language can be a programming language, like C++, or it can be a scripting language, like Perl or REXX. The HTTP Server also supports CGI programs written in the Java language. Programs written in programming languages need to be compiled, and typically run faster than uncompiled programs. On the other hand, those written in scripting languages tend to be easier to write, maintain, and debug.

The functions and tasks that CGI programs can perform range from the simple to the very advanced. In general, those that perform the simple tasks are called CGI scripts (because they are not compiled). Those that perform more complex tasks are often called gateway programs. In this chapter, we refer to both types as CGI programs.

There are many uses for CGI programs. Basically, they are designed to handle dynamic information. Dynamic in this context refers to temporary information that is created for a one-time use and not stored anywhere on the Web. This information may be a document, an e-mail message, or the results of a conversion program.
CGI and dynamic documents

There are many types of files that exist on the Web. Primarily they fall into one of the following categories:

- Images
- Multimedia
- Programs
- HTML documents

Servers break HTML documents into two distinct types:

- Static
- Dynamic

**Static documents** exist in source form on the Web server. **Dynamic documents** are created as temporary documents to satisfy a specific, individual request.

Consider the process of "serving" these two types of documents. Responding to requests for static documents is fairly simple. For example, Jill User accesses the Acme Web server to get information on the Pro-Expert gas grill. She clicks on Products, then on Grills, and finally on Pro-Expert. Each time Jill clicks on a link, the Web browser uses the URL attached to the link to request a specific document from the Web server and the server responds by sending a copy of the document to Jill’s browser.

But, what if Jill then decides she wants to search through the information on the Acme Web server for all documents that contain information on Acme grills, such as news articles, press releases, price listings, and service agreements? This is a more difficult request to process. This is not a request for an existing document. Instead, it is a request for a dynamically generated list of documents that meet certain criteria. This is where CGI comes in.

Uses for CGI

You can use a CGI program to parse the request, search through the documents on your Web server, and create a list with hypertext links to each of the documents that contain the specified word or string.

HTML allows you to access resources on the Internet using many protocols by specifying the protocol in the URL. One such protocol is `mailto`. If your HTML document includes a mailto link followed by an e-mail address, the link will result in a generic mail form.

What if you wanted your customers to provide specific information, such as how often they use the Web or how they heard about your company? Rather than using the generic mail form, you can create a form that asks these questions and more. You can then use a CGI program to interpret the information, include it in an e-mail message, and send it to the appropriate person.

CGI programs are not limited to processing search requests and e-mail. They can be used for a wide variety of purposes. Basically, anytime you want to take input from the browser and generate a response, you can use a CGI program. For example, many people are interested in how often people have visited their home page. A common way to track this is with a CGI program that counts the number of requests for this home page and displays the new total each time someone links to it.
Note: Be aware that using CGI to serve documents can significantly impact your server's performance. Serving a document using CGI can take as much as 10 times more resources than serving a static document.

FastCGI support

Overview

FastCGI is a way to combine the advantages of normal CGI programming with some of the performance benefits you get by using the GWAPI interface.

FastCGI is written by Open Market, Inc. and is an extension to normal Web server processing. FastCGI allows you to start applications in independent address spaces and to pass requests for these applications from the Web server to these processes. The communication is handled by the TCP/IP sockets interface or by a UNIX domain socket bind path in the HFS file system.

By default, FastCGI support in the Web server is disabled. To enable support, you need to define directives in the following configuration files:

- Web server configuration file: httpd.conf
- FastCGI configuration file: lgw_fcgi.conf

For instructions, see "Updates to the Web server configuration file for FastCGI" and "Updates to the FastCGI configuration file".

You must also download and install the Open Market toolkit. The FastCGI Developer's Kit, which includes the toolkit, sample applications, and information on FastCGI, can be found on the Open Market Web site at URL:


For instructions on installing the Developer's Kit on OS/390, see "Installing the FastCGI Developer's Kit" on page 175.

Updates to the Web server configuration file for FastCGI

Directives, their paths, and libraries that are essential for the operation of FastCGI support are:

- ServerInit /usr/lpp/internet/bin/libfcgi.so:FCGIInit /etc/lgw_fcgi.conf
- Service /fcgi-bin/* /usr/lpp/internet/bin/libfcgi.so:FCGIDispatcher*
- ServerTerm /usr/lpp/internet/bin/libfcgi.so:FCGIStop

Notes:

1. The ServerInit directive specifies the location of the FastCGI configuration file, lgw_fcgi.conf.
2. The Service directive indicates the path of the FastCGI object library.
3. The ServerTerm directive stops the FastCGI process initialization if the Web server is restarted.

Updates to the FastCGI configuration file

All FastCGI applications have to be registered in the lgw_fcgi.conf file. An example of this file is in the /usr/lpp/internet/samples/config directory. All these applications are started during server initialization so you must stop and restart the Web server to use updates made after initialization.

In general, FastCGI supports two types of applications:
Local

The FastCGI task runs on the same system as the Web server. Communication is handled by using a dedicated TCP/IP socket or a BindPath in a character special HFS file.

External

The FastCGI task runs on a remote system. Communication can only be handled using a dedicated TCP/IP socket. Note that you need a Web server with FastCGI server support to control the startup and closedown of the application address spaces on the remote system.

You use Local and External directives to identify FastCGI applications in the lgw_fcgi.conf file.

Note: You can use more than one Arg or Environ subdirective to pass multiple values. For all other subdirectives, the Web server uses only the last subdirective in the configuration file.

Local directive and subdirectives

The Local directive indicates that the FastCGI application is on the same workstation as the Web server and that the Web server will start and control the application. Subdirectives are:

Exec fully-specified path
   Required. Name and path of the FastCGI application.

Role Responder | Authorizer
   Required. Type of FastCGI application:
   • Responder applications perform like a normal CGI program.
   • Authorizer applications perform user authentication.

URL server URL
   Required. The relative uniform resource location of the FastCGI application.

BindPath | Port Unix domain socket bind path | Port number
   Required. The path name of the file or the port number which will be used to communicate between the Web server and the FastCGI application.

NumProcesses number
   Optional. The number of processes the Web server will create for the FastCGI application. The default is one.

NPH-Scriptflag
   Optional. Specifies that the FastCGI application will produce nonparsed header output.

Arg command line argument
   Optional. Allows the user to enter additional command line arguments to start the FastCGI application.

Environ environment_variable=value
   Optional. Allows the user to set environment variables to pass to the FastCGI application at initialization.

User userID
   Optional. Identifies the userID of the FastCGI application.

   The DefaultUser directive can be used to identify the userID of the FastCGI application if a User directive is not specified. The User subdirective overrides the DefaultUser directive.
For more information on the options available for the Local directive, see the comments in the FastCGI configuration file: lgw_fcgig.conf

External directive and subdirectives
The External directive identifies a FastCGI application that resides on a remote workstation. Subdirectives are:

URL server URL
Required. The relative uniform resource location of the FastCGI application.

Host host name or IP address
Required. The host name or IP address of the workstation the Web server should connect to.

Port port number
Required. The port number which will be used to communicate between the Web server and the FastCGI application.

Role Responder | Authorizer
Required. Type of FastCGI application:
• Responder applications perform like a normal CGI program.
• Authorizer applications perform user authentication.

NPH-Script flag
Optional. Indicates that the FastCGI application produces nonparsed header output.

For more information on the options available for the External directive, see the comments in the FastCGI configuration file: lgw_fcgig.conf

Installing the FastCGI Developer’s Kit
The current implementation of the FastCGI Developer’s Kit from Open Market, Inc., supports programs written in C/C++, Perl, and Java.

Note: The Perl and Java FastCGI implementations have to be modified for use on OS/390. At the time of this writing, there were no ported versions available.

To install the C version of the FastCGI Developer’s Kit:

Step 1. Download the FastCGI Developer’s Kit to your workstation
Download the Developer’s Kit from the Open Market Web site at URL:
http://www.fastcgi.com

The Developer’s Kit is distributed as a UNIX compressed gzip file and requires the gzip utility to uncompress it into a tar file. To download the gzip utility, go to the OS/390 UNIX Tools and Toys Web page at URL:

Click Ported Tools to access the download page.

To install the gzip utility, use the installation instructions on the Web site. Make sure your REGION size is large enough to run the make command. A size of 48000 in the TSO logon screen was used when testing this procedure. To unpack the gzip utility, issue the following command:
pax -o from=ISO8859-1,to=IBM-1047 -rf gzip-1_2_4.tar

Hints and tips: The installation instructions tell you to use the following settings:
If you continue to use the same shell to build the FastCGI library after creating the gzip utility, the Makefile command will fail. You will see an error message regarding a file syntax error. This is because the utility uses the c89 compiler command instead of cc. To eliminate this problem, either open another shell or export CC=cc and continue.

**Step 2. Transfer the fcgi-devkit-2_1_tar.gz file to OS/390**

To copy the file from your workstation to the UNIX HFS file system on OS/390, you can use either binary FTP or PC file transfer.

**Step 3. Uncompress and reload the fcgi-devkit-2_1_tar.gz file**

Run the gzip utility to unpack the fcgi-devkit-2_1.tar.gz file by issuing the following command:

```
gzip -d fcgi-devkit-2_1_tar.gz
```

The tar file produced (fcgi-devkit-2_1.tar) contains binary files and text files in ASCII format. In UNIX System Services, EBCDIC is normally used. Therefore, for some of the files, you use the pax command with a translation option, and a binary transfer for others.

You can use the UNIX shell script (ext.sh) to control the untar process. This script uses the translation option only on text files. You can get a copy of this script on the OS/390 UNIX Web site at URL:


The following command creates subdirectory fcgi-devkit-2.1 in the current working directory for the Developer’s Kit:

```
pax -o from=ISO8859-1,to=IBM-1047 -rf fcgi-devkit-2_1_tar
```

**Step 4. Rebuild the FastCGI Developer’s Kit on OS/390**

1. Execute the following two commands:
   ```
   cd fcgi-devkit-2.1
   ./configure
   ```

2. Edit examples/sample-store.c
   - Change lines 504 and 524
   ```c
   from: cart = Tc1_GetHashValue(cartEntry);
   to: cart = (CartObj *)Tc1_GetHashValue(cartEntry);
   ```
   - Change lines 532, 744, 758, 803, and 825
   ```c
   from: CartObj *cart = Tc1_GetHashValue(cartEntry);
   to: CartObj *cart = (CartObj *)Tc1_GetHashValue(cartEntry);
   ```

3. Edit libfcgi/os_unix.c to change line 54
   ```c
   from:#include <netinet/tcp.h>
   to:#include <xti.h>
   ```

4. Edit libfcgi/fcgi_stdio.c
   - Change line 51
   ```c
   from:#ifdef _WIN32
   to:/*#ifdef _WIN32*/
   ```
   - Change line 53
from: #else

to: /* #else

5. Edit the libfcgi/Makefile to change line 29
   from: $(RANLIB) libfcgi.$(L)
   to: $(RANLIB) libfcgi.$(L)

6. Edit the include/fcgi_stdio.h file to change all occurrences of X'0D' X'40'. If you
   are editing using OEDIT, you can use the command c all X'0D' X'40'.

7. Edit cgi-fcgi/Makefile to change line 30
   from: $(CC) $(CFLAGS) cgi-fcgi.$(O) -o cgi-fcgi $(LIBFCGI) $(LIBS)
   to: $(CC) $(CFLAGS) -o cgi-fcgi cgi-fcgi.$(O) $(LIBFCGI) $(LIBS)

8. Edit examples/Makefile
   Change line 16
   from: CFLAGS = -g -I$(INCLUDEDIR)
   to: CFLAGS = -g -I$(INCLUDEDIR) -DNDEBUG

   Change line 35
   from: $(CC) $(CFLAGS) tiny-cgi.$(O) -o tiny-cgi.cgi
   to: $(CC) $(CFLAGS) -o tiny-cgi tiny-cgi.$(O)

   Change line 38
   from: $(CC) $(CFLAGS) tiny-fcgi.$(O) -o tiny-fcgi $(LIBFCGI) $(LIBS)
   to: $(CC) $(CFLAGS) -o tiny-fcgi tiny-fcgi.$(O) $(LIBFCGI) $(LIBS)

   Change line 41
   from: $(CC) $(CFLAGS) tiny-fcgi2.$(O) -o tiny-fcgi2 $(LIBFCGI) $(LIBS)
   to: $(CC) $(CFLAGS) -o tiny-fcgi2 tiny-fcgi2.$(O) $(LIBFCGI) $(LIBS)

   Change line 44
   from: $(CC) $(CFLAGS) tiny-authorizer.$(O) -o tiny-authorizer $(LIBFCGI) $(LIBS)
   to: $(CC) $(CFLAGS) -o tiny-authorizer tiny-authorizer.$(O) $(LIBFCGI) $(LIBS)

   Change line 47
   from: $(CC) $(CFLAGS) echo.$(O) -o echo $(LIBFCGI) $(LIBS)
   to: $(CC) $(CFLAGS) -o echo echo.$(O) $(LIBFCGI) $(LIBS)

   Change line 50
   from: $(CC) $(CFLAGS) echo2.$(O) -o echo2 $(LIBFCGI) $(LIBS)
   to: $(CC) $(CFLAGS) -o echo2 echo2.$(O) $(LIBFCGI) $(LIBS)

   Change line 53
   from: $(CC) $(CFLAGS) sample-store.$(O) tclHash.$(O) -o sample-store $(LIBFCGI) $(LIBS)
   to: $(CC) $(CFLAGS) -o sample-store sample-store.$(O) tclHash.$(O) $(LIBFCGI) $(LIBS)

   Change line 56
   from: $(CC) $(CFLAGS) log-dump.$(O) -o log-dump $(LIBFCGI) $(LIBS)
   to: $(CC) $(CFLAGS) -o log-dump log-dump.$(O) $(LIBFCGI) $(LIBS)

9. Execute the following two commands:
You now have a full copy of the Developer's Kit on your OS/390 system.

**Hints and tips**
If the compile fails at any stage, you must run the make clean command, fix the error, and run the make command again.

**Forms and data processing**

The CGI process involves three players: the Web browser, the Web server, and the CGI program. The CGI Sample Test Case shown here is an HTML form that uses a CGI program to process the input. Let's assume that the Web browser has already requested and obtained this document.

---

**CGI Sample Test Case**

Fill in the following fields and press APPLY. The values you enter will be read by the CCGIXMP.EXE program and displayed in a simple HTML form which is generated dynamically by the program.

**Checkbox Field**

- [ ] Check to set variable VAR1 to 123
- [x] Check to set variable VAR2 to XyZ

**Radio Button Field**

- [ ] Select to set variable VAR3 to 1
- [ ] Select to set variable VAR3 to 2
- [ ] Select to set variable VAR3 to 3
- [ ] Select to set variable VAR3 to 4

*Figure 2. Sample Form, Page 1*
The following sequence of events shows the roles played by the Web browser, Web server, and CGI program in processing this form:

1. The user fills out the form (clicks buttons or enters information in appropriate fields) and then clicks the Submit button.

2. The Web browser sends the data to the Web server in an encoded format. In our example, this data consists of responses the user entered on this form.

3. Upon receiving the data, the Web server converts the data to a format compliant with the CGI specification for input and sends it to the CGI program.

4. The CGI program decodes the data and processes it per its instruction.

5. The CGI program sends the response back to the Web server in a form that is compliant with the CGI specification for output.

6. The Web server interprets the response and forwards it to the Web browser.

Creating HTML documents that reference CGI programs

CGI programs are referenced from within HTML documents. When you design the layout of your HTML document, keep in mind how a CGI program might affect the look of your document. Developing the CGI program along with the HTML document can help you avoid many design mistakes.
The following HTML source produces our CGI Sample Test Case shown in Figure 2 on page 178. It shows the use of the HTML <form> tag. This tag defines the form within in the document and specifies both the HTTP method by which the data will be sent to the server (POST) and the URL of the CGI program that will process the data (/cgi-bin/cgixmp). The following HTML source also shows how to code various types of input fields that you can include in your HTML forms.

Example

```html
<HTML>
<HEAD>
<TITLE>CGIXMP Test Case</TITLE>
</HEAD>
<BODY>
<H1>CGI Sample Test Case</H1>
Fill in the following fields and press Submit.
The values you enter will be read by the CGIXMP program and displayed in a simple HTML form which is generated dynamically by the program.
</P>
</PRE>
<form method=POST action="/cgi-bin/cgixmp">
<P>
<H3>Checkbox Field</H3>
<pre>
<input type="checkbox" name="var1" value="123">
Check to set variable VAR1 to 123
<input type="checkbox" name="var2" value="XyZ" checked>
Check to set variable VAR2 to XyZ
</PRE>
</H3>
</P>
<H3>Radio Button Field</H3>
<pre>
<input type="radio" name="var3" value="1">
Select to set variable VAR3 to 1
<input type="radio" name="var3" value="2">
Select to set variable VAR3 to 2
<input type="radio" name="var3" value="3" checked>
Select to set variable VAR3 to 3
<input type="radio" name="var3" value="4">
Select to set variable VAR3 to 4
</PRE>
</H3>
</P>
<H3>Single selection List Field</H3>
<pre>
Select a value for variable VAR4 <select size=1 name="var4">
<option>0<option>1<option>2<option>3
<option>4<option>5
</select>
</PRE>
</H3>
</P>
<H3>Text Entry Field</H3>
<pre>
Enter value for variable VAR5 <input type="text" name="var5" size=20 maxlength=256 value="TEST value">
</PRE>
</H3>
</P>
<H3>Multiple selection List Field</H3>
<pre>
Select a value for variable VAR6 <select multiple size=2 name="var6">
<option>Ford<option>Chevrolet<option>Chrysler
</option>
</PRE>
</P>
```
Sending information to the server

When a user fills out an HTML form or enters a phrase in a search field and clicks on the Submit button, the Web browser sends the request to the server in a format described as URL-encoded.

In URL-encoded information:
- The URL starts with the URL of the processing program.
- Attached data, such as name=value information from a form, is appended to the URL and preceded by a question mark.
- Fields are separated by an ampersand (&).
- Space are represented by a plus sign (+).
- Special characters, such as a period or slash, are represented by a percent sign (%) followed by the ASCII hexadecimal equivalent of the symbol.
- Multiple values for a field, such as check boxes, are sent as a string separated by an ampersand (&).

Upon receiving the data, the Web server converts it to a format compliant with the CGI input specification and passes it to the CGI program specified in the HTML document.

Processing the information

Most CGI programs process the data they receive in the following three stages:
- Parsing
- Data manipulation
- Response generation

Parsing

Parsing is the first stage of a CGI program. In this stage, the program takes the data in one or more of the possible formats (environment variables, command-line arguments, or standard input devices), breaks it into components, and decodes the information in the components.
For example, the following could be received using the environment variable QUERY_STRING:

```
NAME=Eugene+T%2E+Fox&ADDR=etfox%40ibm.net&INTEREST=RCO
```

Parsing breaks the fields at the ampersands and decodes the ASCII hexadecimal characters. The results look like this:

```
NAME=Eugene T. Fox
ADDR=etfox@ibm.net
INTEREST=RCO
```

You can use the cgiparse command to automatically parse query strings, read and write CONTENT_LENGTH characters, and count the number of form fields submitted. For a complete description of the cgiparse command, see “cgiparse command” on page 241. Also, see the sample code in “CGI Examples” on page 188.

**Data manipulation**

Data manipulation is the second stage of a CGI program. In this stage, the CGI program takes the parsed data and performs the appropriate action. For example, a CGI program designed to process an application form might:

1. Take the input from the parsing stage
2. Convert abbreviations into more meaningful information
3. Plug the information into an e-mail template
4. Call the sendmail program
5. Send the filled-in template to a specified e-mail address

**Response generation**

Response generation is the final stage of a CGI program. In this stage, the CGI program formulates its response to the Web server, which forwards it to the Web browser. The response begins with **MIME** (Multipurpose Internet Mail Extensions) headers that give meta-information about the returned document. Depending on the type of response, the headers may vary, but the response must contain at least one MIME header (Content-Type) followed by a blank line. The blank line separates the headers from the content of the response.

**Static documents:** If the response is a static document, the program returns the URL of the document using the HTTP Location header, followed by a blank line. For example:

```
Location: http://www.acme.com/products.html
```

Upon receiving this information from the CGI program, the Web server will retrieve the specified document and send a copy of it to the Web browser. Using relative URLs instead of full URLs can improve the response time for clients.

**Dynamic documents:** If the response is a dynamic document, such as a list of hypertext links to documents that meet specified criteria, the program should indicate that the response is an HTML document, using the Content-Type header followed by a blank line, and then include links to the documents in HTML format. With a search, the links might be the URLs of all the documents that met the search criteria.

If the response is an HTML file, the program should indicate that the response is an HTML file, using the Content-Type header followed by a blank line, and then the body of the document.
With a request that results in e-mail, the response might be a message confirming that the e-mail was sent.

**Using nph:** If you do not want the Web server to interpret the response, but just to forward it to the Web browser, the name of your CGI program must begin with **nph-** (no-parse header). A no-parse header program creates output that is a complete HTTP response, including the HTTP return code and status information. It requires no further action, such as interpretation or modification, on the part of the server.

Use the meta refresh HTML tag rather than nph if you want to periodically refresh the page because nph actually degrades performance. Possible reasons for using nph are:

- When returning output over a slow period of time, such as displaying unbuffered results of a slow operation in real time
- if you are implementing a server push of a multipart object such as a dynamically generated animated gif

**Note:** it is the responsibility of the **nph-** script writer to generate the entire HTTP response, including the HTTP status line.

**HTML output:** The response from our CGI Sample Test Case produces the following HTML file that shows the Content-Type header:

```
Content-Type: text/html

<HTML>
<HEAD>
<TITLE>Test HTML Page</TITLE>
</HEAD>
<BODY>
<H1>Variable Information</H1>
<HR>
<P>
<pre>Variable "var1" = 123</pre>
</p>
<p>
<pre>Variable "var2" = XyZ</pre>
</p>
<p>
<pre>Variable "var3" = 3</pre>
</p>
<p>
<pre>Variable "var4" = 0</pre>
</p>
<p>
<pre>Variable "var5" = TEST value</pre>
</p>
<p>
<pre>Variable "var6" = Ford</pre>
</p>
<p>
<pre>Variable "pword" = </pre>
</p>
<p>
<pre>Variable "hidden" = Text not shown on form...</pre>
</p>
</PRE>
</BODY>
</HTML>
```

**Producing headers with cgiutils:** You can use the cgiutils command in your CGI program to produce full or partial sets of MIME headers. You can use optional flags with this command to keep the date from appearing or to display version information, expiration information, and more.

This example shows some of the MIME headers you can produce:

```
HTTP/1.0 200 OK
MIME-Version:1.0
Date: Monday, 25 Oct 95 13:14:15 GMT
Content-Type: text/html
Document-Name: /tcpip/form.html
```
Returning output

When the CGI program is finished, it passes the resulting response to the Web server using standard output (stdout). The Web server interprets the response and sends it to the Web browser.

If the CGI program encounters errors, it may write error information to standard error (stderr). The HTTP Server redirects stderr to the cgi_error log.

For more information on the ErrorLog directive, see “ErrorLog - Name the file where you want to log internal server errors” on page 335.

Protecting your programs

Storing your programs in the cgi-bin subdirectory provides some level of protection because you can ensure that typical users do not have write access to it. In most cases, this is sufficient. However, there are some devious users out there who can figure out how to use your CGI program to “break into” your server.

You can guard against this by understanding the limitations of your code. For instance, if a Perl program encounters an escape character as input, it will abort and dump the user out to the machine’s root directory. If you do not compensate for this, a user could send escape characters in the URL-encoded data as input and gain access to your server. The easiest way to guard against this is by using compiled programs instead of scripts. You can use protection setups and ACL files to protect your programs as well.

For information on access control settings for the Web server, see “Access control - Set up access control for the server” on page 275.

Environment variables

Before writing your CGI program, you need to understand the format in which the server will pass the data. The server receives the URL-encoded information and, depending on the type of request, passes the information to the CGI program using environment variables, command line arguments, or standard input.

In general, the HTML document defines the environment variables that specify how information is passed. For all requests, regardless of type, certain information is passed using the environment variables.

For a description of environment variables you can use in CGI programs, see “Appendix E. Environment variables” on page 393.

Processing standard search (ISINDEX) documents

ISINDEX is an HTML tag that identifies the document as a standard search document and causes the browser to automatically generate an entry field. When information is sent from an ISINDEX document, the server takes the appended data (the information following the ?), breaks it at the pluses (+), and sends the data to the CGI program as command line arguments (argv). For example:

<ISINDEX>
Note: It is possible to write CGI programs that display all environment variables. At times these variables may include sensitive data such as user IDs and passwords for various products. So you must be careful about displaying environment variables in your CGI programs and you must be careful about who has access to them.

Passing SSL environment variables to a CGI program

For a list of SSL-related environment variables you can use in CGI programs, see "Appendix E. Environment variables" on page 395.

Creating CGI source

CGI programs can be written in any language supported by the operating system on which the server is running. The examples in this chapter show CGI programs written in C, REXX, and shell script. To review these examples, see "CGI Examples" on page 188.

The OS/390 operating system has a requirement for translating characters from EBCDIC to ASCII encoding. For an example using this translation, see "CGI program" on page 188.

To access coding samples, go to the HTTP Server Web site Library page at URL: http://www.ibm.com/software/websphere/httpservers/doc52.html.

Parsing Routines

On OS/390, an additional complication with CGI programs is that most clients run on ASCII systems, while OS/390 uses EBCDIC encoding of characters.

Note that:
- for most characters sent from a browser to the HTTP Server, the web server will translate the character from ASCII to EBCDIC.
- for special characters, the browser “escapes” the character prior to sending it to the web server. For example, the ? character is changed to %3F, x’3F’ being the hexadecimal representation in ASCII of ?. The web server receives the character string %3F and converts the string to EBCDIC. Your CGI program is then responsible for translating the string back to a single character. The supplied sample CGI program has a routine TranslateEscapes, which converts the escaped characters back to ASCII. x’3F’ in EBCDIC is not a valid character. The hexadecimal representation of ? in EBCDIC is x’6F’.

Most CGI programs OS/390 Web servers do not take this into account, and therefore will not process special characters entered in an HTML form correctly. The documentation provided with the HTTP Server does describe how to resolve this issue in your own CGI programs, but the point is important so warrants repeating.

String Compare Reminders

If you are trying to compare data passed by the user with some expected value; you will probably use the C function strcmp(). We found that the last value passed by the user can sometimes not match the expected value. It is good practice to use strncmp() and compare for a certain length, rather than using an unbounded compare. As an example, you should use code such as that shown following:
if (!strncmp(html, "yes", 3))

rather than

if (!strcmp(html, "yes"))

when you are comparing with a specific string value.

Another common problem we had when writing gateway programs was in processing logic based on the result of a compare, such as:

```c
if (!strncmp(html, "yes", 3))
{
  ht = ON;
}
else
{
  ht = OFF;
}
```

Note that if two strings compare equally, the result of the string compare is 0, therefore if you want to perform logic such as that just shown, you must use the NOT form (!strncmp).

### Running CGI programs written in Java

On the OS/390 system, the HTTP Server depends on the settings of LIBPATH and CLASSPATH to successfully execute CGI programs written in the Java language. The server refers to the LIBPATH setting to dynamically load the OS/390 Java run-time libraries and it consults the CLASSPATH setting to locate OS/390 Java class libraries and the CGI programs. You can find these settings in /etc/httpd.envvars or its equivalent if you overrode it with ENVAR("_CEE_ENVFILE=...").

You will need to modify these settings for your configuration. For example, if you have installed OS/390 Java at /J1.1.1 and have placed your CGI program, myjavacgi.class, under /usr/lpp/internet/server_root/cgi-bin, then you will need to modify your environment variables file (/etc/httpd.envvars or its equivalent) as follows:

```
LIBPATH=/usr/lpp/internet/bin:/usr/lpp/internet/sbin:/J1.1.1/lib/mvs/native_threads
CLASSPATH=/usr/lpp/internet/server_root/cgi-bin:/J1.1.1/lib/classes.zip
```

**Note:** You must first compile your CGI programs written in Java with javac and then use the resulting class file, optionally including the .class suffix, as the name of the CGI program in your HTML file (for example, /cgi-bin/myjavacgi.class). You must also place the class file in a directory listed in the CLASSPATH environment variable. If you want to use a directory other than the ones currently in CLASSPATH, append this additional directory to the list in CLASSPATH, separated by a colon.

You must then have your Webmaster update the server’s configuration file with an Exec directive that will cause the Java CGI to be executed when a specific URL is received. For more information, see "Exec - Run a CGI program for matching requests" on page 359.

After setting up the server to run your Java CGI, attempt to run it from a browser. If the interface to Java fails, an error page may be returned to the browser with the following:

```
Error loading Java to run cginame (num)
```
cginame is the name of the Java CGI the server was asked to execute. num is a number that indicates the reason for the failure. num can be one of the following:

1  The requested class file did not have the .class extension.
2  The CLASSPATH environment variable was not set.
3  The LIBPATH environment variable was not set.
4  Error attempting to load the libjava.a run-time DLL. This can occur for several reasons:
   • The correct path was not added to the LIBPATH environment variable for Java.
   • The file libjava.a was not found in the path specified on the LIBPATH environment variable.
   • Incorrect permission settings are set for file libjava.a or any directory it is contained in.
5  The entry point java_main could not be located.
6  The allocation of temporary storage failed.
7  The entry point iconv_init could not be located.
8  An error returned from the iconv_init() function.
9  The java_main was improperly returned to the interface routine.

An additional line of text is included in 4, 5, 7 and 8 that indicates the current values of errno and errno2.

Response generation

You also need to consider character translation in your CGI programs. Most browsers run on operating systems that use ASCII encoding of characters, while the OS/390 system uses EBCDIC encoding. Therefore, on the OS/390 system, you can also include a Content-Encoding header in your CGI response to specify whether or not the server should translate the content of the response before sending it back to the browser. The recognized values for the Content-Encoding header are:

7bit  Data is all represented as short (less than 1000 characters) lines of US-ASCII data. Source code or plain text files usually fall into this category. Exceptions would be files containing line-drawing characters or accented characters.
8bit  Data is represented as short lines, but may contain characters with the high bit set (for example, line-drawing characters or accented characters). PostScript files and text files from European sites usually fall into this category.
binary This encoding can be used for all data types. Data may contain not only non-ASCII characters, but also long (greater than 1000 characters) lines. Almost every file of type image/*, audio/*, and video/* falls into this category, as do binary data files of type application/*.
ebcdic Data is represented as lines of EBCDIC text (IBM-1047) and data is translated to ASCII text (ISO8859-1) when served.
other Any other value of encoding receives the same treatment as binary.
If the Content-Encoding header contains the value ebcDIC, the server will translate the content to ASCII characters. If the Content-Encoding header contains any other value, the server will not translate the content.

If the Content-Encoding header contains the value 7bit, 8bit, binary or ebcDIC, the server will not forward the Content-Encoding header to the browser.

If the Content-Encoding header is missing, the server will use the information in the Content-Type header to determine if it should translate the content. In this case, if the Content-Type header specifies a value starting with text/, the server will translate the contents to ASCII characters. With any other value, it will not translate.

---

**CGI Examples**

This section includes the following examples:
- "CGI C program"
- "CGI REXX program" on page 193
- "CGI shell script" on page 195

**CGI C program**

The following example is a version of the C program, cgixmp, used in our CGI Sample Test Case shown in the example on page 180. It includes modifications to the TranslateEscapes routine which are required by the OS/390 system for the conversion of escaped characters.

Most Web browsers run on ASCII based operating systems, while the OS/390 system uses EBCDIC encoding of characters. When a browser sends data to the server running on the OS/390 system, the server translates the characters from ASCII to EBCDIC encoding. But browsers "escape" special characters prior to sending them to the Web server and this can pose problems.

For example, when the browser escapes the ? character, it changes it to %3F before sending it (x'3F' is the hexadecimal representation in ASCII of ?). The OS/390 server receives the character string of %3F and has to convert the string to EBCDIC. The CGI program is then responsible for converting the EBCDIC string back to a single character. In this case, x'3F' is not a valid EBCDIC character; the valid representation of ? in EBCDIC is x'6F'. The conversion is applied in following example.

**Example (cgixmp)**

```c
/* Includes */
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define TRUE 1
/* Definition of structure used in linked list of variables */
typedef struct _argument
{
    char *VariableName;
    char *Value;
    struct _argument *pNext;
} PARMLIST, *PPARMLIST;

/* Function definitions */
intErrMsg (char *msg);
PPARMLIST ReadArguments(int InputLength);
static void PlusesToSpaces(char *Str);
```

---

188  HTTP Server Planning, Installing, and Using
static int HexVal(char c);
static void TranslateEscapes(char *Str);

/******************************************************************************/
/**/
/* Function : main */
/* */
/* */
/* Description : This is a CGI program which takes the output of a form */
/* submitted with a method of POST, and displays a list */
/* of the variable names and values. */
/* */
/******************************************************************************/
main(int argc, char *argv[])
{
    char *requestMethod;
    char *contentLength;
    int argLength;
    PPARMLIST pParm = NULL;
    PPARMLIST pHead = NULL;
/* This CGI program must be called with a method of POST */
    requestMethod = getenv("REQUEST_METHOD");
    if ((requestMethod == NULL) ||
        (strcmp(requestMethod, "POST")))
    {
       ErrMsg("REQUEST_METHOD environment variable not properly set to POST\n");
    }
    else
    {
        /* Get the length of the arguments passed in to this program */
        contentLength = getenv("CONTENT_LENGTH");
        if (contentLength == NULL)
        {
           ErrMsg("CONTENT_LENGTH environment variable not set\n");
        }
        else
        {
            /* Begin output of HTML to display results of CGI program */
            printf("Content-type: text/html\n\n");
            printf("<HTML>\n");
            printf("<HEAD>\n");
            printf("<TITLE>Sample HTML Page</TITLE>\n");
            printf("</HEAD>\n");
            printf("<BODY>\n");
            printf("<H1>Variable Information</H1>\n");
            printf("<HR>\n");
            printf("<P>\n");
            /* Read the arguments passed in to this program and place them in */
            /* a singly linked list - one link per variable */
            argLength = atoi(contentLength);
            pHead = ReadArguments(argLength);
            pParm = pHead;
            /* Output the list of variable names and values */
            while (pParm)
            {
                printf("<PRE>Variable "");
                printf("%s" = %s</PRE><P>",
                        pParm->VariableName,
                        pParm->Value);
                pParm = pParm->pNext;
            }
            /* Output the remainder of the HTML used to display the results */
            printf("<P>\n");
            printf("<HR>\n");
            printf("</BODY>\n");
        }
    }
}

Chapter 17. Writing Common Gateway Interface programs  189
printf("</HTML>
");
}
}

/******************************************************************************/
/* */
/* Function: ReadArguments */
/* */
/* Description: Read the arguments from stdin that are supplied */
/* to a CGI program when the method is POST. */
/* Breaks up the input into */
/* (Variable, Value) pairs. */
/* Handles translating of all the special characters */
/* that HTTP puts into the strings. */
/**/
******************************************************************************/

PPARMLIST ReadArguments(int InputLength)
{
    PPARMLIST pCur = NULL;
    PPARMLIST pHead = NULL;
    PPARMLIST pPrev = NULL;
    char *Input;
    char *pToken;

    if (InputLength < 1)
    {
        return(NULL);
    }
    /* Allocate a buffer for the input */
    Input = malloc(InputLength + 1);
    if (Input == NULL)
    {
        return(NULL);
    }
    /* Read the input */
    gets(Input);
    /* Variables are separated by the "&" character */
    pToken = strtok(Input, "&");

    while (pToken)
    {
        /* Create and fill in linked list of variable information */
        pCur = malloc(sizeof(PARMLIST));
        pCur->VariableName = pToken;
        pToken = strchr(pToken, '=');
        if (pToken)
        {
            *pToken = '\0';
            pCur->Value = ++pToken;
            PlusToSpaces( pToken );
            TranslateEscapes( pToken );
        }
        else
        {
            pCur->Value = NULL;
        }
        if (pPrev)
        {
            pPrev->pNext = pCur;
        }
        if (!pHead)
        {
            pHead = pCur;
        }
        if (InputLength < 1)
        {
            return(NULL);
        }
    }
    return (pCur);
}
{  
    pHead = pCur;
}
pPrev = pCur;
pToken = strtok(NULL, ",\&");
}

if (pHead)
{
    pPrev->pNext = NULL;
}
return(pHead);

/******************************************************************************/
/*
/* Function : PlusesToSpaces (STATIC)
/*
/* Description : This one's easy. It just translates any '+' characters found into ' ' characters.
/*
 /******************************************************************************/
static void PlusesToSpaces(char *Str)
{
    if (Str != NULL)
    {
        while (*Str != '\0')
        {
            if (*Str == '+')
            {
                *Str = ' ';
            }
            ++Str;
        }
    }
}

static int HexVal(char c)
{
    int rc;

    switch (c)
    {
    case '1':
        rc = 1;
        break;
    case '2':
        rc = 2;
        break;
    case '3':
        rc = 3;
        break;
    case '4':
        rc = 4;
        break;
    case '5':
        rc = 5;
        break;
    case '6':
        rc = 6;
        break;
    }
case '7':
    r c = 7;
    break;

case '8':
    r c = 8;
    break;

case '9':
    r c = 9;
    break;

case 'A':
case 'a':
    rc = 10;
    break;

case 'B':
case 'b':
    rc = 11;
    break;

case 'C':
case 'c':
    rc = 12;
    break;

case 'D':
case 'd':
    rc = 13;
    break;

case 'E':
case 'e':
    rc = 14;
    break;

case 'F':
case 'f':
    rc = 15;
    break;

default:
    r c = 0;
    break;
}

return(rc);
}

/******************************************************************************/
/* */
/* Function : TranslateEscapes (STATIC) */
/* */
/* Description : Translate the escape sequences induced by HTTP. The sequences consist of %xx, where xx is a hex number. We replace the % character with the actual character (i.e., the one whose ASCII value is xx), and then shift over the rest of the string to remove the xx. This is done in-place. */
/* */
/**/
/******************************************************************************/
static void TranslateEscapes(char *Str)
{
    char *NextEscape;

    }
char ConvertValue[2];
int AsciiValue;

NextEscape = strchr(Str, '%');
while (NextEscape != NULL) {
    AsciiValue = (16 * HexVal(NextEscape[1]) + HexVal(NextEscape[2]));
    ConvertValue[0] = (char) AsciiValue;
    memcpy(ConvertValue, 1);
    strcpy(NextEscape, ConvertValue, 1);
    memmove(&NextEscape[1], &NextEscape[3], strlen(&NextEscape[3]) + 1);
    NextEscape = strchr(&NextEscape[1], '%');
}

/******************************************************************************/
/* This function will output a message if an error occurs when attempting */
/* to display a HTML page. */
/******************************************************************************/
int ErrMsg(char *msg) {
    printf("Content-type: text/html\n\n");
    printf("<HTML>\n");
    printf("<HEAD>\n");
    printf("<TITLE>Error</TITLE>\n\n");
    printf("<BODY>\n\n");
    printf("<H1>Error</H1>\n\n");
    printf("<HR>\n\n");
    printf("<P>\n\n");
    printf("An error occurred in the CGI program.\n\n");
    printf("The specific error message is shown below:\n\n");
    printf("<PRE>%s</PRE>\n\n", msg);
    printf("<P>\n\n");
    printf("<HR>\n\n");
    printf("</BODY>\n\n");
    printf("</HTML>\n\n");
    return(TRUE);
}

CGI REXX program
The next example is written in REXX. It could process the form in the example shown on page 180 by calling the REXX EXEC CGIXMP.CMD. To do this, simply replace the line <form method=POST action="cgi-bin/cgixmp"> in the HTML source with <form method=GET action="cgi-bin/CGIXMP.CMD">.

In this example, cgiutils and cgiparse commands are used to generate MIME type headers and to parse the HTTP request from the form.

Example
/* REXX */
cgiparse -status 200 -ct text/html
say 'This is an example of a CGI script which uses the <CODE>'
say 'cgiutils'</CODE> and <CODE>cgiparse</CODE>'
say 'functions to create a valid HTTP header and parse forms data,'
say 'respectively.'
say 'The <CODE>cgiutils</CODE> program is used to create'
say 'a set of HTTP'
headers for the CGI script output.'

The <CODE>cgiparse</CODE> program is used to extract'
data submitted as HTTP GET or POST requests from'
forms The program parses variables from the CGI QUERY_STRING'
environment variable.'
The <CODE>cgiparse</CODE> program is useful for'
extracting information such as the value of'
a specific form field or the number of unique fields submitted'
from a form.'
This information can then be processed by CGI scripts in'
performing their intended functions.'
Some examples of how the <CODE>cgiparse</CODE> program'
can be used are shown below'
<P>
<P>
The following is an example of the output that is generated by'
using the -form option of the'
<CODE>cgiparse</CODE> command. The -form option'
will parse the QUERY_STRING as a form request'
and output the fields as a string of variables.'
<P>
The format of the <CODE>cgiparse</CODE> command is'
<CODE> cgiparse -form </CODE>.
<P>
<CODE>
'cgiparse -form'
</CODE>
<P>
Querying values using <CODE>cgiparse</CODE>
The following is an example of using the <CODE>cgiparse</CODE>
command to query the values of specific'
fields parsed from the QUERY_STRING.'
<P>
The format of the <CODE>cgiparse</CODE> command is'
<CODE>cgiparse -value <CITE>fieldname</CITE></CODE>
<P>
Value for variable 1 = <CODE>
'cgiparse -value var1'
</CODE>
<P>
Value for variable 2 = <CODE>
'cgiparse -value var2'
</CODE>
<P>
Value for variable 3 = <CODE>
'cgiparse -value var3'
</CODE>
<P>
Value for variable 4 = <CODE>
'cgiparse -value var4'
</CODE>
<P>
Value for variable 5 = <CODE>
'cgiparse -value var5'
</CODE>
<P>
Value for variable 6 = <CODE>
'cgiparse -value var6'
</CODE>
<P>
Determining the number of fields in QUERY_STRING using'
The <CODE>cgiparse</CODE> function can also be used to'

194  HTTP Server Planning, Installing, and Using
CGI shell script

The next example is written in the shell scripting language. It can process the form shown in the example on page 180 by calling the shell script command CGIXMPSH. To do this, simply replace the line `<form method=POST action="cgi-bin/cgixmp">` in the HTML source with `<form method=GET action="cgi-bin/CGIXMP.SH">`.

In this example, cgiutils and cgiparse commands are used to generate MIME type headers and to parse the HTTP request from the form.

**Example**

```bash
#!/bin/sh
# This simple shell script will take the arguments passed in,
# and print them out
cgiutils -status 200 -ct text/html
echo '<HTML>';
echo '<BODY>';
echo '<H1>Processing forms with CGI scripts</H1>';
echo '<P> This is an example of a CGI script which uses the <CODE>cgiutils</CODE> and <CODE>cgiparse</CODE> functions to create a valid HTTP header and parse forms data, respectively.'
echo '<HR>'
echo 'The <CODE>cgiutils</CODE> program is used to create a set of HTTP' 
echo 'headers for the CGI script output.'
echo '<P>'
echo 'The <CODE>cgiparse</CODE> program is used to extract' 
echo 'data submitted as HTTP GET or POST requests from' 
echo 'forms The program parses variables from the CGI QUERY_STRING' 
echo 'environment variable.'
echo 'The <CODE>cgiparse</CODE> program is useful for' 
echo 'extracting information such as the value of' 
echo 'a specific form field or the number of unique fields submitted' 
echo 'from a form.';
echo 'This information can then be processed by CGI scripts in' 
echo 'performing their intended functions.'
echo '<P>'
echo 'Some examples of how the <CODE>cgiparse</CODE> program' 
echo 'can be used are shown below'
echo '<HR>'
echo '<H2>Parsing forms requests using <CODE>cgiparse</CODE>'
echo '<\CODE>echo '</H2>';
echo '
```
The format of the `<CODE>cgiparse</CODE>` command is:

```bash
cgiparse -form
```

Querying values using `<CODE>cgiparse</CODE>`:

```bash
cgiparse -value <CITE>fieldname</CITE>
```

Value for variable 1 = `cgiparse -value var1`
Value for variable 2 = `cgiparse -value var2`
Value for variable 3 = `cgiparse -value var3`
Value for variable 4 = `cgiparse -value var4`
Value for variable 5 = `cgiparse -value var5`
Value for variable 6 = `cgiparse -value var6`

Determining the number of fields in QUERY_STRING using `

```bash
cgiparse -form -count
```

Number of unique fields = `cgiparse -form -count`
Chapter 18. Writing GWAPI programs

Overview of the GWAPI

The Go Webserver Application Programming Interface (GWAPI) is an interface to the HTTP Server that allows you to extend the server’s base functions. You can write extensions to do customized processing, such as:

- Enhance the basic authentication or replace it with a site-specific process
- Add error handling routines to track problems or alert for serious conditions
- Detect and track information that comes in from the requesting client, such as server referrals and user agent code

The GWAPI compares favorably to other similar APIs (such as Netscape’s NSAPI and Microsoft’s ISAPI) for these reasons:

- **Efficiency**
  - Designed specifically for the threaded (rather than forked) processing used by the HTTP Server.

- **Flexibility**
  - Contains rich and compatible functions that can be used to clone other APIs.
  - Is platform independent and language neutral. It runs on all the Web server platforms, and applications can be written in any of the programming languages supported by these platforms.

- **Ease of use**
  - Passes simple data types by reference, not value (for example, long * or char *)
  - Has a fixed number of parameters for each function
  - Has no return values from functions but does have a return code parameter
GWAPI programs

- Includes bindings for the C language
- Does not impact allocated memory (the GWAPI does not free the user's memory and the user does not free the GWAPI's memory)

General procedure for writing GWAPI programs

Before you start writing your GWAPI programs, you need to understand how the HTTP Server works. The server performs a sequence of steps for each client request that it processes. Your program can execute and make function calls at any of these steps. Decide where in the basic server request process you want to add customized functions. For example, do you want to do something after a client request is read but before performing any other processing? Or, maybe you want to perform special routines during authentication and then again after the requested file is sent.

Logically, your program will have two parts: application functions (written by you) to handle one of the steps and predefined functions (provided with the GWAPI) that you call to manipulate the request. You can instruct the server to call the application functions in your program at the appropriate processing steps by using the GWAPI directives in your server configuration file.

Included in this document are:
- Guidelines to follow (page 198)
- An explanation of the basic server request process (page 199)
- Prototypes for the server's application functions you can write that are valid for each step, and their return codes (page 202)
- Definitions of the server predefined functions and macros you can call from within your application functions, and their return codes (page 206)
- GWAPI configuration directives (page 212)

Using these components, you can follow a basic procedure to write your own GWAPI programs.

Guidelines for writing GWAPI programs

1. Write your program, following the syntax and guidelines we provide for the server's application functions. Give each of your application functions a unique function name and call the server predefined functions as needed.

2. Be sure to include HTAPI.h and to use the HTTPD_LINKAGE macro in your function definitions to avoid abending the server. This macro ensures that all the functions use the same calling conventions.

3. The server runs in a multi-threaded environment; therefore, your application functions must be threadsafe. If your application is re-entrant, performance will not decrease.

4. You can write multi-threaded GWAPI applications for the Web server. This support enables GWAPI applications to create more threads if needed. Note that the GWAPI application is responsible for creating and managing new threads. If the application needs more threads, a thread create is required. The maximum number of threads that can be created is determined by the system limit settings for MAXTHREADS and MAXTHREADTASKS in the SYS1.PARMLIB(BPXPRMxx) member.

Important: The Web server requires that you design your multi-threaded GWAPI applications to use only the base Web server thread for communications, for example, calling HTTPD_write. The base thread is the
5. **Keep the actions in your applications to a thread scope.** Do not perform any actions at a process scope, such as exit, change user ID, registering signal handlers, etc.

6. Eliminate global variables or protect them with a mutual exclusion semaphore.

7. Do not forget to set the Content-Type header if you are using HTTPD_write() to send data back to the client.

8. You will also need to take into consideration the Content-Encoding header when you use HTTPD_write() to send data back to the client. This is especially important for the OS/390 platform where the native code page is EBCDIC.

   For example, if the data is stored on OS/390 in the EBCDIC code page, you must use HTTPD_set() to set the CGI environment variable CONTENT_ENCODING to the appropriate code page (in this case, to "ebcdic") before you use HTTPD_write() to send the data out to the client. If you fail to do this, the client will display unreadable content.

9. Always check your return codes and provide conditional processing where necessary.

10. Compile and link your program, referring to the documentation for your compiler to build a DLL or .so file, as required for your operating system. Use the following compile and link commands as a guideline: **OS/390**

    Compile:
    ```
c89 -W"c,expo" -W"c,dll" -l/usr/lpp/internet/samples/API foo.c
    ```

    Link:
    ```
c89 -W"l,dll" -o foo.so foo.o

    /usr/lpp/internet/samples/API/libhttpdapi.x
    ```

    **Notes:**
    a. `-W"c,expo"` causes all symbols from your DLL to be exported. You can include #pragma export directives in your source code to do selective exports. You should use the OS/390 C/C++ compiler. Refer to the IBM C/C++ for OS/390 Programming Guide for more information on building and using DLLs.

    b. The OS/390 C compiler **must** be used on OS/390 since it supports the expo and dll options. The AD/Cycle C compiler might not support these options.

11. Add GWAPI directives to your configuration file so that you can associate your program’s application functions with the appropriate step. There is a separate directive for each server request processing step. Stop and restart your server to make the new directives take effect.

12. Test your program rigorously. Because the server is a threaded server, you should apply more rigorous testing than you would for a forking server. Because the server calls your program directly and they both run in the same process space, errors in your program can crash the server.

### Server request process

The basic server request process can be broken up into a number of steps, based on the type of processing the server is performing during that phase. Each step includes a juncture at which a specified part of your program can execute. By adding GWAPI directives to your configuration file, you indicate which of your
GWAPI programs

application functions you want called during a particular step. You can call several application functions during a request process step by including more than one of the GWAPI directives for that step.

Your compiled program resides in a DLL or .so file, depending on your operating system. As the server proceeds through its request process steps, it calls the application functions associated with each step, until one of the functions indicates it has handled the request. If you have more than one of your application functions indicated for a particular step, they are called in the order in which they appear in the configuration file.

If the request is not handled by an application function (either you did not include a GWAPI directive or your application function for that step returned HTTP_NOACTION), the server will perform its default action for that step.

Note: This is true for all steps except the Service step; the Service step does not have a default action.

Process steps

The following list indicates the purpose of each step and defines the processing order.

Server Init

Read request

Name Translation

Authorize

Object Type

Service

Log

Error

Server Term

Authenticate

error?

CGI/FastCGI
static file (cache or file system)
plug-in (servlet)
PICSDBLookup

.... pre-exit

.... post-exit

HTTP Server Planning, Installing, and Using
Server Initialization
Performs initialization functions before any client requests are read.

PreExit
Performs processing after a request is read but before anything else is done.

If this step returns an indication that the request was processed (HTTP_OK), skips the other steps in the request process and performs only the Data Filter, Log, and PostExit steps.

Authentication
Decodes, verifies, and stores security tokens.

See the "Authentication and authorization" on page 214 for more information.

Name Translation
Translates the virtual path (from URL) to the physical path.

Authorization
Uses stored security tokens to check the physical path (protections, acls) and generates the WWW-Authenticate headers required for basic authentication. If you write your own application function to replace this step, you must generate these headers yourself.

See "Authentication and authorization" on page 214 for more information.

Object Type
Locates the filesystem object indicated by the path.

Service
Satisfies the request (send the file, run the CGI, etc.)

PICSDBLookup
Looks for the PICS labels for the specified URL.

Data Filter
Gives write access to the outgoing data stream.

Log
Allows transaction logging.

Error
Allows customized responses to error conditions.

PostExit
Allows "cleanup" of resources allocated for request processing.

Server Termination
Allows "cleanup" processing when an orderly shutdown or restart occurs.

Application functions
Follow the syntax presented in the "Application function prototypes" on page 203 to write your own program functions for the defined request processing steps.

Each of your functions must fill in the return code parameter with a value that indicates what action was taken.

- HTTP_NOACTION (value of 0) means no action was taken.
- Otherwise one of the valid HTTP return codes is expected, indicating the application function handled the step. As a result, no other application functions will be called to handle that step of this request.
Application function prototypes

The function prototypes for each request step show the format to use and explain the type of processing they can perform. Note that the function names are not predefined. You must give your functions unique names and can choose your own naming conventions. For ease of association, we use names that relate to the server’s request processing steps.

Server Initialization

```c
void HTTPD_LINKAGE ServerInit(
    unsigned char *handle, unsigned long *major_version,
    unsigned long *minor_version, long *return_code);
```

Called once when your module is loaded during server initialization, it is your opportunity to perform initialization before any requests have been accepted. Although all server initialization functions are called, error return codes from this step will cause the server to ignore all other functions configured in this program.

PreExit

```c
void HTTPD_LINKAGE PreExit(
    unsigned char *handle, long *return_code);
```

Called after the request has been read, but before any processing has occurred.

All server predefined functions are valid during this step.

Authentication

```c
void HTTPD_LINKAGE Authentication(
    unsigned char *handle, long *return_code);
```

Allows user verification of the security tokens. This step is performed based on the authentication scheme.

Only HTTPD_extract(), HTTPD_set(), and HTTPDAuthenticate() are valid during this step.

On OS/390 ONLY, HTTPD_local_security is valid during this step.

Name Translation

```c
void HTTPD_LINKAGE NameTrans(
    unsigned char *handle, long *return_code);
```

Provides a mechanism for mapping URLs to objects.

Only HTTPD_extract(), HTTPD_set(), and HTTPDAuthenticate() are valid during this step.

Authorization

```c
void HTTPD_LINKAGE Authorization(
    unsigned char *handle, long *return_code);
```
Verifies that the identified object may be returned to the client. If you are doing basic authentication, you must generate the required WWW-Authenticate headers.

Only HTTPD_extract() and HTTPD_set() are valid functions during this step.

**Object Type**

```c
void
HTTPD_LINKAGE ObjType(
    unsigned char *handle, long *return_code);
```

Checks to see if the object exists and performs object typing.

Only HTTPD_extract() and HTTPD_set() are valid during this step.

**Service**

```c
void
HTTPD_LINKAGE Service(
    unsigned char *handle, long *return_code);
```

Satisfies the request, if not satisfied in the PreExit.

All server predefined functions are valid during this step. For information on configuring your Service function to execute based on the HTTP method rather than the URL, see the description of the Enable directive in "Enable - Enable HTTP methods" on page 349.

**PICSDBLookup**

```c
void
HTTPD_LINKAGE PICSDBLookup(
    unsigned char *handle, long *return_code);
```

Allows you to either dynamically create a PICS label for the requested document or to search for a PICS label in an alternative file or database. Your application function can evaluate the PICS_SERVICENAME, PICS_SITENAME, and PICS_PATHNAME variables to determine which document was requested.

If your application function finds or creates a label, it must call the HTTPD_supply_label() predefined function to make the label available to the HTTP Server and return a code of 200. Only HTTPD-extract(), HTTPD_set() and HTTPD_supply_label() are valid during this step.

**Note:** You can process this step more than once for each request. If a request asks for labels for multiple documents or from multiple rating services, the server will call the application function once for each label that is requested.

**Data Filter**

Filters data as a "stream class", which means that each of its functions acts like a segment of pipe down which data flows. For this step, you must implement three application functions:

```c
void
HTTPD_LINKAGE open(
    unsigned char *handle, long *return_code);
```
GWAPI programs

Performs any initialization (such as buffer allocation) required to process the data for this stream. An error return code will cause this filter to abort (the write and close functions will not be called).

```c
void HTTPD_LINKAGE write(
    unsigned char *handle, unsigned char *data,
    unsigned long *length, long *return_code);
```

Or, on OS390

```c
void HTTPD_LINKAGE write(
    unsigned char *handle, unsigned char *data,
    unsigned long *length, long *return_code,
    HTCodePage_t *codepage);
```

Processes the data and calls the server's write function with the new or changed data. The application must not attempt to free the buffer passed to it nor expect the server to free the buffer it receives.

The implementation of the HTCodePage_t parameter is available for OS/390 only in release 4.6.1 and later. GWAPI users cannot use the DataFilter's compiled using this implementation with previous versions of the server. The earlier versions will not support the additional codepage parameter and unexpected behavior will result.

```c
void HTTPD_LINKAGE HTTPD_writeCP(
    unsigned char *handle, unsigned char *value, unsigned long *value_length,
    HTCodePage_t *code_page, long *return_code);
```

allows the GWAPI user to specify the body code page. To view the definition for the HTCodePage_t data type see "HTCodePage_t, the other data type" on page 226.

**Note:** The code page specification is available for OS/390 only.

```c
void HTTPD_LINKAGE close(
    unsigned char *handle, long *return_code);
```

Performs any cleanup (such as flushing and freeing the buffer) required to complete processing the data for this stream.

**Note:** Be aware that undesirable effect may occur if data filtering applications are not properly selective on their filtering of data streams. CGI's may not work if filtered incorrectly, GIF files may not display, and other binary streams may not work as expected.

**Log**

```c
void HTTPD_LINKAGE Log(
    unsigned char *handle, long *return_code);
```

Called after each request has been processed and the communication to the client has been closed, regardless of the success or failure of the request processing. Only HTTPD_extract() and HTTPD_set() are valid during this step.

**Error**
void HTTPD_LINKAGE Error(
    unsigned char *handle, long *return_code);

Called only when an error is encountered and provides an opportunity to customize the response.

PostExit

void HTTPD_LINKAGE PostExit(
    unsigned char *handle, long *return_code);

Called, regardless of the success or failure of the request, so that you can clean up any resources allocated by your application to process the request.

Server Termination

void HTTPD_LINKAGE ServerTerm(
    unsigned char *handle, long *return_code);

This function is processed when an orderly shutdown or restart of the server occurs. It allows you to clean up resources allocated during the Server Initialization step. Do not call any HTTP_* functions in this step (the results are unpredictable). If you have more than one GWAPI directive in your configuration file for Server Termination, they will all be called.

HTTP return codes and values
These return codes follow the HTTP specification published by the World Wide Web Consortium at URL: [http://www.w3.org/Protocols/]

Your application functions should return one of the following values:

<table>
<thead>
<tr>
<th>Value</th>
<th>Return code</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>HTTP_NOACTION</td>
</tr>
<tr>
<td>100</td>
<td>HTTP_CONTINUE</td>
</tr>
<tr>
<td>101</td>
<td>HTTP_SWITCHING_PROTOCOLS</td>
</tr>
<tr>
<td>200</td>
<td>HTTP_OK</td>
</tr>
<tr>
<td>201</td>
<td>HTTP_CREATED</td>
</tr>
<tr>
<td>202</td>
<td>HTTP_ACCEPTED</td>
</tr>
<tr>
<td>203</td>
<td>HTTP_NON_AUTHORITATIVE</td>
</tr>
<tr>
<td>204</td>
<td>HTTP_NO_CONTENT</td>
</tr>
<tr>
<td>205</td>
<td>HTTP_RESET_CONTENT</td>
</tr>
<tr>
<td>206</td>
<td>HTTP_PARTIAL_CONTENT</td>
</tr>
<tr>
<td>300</td>
<td>HTTP_MULTIPLE_CHOICES</td>
</tr>
<tr>
<td>301</td>
<td>HTTP_MOVED_PERMANENTLY</td>
</tr>
<tr>
<td>302</td>
<td>HTTP_MOVED_TEMPORARILY</td>
</tr>
<tr>
<td>303</td>
<td>HTTPSEE_OTHER</td>
</tr>
<tr>
<td>304</td>
<td>HTTP_NOT_MODIFIED</td>
</tr>
<tr>
<td>305</td>
<td>HTTP_USE_PROXY</td>
</tr>
<tr>
<td>400</td>
<td>HTTP_BAD_REQUEST</td>
</tr>
<tr>
<td>401</td>
<td>HTTP_UNAUTHORIZED</td>
</tr>
</tbody>
</table>
GWAPI programs

<table>
<thead>
<tr>
<th>Value</th>
<th>Return code</th>
</tr>
</thead>
<tbody>
<tr>
<td>403</td>
<td>HTTP_FORBIDDEN</td>
</tr>
<tr>
<td>404</td>
<td>HTTP_NOT_FOUND</td>
</tr>
<tr>
<td>405</td>
<td>HTTP_METHOD_NOT_ALLOWED</td>
</tr>
<tr>
<td>406</td>
<td>HTTP_NOT_ACCEPTABLE</td>
</tr>
<tr>
<td>407</td>
<td>HTTP_PROXY_UNAUTHORIZED</td>
</tr>
<tr>
<td>408</td>
<td>HTTP_REQUEST_TIMEOUT</td>
</tr>
<tr>
<td>409</td>
<td>HTTP_CONFLICT</td>
</tr>
<tr>
<td>410</td>
<td>HTTP_GONE</td>
</tr>
<tr>
<td>411</td>
<td>HTTP_LENGTH_REQUIRED</td>
</tr>
<tr>
<td>412</td>
<td>HTTP_PRECONDITION_FAILED</td>
</tr>
<tr>
<td>413</td>
<td>HTTP_ENTITY_TOO_LARGE</td>
</tr>
<tr>
<td>414</td>
<td>HTTP_URL_TOO_LONG</td>
</tr>
<tr>
<td>415</td>
<td>HTTP_BAD_MEDIA_TYPE</td>
</tr>
<tr>
<td>500</td>
<td>HTTP_SERVER_ERROR</td>
</tr>
<tr>
<td>501</td>
<td>HTTP_NOT_IMPLEMENTED</td>
</tr>
<tr>
<td>502</td>
<td>HTTP_BAD_GATEWAY</td>
</tr>
<tr>
<td>503</td>
<td>HTTP_SERVICE_UNAVAILABLE</td>
</tr>
<tr>
<td>504</td>
<td>HTTP_GATEWAY_TIMEOUT</td>
</tr>
<tr>
<td>505</td>
<td>HTTP_BAD_VERSION</td>
</tr>
</tbody>
</table>

Predefined functions and macros

You can call the server’s predefined functions and macros from your own application functions. You must use their predefined names and follow the format we describe below. Note that the parameter descriptions use the letter “i” to indicate input, “o” to indicate output, and “i/o” to indicate that a parameter is both input and output.

Each of these functions will return one of the HTTPD return codes, depending on the success of the request. Use the handle as the first parameter when calling these functions; Otherwise, the function will return a HTTPD_PARAMETER_CHECK error code. NULL is not accepted as a valid handle.

HTTPD_authenticate()  
Authenticates a userid/password. Valid only in PreExit, Authenticate, and Authorization steps.

```c
void HTTPD_authenticate(  
    unsigned char *handle, /* i; handle */  
    long *return_code); /* o; return code */
```

This function returns HTTPD_AUTHENTICATE_FAILED if the authenticate fails.

HTTPD_attributes()  
Extracts the attributes of a file. Valid in all steps.

```c
void HTTPD_attributes(  
    unsigned char *handle, /* i; handle */  
    unsigned char *name, /* i; name of the file */
```
HTTPD_extract()

Extracts the value of a variable associated with this request. The valid
variables you can use for the name parameter are the same as those used in
the CGI. See “Environment variables” on page 217 for more information.
This function is valid in all steps, though not all variables will be.

HTTPD_reverse_translate()

translates a file system path to a URL. Valid in all steps.

HTTPD_translate()

translates a URL to a file system path. Valid in all steps.

HTTPD_log_access()

writes a string to the server’s access log. When writing a message to the
access log, it is not necessary to use the escape character when printing the
percent (%) character.

Chapter 18. Writing GWAPI programs 207
GWAPI programs

```c
unsigned char *value, /* i; data to write */
unsigned long *value_length, /* i; length of the data */
long *return_code); /* o; return code */

HTTPD_set()
Sets the value of a variable associated with this request. The valid variables you can use for the name parameter are the same as those used in the CGI. See "Environment variables" on page 217 for more information.

Note that you can also create variables with this function. If any variables you create are prefixed by "HTTP_", they will be sent as headers in the response, without the "HTTP_" prefix. For example, if you want to see a Location header, use HTTPD_set() with the variable name HTTP_LOCATION.

On OS/390, you can use this function to send Content-Encoding headers in the response. See "Response generation" on page 187 in the CGI chapter for more information on the Content-Encoding header and OS/390 code page translation.

This function is valid in all steps, though not all variables will be.

```void
void HTTPD_set{
    unsigned char *handle, /* i; handle */
    unsigned char *name, /* i; name of the value to set */
    unsigned long *name_length, /* i; length of the name */
    unsigned char *value, /* i; buffer containing the value */
    unsigned long *value_length, /* i; length of value */
    long *return_code); /* o; return code */
```

HTTPD_supply_label
Provides the HTTP Server with a PICS label that was generated or retrieved during the PICSDBLookup step.

Valid only during the PICSDBLookup step.

```void
void HTTPD_supply_label{
    unsigned char *handle, /* i; handle */
    unsigned char *value, /* i; data to send */
    unsigned long *value_length, /* i; length of the data */
    long *return_code); /* o; return code */
```

HTTPD_file()
Sends a file to satisfy this request. Valid only in PreExit, Service, NameTrans, Error, and DataFilter steps.

```void
void HTTPD_file{
    unsigned char *handle, /* i; handle */
    unsigned char *name, /* i; name of file to send */
    unsigned long *name_length, /* i; length of the name */
    long *return_code); /* o; return code */
```

HTTPD_exec()
Executes a script to satisfy this request. Valid in PreExit, Service, NameTrans, and Error steps.

```void
void HTTPD_exec{
    unsigned char *handle, /* i; handle */
    unsigned char *name, /* i; name of script to execute */
    unsigned long *name_length, /* i; length of the name */
    long *return_code); /* o; return code */
```
HTTPD_read()
Reads the body of the client’s request. Uses HTTPD_extract for headers. Valid only in the PreExit, Service and Data Filter steps.

Migration note: On the OS/390 system, your applications no longer need to take the additional step to translate the data from ASCII to EBCDIC. If you are migrating from a previous version of the HTTP Server, check your GWAPI functions for this processing.

```
void HTTPD_read(
    unsigned char *handle, /* i; handle */
    unsigned char *value, /* i; buffer in which to place data */
    unsigned long *value_length, /* i/o; buffer size/length of data */
    long *return_code); /* o; return code */
```

HTTPD_write()
Writes the body of the response. Uses HTTPD_set and HTTPD_extract for headers. Valid in the PreExit, Service, NameTrans, Error, and Data Filter steps.

If you do not use HTTPD_set() to set the content type before the first time you call this function, the server will assume you are sending a CGI data stream.

You may need to use HTTPD_set() to set the CGI environment variable CONTENT_ENCODING to the appropriate code page for the content of your response (for example “ebcdic”) before you send the data to the client.

```
void HTTPD_write(
    unsigned char *handle, /* i; handle */
    unsigned char *value, /* i; data to send */
    unsigned long *value_length, /* i; length of the data */
    long *return_code); /* o; return code */
```

HTTPD_log_error()
Writes a string to the server’s error log.

```
void HTTPD_LINKAGE HTTPD_log_error(
    unsigned char *handle, /* i; handle */
    unsigned char *value, /* i; data to write */
    unsigned long *value_length, /* i; length of the data */
    long *return_code); /* o; return code */
```

Attention: Plug-in writers do not need to escape their percent (%) signs when using HTTPD_log_error. Previous releases use the string as a printf format string which required you to escape (%%) any percent characters in the error string.

HTTPD_log_trace()
Writes a string to the server’s trace log. HTTPD_log_trace() uses printf format strings to write to the trace log. If you include a percent (%) character in the message, you must escape it using %%. 

```
unsigned char *value="CPU utilization at 100%";
unsigned long value_length=strlen(value);
long return_code;

HTTPD_log_trace(handle, value, &value_length, &return_code);
```

The above example will display the message CPU utilization at 100%.
void
HTTPD_LINKAGE
HTTPD_log_trace(
    unsigned char *handle,    /* i; handle */
    unsigned char *value,     /* i; data to write */
    unsigned long *value_length, /* i; length of the data */
    long *return_code);    /* o; return code */

HTTPD_restart()
Restarts the server after all active requests have been processed. Valid in all steps EXCEPT for ServerInit, ServerTerm, and Datafilter.
void
HTTPD_restart(
    long *return_code);    /* o; return code */

HTTPD_proxy()
Makes a proxy request. Valid in PreExit and Service steps.

Note: This is a completion function; the response is complete after this function.
void
HTTPD_LINKAGE
HTTPD_proxy(
    unsigned char *handle,    /* i; handle */
    unsigned char *url_name,  /* i; url for the proxy request */
    unsigned long *name_length, /* i; length of the url */
    unsigned char *request_body, /* i; body of the request */
    unsigned long *body_length, /* i; length of the body */
    long *return_code);    /* o; return code */

HTTPD_local_security()
For OS/390, this function establishes a local host (RACF) user ID based on an application name (Lotus Notes) and user ID. The system default authorization step establishes the local security environment based on configured Protect rules and the local user ID. The local (RACF) user ID is obtained by calling the RACF Generic ID Mapping function. This function is valid ONLY in the authentication step.

For detailed steps on activating the Lotus Notes Adapter function for the HTTPD_local_security() to work properly, see the installation steps in “Step 6. Activate the Lotus Notes Adapter function and HTTPD_local_security function (optional)” on page 22. Also, see OS/390 Security Server (RACF) Callable Services or Macros and Interfaces documentation for RACF function details.

void
HTTPD_local_security(
    unsigned char *handle,
    unsigned char *appl_userid, /* Notes short name */
    unsigned long *appl_userid_len,
    unsigned char *appl_name, /* NOTES */
    unsigned long *appl_name_len,
    long *reason_code,
    long *return_code)

Note: The application’s GWAPI is responsible to verify the application user ID and password are authentic before calling HTTPD_local_security. HTTPD_local_security does not authenticate an application user ID or password; it only retrieves and sets the associated RACF user ID if one exists.
The server will fill in the reason code parameter with one of these values depending on the success of the HTTPD_local_security() request.

<table>
<thead>
<tr>
<th>Reason Code Value</th>
<th>Return Code Value/Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>HTTPD_SUCCESS</td>
</tr>
<tr>
<td></td>
<td>The local user ID is correct.</td>
</tr>
<tr>
<td>2</td>
<td>HTTPD_FAILURE</td>
</tr>
<tr>
<td></td>
<td>Generic ID Mapping failed with one of the following RACF return codes: 16, 24, 28, 32 (no user ID mapping available).</td>
</tr>
<tr>
<td>3</td>
<td>HTTPD_FAILURE</td>
</tr>
<tr>
<td></td>
<td>Generic ID Mapping failed with a RACF return code of 20. Not authorized to use this service.</td>
</tr>
<tr>
<td>4</td>
<td>HTTPD_FAILURE</td>
</tr>
<tr>
<td></td>
<td>Generic ID Mapping failed with other RACF return codes. Internal parameter list error or RACF error.</td>
</tr>
<tr>
<td>5</td>
<td>HTTPD_Failure</td>
</tr>
<tr>
<td></td>
<td>Internal error setting local user ID. Contact the IBM Software Support Center.</td>
</tr>
</tbody>
</table>

**Note:** Once an HTTPD_function returns, it is safe for you to free any memory you passed with it.

**Return codes from predefined functions and macros**

The server will fill in the return code parameter with one of these values depending on the success of the request.

<table>
<thead>
<tr>
<th>Value</th>
<th>Status/Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1</td>
<td>HTTPD_UNSUPPORTED</td>
</tr>
<tr>
<td></td>
<td>The function is not supported.</td>
</tr>
<tr>
<td>0</td>
<td>HTTPD_SUCCESS</td>
</tr>
<tr>
<td></td>
<td>The function succeeded and the output fields are valid.</td>
</tr>
<tr>
<td>1</td>
<td>HTTPD_FAILURE</td>
</tr>
<tr>
<td></td>
<td>The function failed.</td>
</tr>
<tr>
<td>2</td>
<td>HTTPD_INTERNAL_ERROR</td>
</tr>
<tr>
<td></td>
<td>Encountered an internal error and cannot continue processing this request.</td>
</tr>
<tr>
<td>3</td>
<td>HTTPD_PARAMETER_ERROR</td>
</tr>
<tr>
<td></td>
<td>One or more invalid parameters were passed. For example, the variable you tried to extract is unknown.</td>
</tr>
<tr>
<td>4</td>
<td>HTTPD_STATE_CHECK</td>
</tr>
<tr>
<td></td>
<td>The function is not valid in this step.</td>
</tr>
</tbody>
</table>
### GWAPI programs

<table>
<thead>
<tr>
<th>Value</th>
<th>Status/Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>HTTPD_READ_ONLY</td>
</tr>
<tr>
<td></td>
<td>Returned only by HTTPD_set(). The variable is read-only and cannot be set by the application.</td>
</tr>
<tr>
<td>6</td>
<td>HTTPD_BUFFER_TOO_SMALL</td>
</tr>
<tr>
<td></td>
<td>Returned only by HTTPD_extract(). The provided buffer was too small.</td>
</tr>
<tr>
<td>7</td>
<td>HTTPD_AUTHENTICATE_FAILED</td>
</tr>
<tr>
<td></td>
<td>Returned only by HTTPDAuthenticate(). Examine the HTTP_RESPONSE and HTTP_REASON variables for more information.</td>
</tr>
<tr>
<td>8</td>
<td>HTTPD_EOF</td>
</tr>
<tr>
<td></td>
<td>Returned only by HTTPD_read(). Indicates the end of the request body.</td>
</tr>
<tr>
<td>9</td>
<td>HTTPD_ABORT_REQUEST</td>
</tr>
<tr>
<td></td>
<td>The request has been aborted because the client has provided an entity tag that did not match the condition specified by the request.</td>
</tr>
<tr>
<td>10</td>
<td>HTTPD_REQUEST_SERVICED</td>
</tr>
<tr>
<td></td>
<td>Returned by HTTPD_proxy. Indicates the function that was called completed the response for this request.</td>
</tr>
<tr>
<td>11</td>
<td>HTTPD_RESPONSE_AREADY_COMPLETED</td>
</tr>
<tr>
<td></td>
<td>The function failed because the response for that request has already been completed.</td>
</tr>
</tbody>
</table>

### GWAPI configuration directives

Each step in the request process has a configuration directive that allows you to indicate which of your application functions you want called and executed during that step. You can add these directives to your server’s configuration file by manually editing and updating it or by using the GWAPI Request Processing form in the server’s Remote Configuration and Administration forms.

### GWAPI usage notes

- When appropriate, you can indicate that you want your application function called for all URL requests or only for URL requests that match a specified mask.
- You can also have your Authentication functions called for every request or just for those with a type of Basic.
- The GWAPI directives, except for the Service and NameTrans directives, can be in any order in the configuration file and you do not need to include every one. If you do not have an application function for a particular step, just omit the corresponding directive.
- The Service and NameTrans directives work like the Exec directive and are dependent on its occurrence and placement relative to other mapping directives within the configuration file. This means that the server processes the Service, NameTrans, Map, Pass, Exec, Redirect, and Fail directives in their sequential order within the configuration file. When it successfully maps a URL to a file, it does not read or process any other of these directives.
- You can also have more than one configuration directive for a step. For example, you could include two NameTrans directives, each pointing to a different
application function. When the server performs the name translation step, it will process your name translation functions in the order in which they appear within the configuration file.

- Multiple IP configuration (using a trailing IP address or template) is supported only for the Service and NameTrans directives.
- If the server fails to load a specific application function or you have a ServerInit directive that does not return an OK return code, no other application functions for that compiled GWAPI program will be called and any processing specific to that program which was done up to this point will be ignored. Other GWAPI programs that you include in these directives, and their application functions, will not be affected.

**GWAPI directives and syntax**

<table>
<thead>
<tr>
<th>Directive</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>ServerInit</td>
<td>/path/file:function_name init_string</td>
</tr>
<tr>
<td>PreExit</td>
<td>/path/file:function_name</td>
</tr>
<tr>
<td>Authentication</td>
<td>type /path/file:function_name</td>
</tr>
<tr>
<td>NameTrans</td>
<td>/URL /path/file:function_name IP_address_template</td>
</tr>
<tr>
<td>Authorization</td>
<td>/URL /path/file:function_name</td>
</tr>
<tr>
<td>ObjectType</td>
<td>/URL /path/file:function_name</td>
</tr>
<tr>
<td>Service</td>
<td>/URL /path/file:function_name* IP_address_template</td>
</tr>
<tr>
<td>Data Filter</td>
<td>/path/file:function_name:function_name:function_name</td>
</tr>
<tr>
<td>PICSDBLookup</td>
<td>/path/file:function_name</td>
</tr>
<tr>
<td>Log</td>
<td>/URL /path/file:function_name</td>
</tr>
<tr>
<td>Error</td>
<td>/URL /path/file:function_name</td>
</tr>
<tr>
<td>PostExit</td>
<td>/path/file:function_name</td>
</tr>
<tr>
<td>ServerTerm</td>
<td>/path/file:function_name</td>
</tr>
</tbody>
</table>

**GWAPI directive variables**

The variables in these directives have the following meanings:

- **type**: Used only with the Authentication directive. Determines if your application function will be called. Valid values are:
  - Basic: Application function is called only for basic authentication requests
  - *: Application function is called for all requests
  - /URL: Determines for which URLs your application function will be called. URL specifications in these directives are virtual (they do not include the protocol) but are preceded by a slash (/). For example, /www.ics.raleigh.ibm.com is correct but http://www.ics.raleigh.ibm.com is not. Valid values are:
    - A specific URL: Application function is called only for that URL
    - URL template: Application function is called only for URLs that match the template. You can specify a template as /URL*, /*, or *.

  - **Note**: A URL template is required with the Service directive if you want path translation to occur.

- **/path/file**: The fully qualified file name of your compiled program
GWAPI programs

:function_name
The name you gave your application function within your program

In the DataFilter directive, you must supply the names of the open, write, and close functions.

The Service directive requires an asterisk (*) after the :function_name, if you want to have access to path information.

init_string
Optional on the ServerInit directive, this can contain any arbitrary text you want to pass to your application function. Use HTTPD_extract() to extract the text from the INIT_STRING variable.

IP_address_template
Used only with the Service and NameTrans directives on servers that have more than one IP address. Determines if your application function will be called only for requests coming in on a specific IP address or on a range of IP addresses.

Compatibility with other APIs

The GWAPI is compatible with other APIs, such as CGI. You can run your existing CGI programs on all the server’s operating systems.

Porting CGI programs

Here are a few guidelines for porting CGI applications written in C to use the GWAPI:

1. Remove your main() entrypoint or rename it so you can build a DLL.
2. Eliminate global variables or protect them with a mutual exclusion semaphore.
3. Change the following calls in your programs:
   • Change printf() header calls to HTTPD_set().
   • Change printf() data calls to HTTPD_write().
   • Change getenv() calls to HTTPD_extract(). Note, this returns unallocated memory, so you must free the result.
4. Remember, the server runs in a multi-threaded environment and your application functions must be threadsafe. If the functions are re-entrant, performance will not decrease.
5. Do not forget to set the Content-Type header if you are using HTTPD_write() to send data back to the client.
7. Think about your error paths. If you generate error messages yourself and send them back as HTML, you should return HTTPD_OK from your service function(s).

GWAPI reference information

Authentication and authorization

First, a short review of the terminology:

Authentication
The verification of the security tokens associated with this request
Authorization

Process using the security tokens to determine if the requester has access to the resource

A visual overview of the server’s authentication and authorization process:

In the HTTP Server, authentication is part of the authorization process; it occurs only when authorization is required.

If your GWAPI application provides its own authorization process, it will override the default server authorization and authentication. Therefore, if you have Authorization directives in your configuration file, the application functions associated with them must also handle any necessary authentication. The predefined HTTPD_authenticate() function is to assist you with providing authentication.
There are three ways you can provide for authentication in your authorization application functions:

- Write your own separate authorization and authentication application functions. In your configuration file, use both the Authorization and the Authentication directives to specify these functions. Be sure to include HTTPD_authenticate() in your authorization application function.
  When the Authorization step is executed, it will perform your authorization application function which will, in turn, call your authentication application function.

- Write your own authorization application function but have it call the default server authentication. In your configuration file, use the Authorization directive to specify your function. In this case, you will not need the Authentication directive. Be sure to include HTTPD_authenticate() in your authorization application function.
  When the Authorization step is executed, it will perform your authorization application function which will, in turn, call the default server authentication.

- Write your own authorization application function and include all your authentication processing right in it. Do not use HTTPD_authenticate() in your authorization application function. In your configuration file, use the Authorization directive to specify your function. In this case, you will not need the Authentication directive.
  When the Authorization step is executed, it will perform your authorization application function and any authentication it included.

If your GWAPI application does not provide its own authorization process, you can still provide customized authentication. To do this:

- Write your own authentication application function. In your configuration file, use the Authentication directives to specify your function. In this case, you will not need the Authorization directive.

When the Authorization step is executed, it will perform the default server authorization which will, in turn, call your authentication application function.

Things to remember:

If you do not have any Authorization directives in your configuration file, or their specified application functions decline to handle the request, the server’s default authorization will occur.

If you do have Authorization directives in your configuration file and their application functions include HTTPD_authenticate(), the server will call any authentication functions specified in the Authentication directives. If you do not have any Authentication directives defined, or their specified application functions decline to handle the request, the server’s default authentication will occur.

If you do have Authorization directives in your configuration file but their application functions do not include HTTPD_authenticate(), no authentication functions will be called by the server. You must code your own authentication processing as part of your authorization application functions or make your own calls to other authentication modules.
The HTTP Server will automatically generate the challenge (prompting the browser to return userID and password) if you return 401 or 407 from your authorization exit. However, you must still configure a protection setup so that this will occur correctly.

**Environment variables**

For a description of variable names you can use in the Web server predefined functions, HTTPD_extract() and HTTPD_set(), see "Appendix E. Environment variables" on page 395. These variables contain values you can extract from a client request (read-only) or values you can set or create when processing a client request.

**Notes:**

1. User-defined variable names cannot have a prefix of `SERVER_`. The GWAPI function reserves any variable starting with `SERVER_` for the server and, therefore, is READ ONLY.

2. All headers sent by the client (such as Set-Cookie) are prefixed by "HTTP_" and their values can be extracted. To access variables that are headers, prefix the variable name with "HTTP_". You can also create new variables using the HTTPD_set() predefined function. See RCF 2068 for details about these headers.

**GWAPI on OS/390**

**OS/390 GWAPI REXX applications**

This section describes support for writing OS/390 GWAPI exit routines in REXX. The GWAPI REXX application supports two major activities:

- Invoking REXX executable programs from GWAPI exit routines

GWAPI REXX support allows REXX executable programs to be plugged in at the following GWAPI exit routines:

  - Authentication
  - Authorization
  - DataFilter
  - Error
  - Log
  - NameTrans
  - ObjectType
  - PICSDBLookup
  - PostExit
  - PreExit
  - ServerInit
  - ServerTerm
  - Service

**Note:** When calling the above exit routines, the first parameter MUST be the handle parameter

- Accessing GWAPI functions from REXX executable programs

Standard GWAPI support provides a set of functions for use by exit routines. As part of the GWAPI REXX support, a corresponding set of linkage routines are provided that enable REXX executable programs to call the GWAPI functions. The following table lists the REXX and GWAPI function names and a function...
**Invoking REXX executable programs as GWAPI applications**

Before invoking a REXX executable program as a GWAPI application, you need to know about:

- **GWAPI REXX DLL and function names**

  GWAPI REXX processing begins when a GWAPI directive is matched and the directive specifies one of the GWAPI REXX function names. The GWAPI REXX function establishes a REXX environment, then executes the REXX executable program whose name it derived from information on the directive and in the URL.

  The GWAPI REXX DLL and function names are:

  - **ServerInit exit**
    - IMWX00.so:IMWXSI
  
  - **DataFilter exits for open, write, and close, respectively**
    - IMWX00.so:IMWXD1:IMWXD2:IMWXD3
  
  - **All other GWAPI exits**
    - IMWX00.so:IMWX00

  More than one GWAPI REXX function is necessary to accommodate variations in the arguments passed.

- **REXX Exec Name and PATH_INFO**

  When a GWAPI REXX function is invoked, the name of the REXX executable program is obtained from the server PATH_INFO variable. As it does for any GWAPI exit routine, the server creates PATH_INFO from information specified in the directive and the request URL. The following example illustrates the process:

---

<table>
<thead>
<tr>
<th>REXX Function Name</th>
<th>GWAPI Function Name</th>
<th>Function Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMWXATR</td>
<td>HTTPD_attributes</td>
<td>Extracting file attributes</td>
</tr>
<tr>
<td>IMWXAUT</td>
<td>HTTPD_authenticate</td>
<td>Invoking the authentication function</td>
</tr>
<tr>
<td>IMWXFIL</td>
<td>HTTPD_file</td>
<td>Sending a file to the client</td>
</tr>
<tr>
<td>IMWXLGA</td>
<td>HTTPD_log_access</td>
<td>Writing to the server access log</td>
</tr>
<tr>
<td>IMWXLGT</td>
<td>HTTPD_log_trace</td>
<td>Writing to the server trace log</td>
</tr>
<tr>
<td>IMWXLOG</td>
<td>HTTPD_log_error</td>
<td>Writing to the server error log</td>
</tr>
<tr>
<td>IMWXPXY</td>
<td>HTTPD_proxy</td>
<td>Making a proxy request</td>
</tr>
<tr>
<td>IMWXRD</td>
<td>HTTPD_read</td>
<td>Reading input from the client</td>
</tr>
<tr>
<td>IMWXRVT</td>
<td>HTTPD_reverse_translate</td>
<td>Reverse translating a file system path to URL</td>
</tr>
<tr>
<td>IMWXSET</td>
<td>HTTPD_set</td>
<td>Setting server and local user variables</td>
</tr>
<tr>
<td>IMWXSLB</td>
<td>HTTPD_supply_label</td>
<td>Supplying a PICS label</td>
</tr>
<tr>
<td>IMWXTRN</td>
<td>HTTPD_translate</td>
<td>Translating a URL to a file system path</td>
</tr>
<tr>
<td>IMWXWRT</td>
<td>HTTPD_write</td>
<td>Writing output to the client</td>
</tr>
<tr>
<td>IMWXXTR</td>
<td>HTTPD_extract</td>
<td>Extracting server and local variables</td>
</tr>
</tbody>
</table>
If the Service directive is defined as:
Service /GWAPIRX*usr/lpp/internet/bin/IMWX00.so:
IMWX00/usr/lpp/internet/server_root/cgi-bin*/Omega

**Note:** The directive must be typed on one line, even thought it is shown here on two lines.

If the request URL is:
/GWAPIRX/userpgm.rx/Alpha/Beta

The server assigns the following value to the PATH_INFO variable and passes it to the GWAPI REXX function:
usr/lpp/internet/server_root/cgi-bin/userpgm.rx/Alpha/Beta/Omega

The GWAPI REXX executable program expects the PATH_INFO string passed from the server to begin with the fully-qualified file name of the REXX executable program to be executed. Qualifiers are checked from left to right until a valid file name is found. Then, before invoking the executable program, GWAPI REXX redefines PATH_INFO by stripping the file name from the string. Assuming that GWAPI REXX finds the following valid file name:
/usr/lpp/internet/server_root/cgi-bin/userpgm.rx

PATH_INFO will be redefined to contain the remaining fragment:
/Alpha/Beta/Omega

**Specifying directives for GWAPI REXX**

The following examples illustrate how to invoke a REXX executable program for each of the supported directives. The GWAPI REXX DLL symbolic link (IMWX00.so) default directory path is /usr/lpp/internet/bin and the GWAPI REXX executable program default directory path is /usr/lpp/internet/server_root/cgi-bin.

**EXAMPLE**

**ServerInit**
/usr/lpp/internet/bin/IMWX00.so:
IMWXS1/usr/lpp/internet/server_root/cgi-bin/DGWInit.rx
Authentication Basic
/usr/lpp/internet/bin/IMWX00.so:
IMWX00/usr/lpp/internet/server_root/cgi-bin/Authn.rx
Authorization /DB2*
/usr/lpp/internet/bin/IMWX00.so:
IMWX00/usr/lpp/internet/server_root/cgi-bin/Authz.rx
DataFilter
/usr/lpp/internet/bin/IMWX00.so:
IMWXD1:IMWXD2:IMWXD3/usr/lpp/server_root/cgi-bin/DataF.rx
Error /*
/usr/lpp/internet/bin/IMWX00.so:
IMWX00/usr/lpp/internet/server_root/cgi-bin/Error.rx
Log /cash*
/usr/lpp/internet/bin/IMWX00.so:
IMWX00/usr/lpp/internet/server_root/cgi-bin/Log.rx
NameTrans /file*
/usr/lpp/internet/bin/IMWX00.so:
IMWX00/usr/lpp/internet/server_root/cgi-bin/PostEx.rx
ObjectType /html
/usr/lpp/internet/bin/IMWX00.so:
IMWX00/usr/lpp/internet/server_root/cgi-bin/ObjTyp.rx
Post/Exit
/usr/lpp/internet/bin/IMWX00.so:
IMWX00/usr/lpp/internet/server_root/cgi-bin/PostEx.rx
PreExit
/usr/lpp/internet/bin/IMWX00.so:
IMWX00/usr/lpp/internet/server_root/cgi-bin/PreEx.rx
PICSIDBLookup
/usr/lpp/internet/bin/IMWX00.so:
IMWX00/usr/lpp/internet/server_root/cgi-bin/PICSIDBL.rx
ServerTerm
/usr/lpp/internet/bin/IMWX00.so:
IMWX00/usr/lpp/internet/server_root/cgi-bin/ICSTerm.rx
Service /GWAPIRX*
/usr/lpp/internet/bin/IMWX00.so:
IMWX00*
Note: The directives must be typed on one line, even though it is shown here on two lines.

For descriptions of these directives, see “Appendix C. Configuration directives” on page 267.

- Errors invoking a GWAPI REXX executable program

When a problem occurs attempting to invoke an executable program, GWAPI REXX writes the following message to the server error log:

Error error_code attempting to execute exec_name

error_code is one of the following values:

-99 A storage request failed.
-98 Necessary information, such as a path name or exec name, could not be extracted from the server variables.
-97 Problems occurred processing the executable program file. Possible problems include non-existent file, improper permission, or no free file descriptor.
-96 The executable program contained more than 16,384 lines.
-95 BPXWRBLD service which creates OS/390 REXX environment failed.

Writing GWAPI REXX Executable Programs

This section describes how to write GWAPI REXX executable programs.

GWAPI REXX executable program conditions: By the time a GWAPI REXX executable takes control, some environmental setup has been done. As with REXX CGIs, an array of REXX variables with stem name __environment has been defined. The variable __environment.0 contains the count. Note that environment variables must be prefixed by two underscores.

These contain the server variables passed to the exit. Each variable contains a string like the following:

REQUEST_METHOD=POST

In addition to the environment variables, arguments may be passed to the executable program, depending on the GWAPI exit and GWAPI REXX function involved:

IMWX00
No arguments

IMWXSI
Major version number, Minor version number

IMWXD1
OPEN

IMWXD2
WRITE

IMWXD3
CLOSE
Note in the GWAPI REXX directives shown earlier, only one REXX executable program is specified on the DataFilter directive; the executable is called for all three DataFilter exits. The argument passed to the exec indicates whether the call is for OPEN, WRITE, or CLOSE.

**Calling external subroutines and functions:** Note the method of invoking the initial GWAPI REXX executable program differs from the technique used to invoke external subroutines or functions called from the initial exec. Calls from the initial exec to external subroutines or functions are handled as described in the *OS/390 Using REXX to Access OpenEdition MVS Services* book. Basically, the unqualified executable program file names of external subroutines or functions are resolved using the list of HFS directories in the server PATH variable.

**Debugging and problem determination:** GWAPI REXX captures messages generated by the REXX interpreter during the execution of an executable program. This output, redirected to the server error log file, includes:

- Syntax and execution error messages
- Messages generated by TRACE and SAY instructions

**Transience of REXX variables:** REXX variables do not persist beyond the life of the GWAPI REXX exec that creates them. Therefore, it is not possible to pass information from one exit to another using REXX variables.

**Calling GWAPI predefined functions:** GWAPI applications must use GWAPI-defined services for certain functions, such as reading or writing client data, and manipulating server variables. The GWAPI REXX support provides linkage routines that enable execs to call these GWAPI-defined services. (Familiarity with the GWAPI HTTPD_xxxx service is assumed.)

To invoke a GWAPI service, an exec issues an ADDRESS LINKMVS command to call its corresponding link routine:

```
ADDRESS LINKMVS 'IMWXxxx parm1 parm2 ... parmj'
```

The link routines provided in the GWAPI REXX support are described below. The following points apply to all of the GWAPI REXX link routines:

- The GWAPI handle is automatically supplied to the underlying service.
- Only the character data values are specified to the routines. The length parameters required by the underlying GWAPI services are derived from the lengths of the REXX variables. A variable used for output must be defined, and be of sufficient length if the output may exceed 500 bytes.
- The REXX variables are limited to a maximum length of 32,767 bytes by the LINKMVS command. If the return code from ADDRESS LINKMVS is -2, the REXX routine may have to be coded so that no variable exceeds 32,767 bytes (for example, loop on writes until all data is written).
- The return code value from the underlying GWAPI service is passed back to the exec in the REXX RC variable, with one exception. If a routine returning output finds that a supplied variable is too short to contain the output data, it usually returns a value of 6 (HTTPD_BUFFER_TOO_SMALL). Because of the way LINKMVS works, the output variable is shorter than the output data and the output data exceeds 500 bytes. In this event, the variable RC will be set to the length required to contain the output data. If there is more than one output field, RC will be set to the maximum of the output lengths. Basically, if RC is greater than 500, retry the function after redefining the output fields with lengths equal to RC. For detailed information about the LINKMVS command, see the *REXX/MVS Reference Guide.*
GWAPI programs

Link routines to GWAPI functions: The following describes each of the GWAPI REXX routines.

IMWXATR (calls HTTPD_attributes)
This routine is used to extract the attributes of a file. This example shows how to call the routine to retrieve file attributes:

\[
\begin{align*}
\text{filenm} & = '/tmp/usr/demofile.text' \\
\text{fileattr} & = '' \\
\text{ADDRESS LINKMVS} & 'IMWXATR filenm fileattr'
\end{align*}
\]

IMWXAUT (calls HTTPD_authenticate)
This routine may be used by PreExit or Authorization execs to call Authentication. There are no parameters. The format is:

\[
\text{ADDRESS LINKMVS} 'IMWXAUT'
\]

IMWXFIL (calls HTTPD_file)
This routine is used to send a file to the browser from the server. This example shows how to send a file:

\[
\begin{align*}
\text{fname} & = '/u/gump/sample.txt' \\
\text{ADDRESS LINKMVS} & 'IMWXFIL fname'
\end{align*}
\]

IMWXLGA (calls HTTPD_log_access)
This routine is used to write a message to the server access log. This example shows how to call the routine:

\[
\begin{align*}
\text{msg} & = 'Application XYZ started for ICAPIRX/abc.rexxexec' \\
\text{ADDRESS LINKMVS} & 'IMWXLGA msg'
\end{align*}
\]

IMWXLGT (calls HTTPD_log_trace)
This routine is used to write a message to the server trace log. This example shows how to call the routine:

\[
\begin{align*}
\text{msg} & = 'Application XYZ initializing...' \\
\text{ADDRESS LINKMVS} & 'IMWXLGT msg'
\end{align*}
\]

IMWXLOG (calls HTTPD_log_error)
This routine is used to write a message to the server error log. This example shows how to call the routine:

\[
\begin{align*}
\text{msg} & = 'Application XYZ could not find dataset A.B.C.' \\
\text{ADDRESS LINKMVS} & 'IMWXLOG msg'
\end{align*}
\]

IMWXXY (calls HTTPD_proxy)
This routine is used to make a proxy request. This example shows the format:

\[
\text{ADDRESS LINKMVS} 'IMWXXY proxyurl reqbody'
\]

The REXX variable proxyurl contains the proxy url and reqbody contains the body of the request.

IMWXRD (calls HTTPD_read)
This routine is used to read input from the requester. The following example shows how to ready the client input, assuming that the REXX variable cntntlen contains the length from CONTENT_LENGTH:

If the data is to be converted to EBCDIC, CONVERT_REQUEST_BODY should be set to YES.

If the data is to remain ASCII, CONVERT_REQUEST_BODY should be set to NO. This is the default.

\[
\begin{align*}
inarea & = \text{substr(' ',1,cntntlen)} \\
\text{ADDRESS LINKMVS} & 'IMWXRD inarea'
\end{align*}
\]
IMWXRVT (calls HTTPD_reverse_translate)
This routine is used to translate a file system path to a URL. This example assumes that the REXX variable fspath contains a file name, and url will receive the output URL.
ADDRESS LINKMVS 'IMWXRVT fspath url'

IMWXSET (calls HTTPD_set)
This routine is used to set a server variable. The following example shows how to set variable MY_OWN_VARIABLE:
myname = 'MY_OWN_VARIABLE'
myvalue = 'whatever I want'
ADDRESS LINKMVS 'IMWXSET myname myvalue'

IMWXSLB (calls HTTPD_supply_label)
This routine is used to provide a PICS label. This example shows the format:
ADDRESS LINKMVS 'IMWXSLB data'
The REXX variable data contains label data.

IMWXTRN (calls HTTPD_translate)
This routine is used to translate a URL to a file system path. The following example assumes that REXX variable url contains the input URL, and REXX variables trurl, pathtr, and qstr will receive the translated URL, PATH_TRANSLATED, and QUERY_STRING outputs, respectively:
ADDRESS LINKMVS 'IMWXTRN url trurl pathtr qstr'

IMWXWRT (calls HTTPD_write)
This routine is used to write output to the requestor. The following example shows how to write a few lines of output to the client browser in one call:
stuff = '<PRE>This is the first line<BR>'
stuff = stuff 'and this is the second line<BR>'
stuff = stuff 'and here is the last line</PRE><HR>'
ADDRESS LINKMVS 'IMWXWRT stuff'
Note that IMWXWRT does not append a newline to output, so line breaks must be explicitly specified by the application.

A GWAPI REXX executable program cannot use the cgiutils command to set output headers. Output headers must be set through the appropriate server variables (for example: CONTENT_TYPE, CONTENT_ENCODING) using IMWXSET.

IMWXXTR (calls HTTPD_extract)
This routine is used to retrieve the value of a server variable. The following example shows how to get the value of variable STATUS_INDICATOR:
statname = 'STATUS_INDICATOR'
statind = ''
ADDRESS LINKMVS 'IMWXXTR statname statind'

GWAPI REXX executable program exit conditions:
When a GWAPI REXX executable program exits, it must set a numeric completion code with the EXIT instruction.

When the exec completion code is within the range of HTTP return codes (-1 to 599), the value is passed unchanged to the server, and the server disposes of the
request accordingly. When the exec completion code is outside the HTTP range, GWAPI REXX assumes it is an error indication and writes the following message to the server error log:

Completion code from REXX exec exec_name is completion_code

A completion code of HTTP_BAD_REQUEST (400) is passed back to the server.

**Example GWAPI REXX service executable program**

The following REXX executable program, which runs out of the Service exit, performs the following functions:
- Sets ICS variables in preparation for output
- Displays the set of input environment variables
- Prints an HFS file
- Displays an image
- Sets appropriate return code.

**Important syntax note:** Environment variables must be prefixed by two underscores.

**EXAMPLE**

/* REXX */

```rexx
*** Set HTTP Server variables for output ***
name = 'CONTENT_TYPE'
value = 'text/html'
ADDRESS LINKMVS 'IMWXSET name value'
name = 'CONTENT_ENCODING'
value = 'ebcdic'
ADDRESS LINKMVS 'IMWXSET name value'

*** List environment variables ***
output = 'List of all' __environment.0 'Environment Variables:<PRE>'
do e = 1 to __environment.0
   output = output right(e,'0').' __environment.e '<BR>'
end
output = output || '</PRE><HR>'
ADDRESS LINKMVS 'IMWXWRT output'

*** Print a file ***
filename = '/etc/httpd.envvars'
output = 'Display HFS file' filename '<PRE>'
ADDRESS LINKMVS 'IMWXWRT output'
ADDRESS LINKMVS 'IMWXFIL filename'
hsrcc = rc
output = '</PRE>'
ADDRESS LINKMVS 'IMWXWRT output'
if hsrcc => 0 then output = 'IMWXFIL failed with rc:' hsrcc
output = output '<HR>'
ADDRESS LINKMVS 'IMWXWRT output'

*** Display an image ***
out = 'Is this an image, or what <img src="/icons/unknown.gif">MHR'
ADDRESS LINKMVS 'IMWXWRT out'

*** Set successful return code ***
exit 200
```

**GWAPI REXX downloads**

To assist you in writing GWAPI REXX applications, a package of routines that can be called from REXX execs can be found at:
GWAPI programs

DB2IMSRX contains routines for REXX execs (GWAPI or CGI). The routines are:

**FORMWWWX**
- Parses client form inputs (POST or GET) into REXX variables

**DB2WWWX**
- Executes an SQL statement or a DB2 stored procedure using REXX variables inputs or outputs

**IMSWWWWX**
- Executes an IMS transaction (using APPC) using REXX variables as inputs and outputs

### Example GWAPI REXX data filter executable program

The following REXX executable program, which runs out of the Data Filter exit, opens a data filter that will search the datastream looking for Kappa and convert the string to KAPPA.

```rexx
/* REXX */
type = arg(1)
say 'Type is 'type
select
  when type = 'OPEN' then do;
    nm = 'DOCUMENT_URI'
    ADDRESS LINKMVS 'IMWXXTR nm vl'
    if rc = 0 & POS('rexx',vl) = 0 then orc = 000
    else orc = 200
    nm = 'DOCUMENT_URI'
    ADDRESS LINKMVS 'IMWXXTR nm vl'
    log = ''
    ADDRESS LINKMVS 'IMWXLOG log'
    exit orc
  end

  when type = 'WRITE' then do;
    indata = arg(2)
    outdata = ''
    do until indata = ''
      parse var indata piece 'Kappa' indata
      outdata = outdata || piece
      if indata | = '' then outdata = outdata || 'KAPPA'
    end
    say 'OUTDATA IS: 'outdata
    ADDRESS LINKMVS 'IMWXWRT outdata'

    nm = 'DOCUMENT_URI'
    ADDRESS LINKMVS 'IMWXXTR nm vl'
    wrc = 200
    log = ''
    ADDRESS LINKMVS 'IMWXLOG log'
    exit wrc
  end

  when type = 'CLOSE' then do
    nm = 'DOCUMENT_URI'
    ADDRESS LINKMVS 'IMWXXTR nm vl'
    crc = 200
    log = ''
    ADDRESS LINKMVS 'IMWXLOG log'
    exit crc
  end
end
```

Chapter 18. Writing GWAPI programs 225
Storing GWAPI programs

You can mark your GWAPI programs "Program Controlled" and store them in HFS. See for MVS for more information on HFS and Program Control.

HTCodePage_t, the other data type

A new enumeration data type has been defined in the HTAPI.h file to support specification of a code page by GWAPI users:

```c
typedef enum _HTCodePage_t {
    BINARY = 0,
    ASCII,
    EBCDIC,
    RAWNET,
    RAWFILE
} HTCodePage_t;
```

- **BINARY** no conversion
- **ASCII** equivalent to the code page specified by the DefaultNetCp configuration directive
  - Default value: ISO8859-1
- **EBCDIC** equivalent to the code page specified by the DefaultFsCp configuration directive
  - Default value: IBM-1047
- **RAWNET** data from network socket, format known to converter
- **RAWFILE** data from host file, format known to converter

an example of this data type declaration is:

```c
HTCodePage_t codepage;
```

GWAPI samples

To help you get started with your own GWAPI functions, look at the sample programs on the Web site. You can access these samples from the Web server Library page at URL:

```
http://www.ibm.com/software/websphere/httpservers/doc52.html
```

Debugging C/C++ GWAPI programs

You can use the OS/390 Debug Tool to debug your C/C++ GWAPI programs. This tool is shipped as part of the Language Environment (LE) component of OS/390.
Overview of support and restrictions

This interactive tool allows you to debug C/C++ GWAPI programs remotely from your workstation. Set a break point with the simple click of the mouse. Use the windowing capabilities of your workstation to view multiple segments of your source and your storage, while monitoring a variable at the same time.

Note the following restrictions:

- The Debug Tool supports only one debugging operation on the Web server at a time.
- The Debug Tool supports multi-threaded requests. However, you cannot debug multiple requests concurrently. If the tool has started debugging a request, subsequent requests will be queued and started when the previous request completes.
- You cannot start the Debug Tool from the MVS console. After you set the required Web server directives, the tool will start automatically when the Web server is started.
- To configure the Web server, you must manually update the configuration file; there is currently no support for using the Configuration and Administration Forms graphical user interface.

Getting started

This quick start example shows the required steps for enabling and using the Debug Tool:

**Step 1. Install the remote debugger**
The remote debugger used with the OS/390 Debug Tool is the OS/390 C/C++ Productivity Tools Debugger. This debugger is the workstation component and graphical user interface for the Debug Tool.

For more information, go to URL: [http://www.ibm.com/software/ad/c390/pt/dbg1.htm](http://www.ibm.com/software/ad/c390/pt/dbg1.htm)


**Step 2. Compile your C/C++ program with the TEST compile-time option**
Before testing your C/C++ program with the Debug Tool, you must compile it with the TEST option.

For instructions, see the Debug Tool User’s Guide and Reference.

To access Debug Tool documentation on the Web, go to URL: [http://www.ibm.com/s390/dt/](http://www.ibm.com/s390/dt/)

**Step 3. Edit the Web server configuration file**
To enable support for the Debug Tool, update the following directives in the configuration file:

- “DebugToolAddr - Identify the workstation running the Remote Debugger” on page 312
- GWAPI directives
GWAPI programs

Each step in the request process has a configuration directive that allows you to indicate which of your application functions you want called and executed during that step.

To enable the Debug Tool, you simply add the prefix `dbg` to the appropriate GWAPI directive. For example, to debug a GWAPI program that is called during the PreExit step, add the prefix `dbg` to the PreExit directive in the configuration file:

```
dbgPreExit /path/file:function_name
```

For a list of GWAPI directives, see “GWAPI directives and syntax” on page 213.

Step 4. Start the remote debugger on the workstation

Step 5. Start or restart the Web server

Troubleshooting hints and tips

Debug Tool hints and tips
To access Debug Tool hints and tips on the Web, go to URL:

```
http://www.ibm.com/s390/dt/
```

Web server error messages
Messages IMW0346E-IMW0349E will be issued to the console or to the error log if there are configuration or other errors related to the Debug Tool.

You can use the `-grep` command to help locate an error message. The command syntax is:

```
-grep -message_number
```

If you cannot determine the cause of the problem from error messages, the `-vv` trace log can provide more information.
Chapter 19. Accessing LDAP information with the LDAP API

Overview of LDAP API

The HTTP Server allows access to LDAP information through the use of a plug-in or API.

For example, your plug-in could open its own connection (using ldap_open) to the LDAP server that the HTTP Server is using to access shared information. The plug-in would need information about the LDAP server, such as host name or port numbers, to do this. The API could bind using ldap_bind to connect and authenticate with the server, then search using ldap_search for the relevant LDAP information.

LDAP uses a set of DAP functions which decreases your system requirements:

<table>
<thead>
<tr>
<th>Function</th>
<th>Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>ldap_bind</td>
<td>connect and authenticate with an LDAP server</td>
</tr>
<tr>
<td>ldap.unbind</td>
<td>log off and disconnect from an LDAP server</td>
</tr>
<tr>
<td>ldap_abandon</td>
<td>disconnect from a non responding LDAP server</td>
</tr>
<tr>
<td>ldap_search</td>
<td>query an LDAP server</td>
</tr>
<tr>
<td>ldap_add</td>
<td>add a new entry</td>
</tr>
<tr>
<td>ldap_delete</td>
<td>delete an entry</td>
</tr>
<tr>
<td>ldap_modify</td>
<td>change an existing entry</td>
</tr>
<tr>
<td>ldap_rdn</td>
<td>get the relative distinguished name</td>
</tr>
</tbody>
</table>

For information on configuring your Web server to access LDAP servers, see "Chapter 14. Retrieving LDAP information" on page 155.

To view LDAP documentation, go to the OS/390 Internet Library at URL: http://www.ibm.com/s390/os390/bkserv/

Accessing LDAP configuration information

To connect to the same LDAP server that the HTTP Server uses to store information, you must access the information in the server configuration file. This section describes how to extract server information for use in your own API.

You can access the LDAP configuration information using the HTTPD_extract call. Parameters to access the configuration file using HTTPD_extract are:

LDAP:LABELLIST

Retrieves all labels defined in LDAPInfo directives in the configuration file. Each label is separated by a new line character.
LDAP API

LDAP:LABEL:<label>::<fieldName>
retrieves the value of the subdirective for the corresponding label in the
LDAPInfo directive. Any LDAPInfo subdirective is a value for fieldName.
For example, to access the server authentication type of the LDAPInfo
directive labeled PrimaryLdapServer, use the extract call:

HTTPD_extract (handle,"LDAP:LABEL:PrimaryLdapServer:ServerAuthType", ...)

fieldName above has an additional value not included in the LDAPInfo
subdirectives. ServerPassword is a valid field name, and returns the
decrypted password from the stash file for LDAP authentication.

LDAP:LABEL:CURRENT::<fieldName>
Uses the currently used LDAPInfo directive to retrieve the value of the
specified subdirective. An LDAPInfo directive is current when it is
associated with a protection setup that has matched the current request.

fieldName above has an additional value not included in the LDAPInfo
subdirectives. ServerPassword is a valid field name, and returns the
decrypted password from the stash file for LDAP authentication.

Note: For more information, see the description of the LDAPInfo directive in
"LDAPInfo - Define an external LDAP server" on page 318.

After you access the LDAP server information from the configuration file, a log file
named LDAP_TRACE_LOG is created in the LOGS directory. The log file reports
successful and unsuccessful readings from the configuration file.

Sample LDAP program

You can link to a sample LDAP API program on the HTTP Server Web site at URL:
http://www.ibm.com/software/websphere/httpservers/doc52.html

The example code obtains the labels, or subdirective values, in all LDAPInfo
directives in the configuration file. A log file named ldap is created containing
information on the labels attempted and read. It will also display the values for
each label obtained successfully.
Part 7. Appendixes
Appendix A. Certificate authority utility

Overview of certificate authority options .............................................. 233
Before you begin .................................................................................. 234
Migrate existing CA utility files .............................................................. 234
Install WebSphere Application Server and make required jvm.properties file updates ......................................................... 234
Verify that you are using a compatible browser .......................................... 234
Ensure required MIME types are in the server configuration file ................. 235
Create key databases and certificates using IKEYMAN .......................... 235
Copy CA key database and certificate into the CA utility directory .......... 235
Export your CA key database into the CA utility ........................................ 235
Enable client certificates for %%CLIENT%% support (optional) ............... 235
Accessing the HTTP Server CA utility .................................................... 236
Using HTTP Server CA to process client and server certificates ............... 236
Overview of CA process ...................................................................... 236
Administrator tasks ............................................................................ 236
Exporting the CA key database ............................................................. 236
Processing a certificate request .............................................................. 237
User tasks ......................................................................................... 237
Downloading a CA certificate to a Web browser ..................................... 237
Requesting a browser certificate ........................................................... 237
Receiving an approved browser certificate ............................................ 238
Webmaster tasks ............................................................................... 239
Downloading a CA certificate for another server ..................................... 239
Requesting a server certificate .............................................................. 239
Receiving an approved server certificate ................................................ 240

Overview of certificate authority options

To conduct commercial business on the Internet, we recommend that you purchase a secure server certificate from an external certificate authority (CA) such as VeriSign. For a list of supported CAs, see “Certificate authorities supported by the HTTP Server” on page 54.

If you act as your own CA, you can sign your own or anyone else’s certificate requests. This is a good choice if you only need the certificates within your private Web network and not for outside Internet commerce. Clients must have browsers, such as Netscape Navigator or Microsoft Internet Explorer, that can receive your CA certificate and designate you as a trusted CA.

To act as your own CA, you can use your server key management utilities (IKEYMAN and HTTP Server CA), or you can purchase certificate authority software from a CA provider.

Note: If you expect to administer over 250 certificates, you may want to consider purchasing software from a CA provider. This limit is based on the number of certificate requests that can be stored in the CA key database. After storing 250 certificate requests, performance of the utility is very slow.

- For an example using IKEYMAN and HTTP Server CA, see “Setting up secure connections using self-signed certificates” on page 63.

- If you purchase certificate authority software, the HTTP Server supports the following CA software:
  - IBM Registry
    If you are using certificate revocation lists (CRLs) for client authentication, you must use CA software from the IBM Registry. For more information, go to URL http://www.ibm.com/software/commerce/registry/
  - Entrust Demo Certificates and Entrust WEBCA
    For more information, go to URL http://www.entrust.com/
  - Netscape Certificate Server
    For more information, go to URL http://www.netscape.com/.
HTTP Server CA

- XCert
  For more information, go to URL http://www.xcert.com/.
- Any X.509–compliant certificate authority

Before you begin

Migrate existing CA utility files

In Release 5.1, the installation path for the CA utility changed. For important migration information, see “Certificate authority utility changes” on page 13.

Install WebSphere Application Server and make required jvm.properties file updates

Before using the CA utility, you must install the WebSphere Application Server as your Java-based servlet engine. For installation instructions, refer to the Application Server documentation. To access the most current V1.2 and V1.1 Application Server documentation, go to URL http://www.ibm.com/software/websphere/appserv/library.html.

After installing WebSphere Application Server, you must make the following configuration updates to the Application Server jvm.properties file by updating the Application Server was.conf file:

- The directory where the CA utility is located must be added to the ncf.jvm.classpath property in the jvm.properties file.
  The CA utility is located in the following directory:
  \install_path\server_root\CAServlet\language_code

  \install_path\ is the root directory of your Web server installation; the default install path is /usr/lpp/internet. \language_code is either C (English) or Ja_JP (Japanese).

  For example, if you are using the default install path and an English server, you would add the following directory to the ncf.jvm.classpath property:
  \usr/lpp/internet/server_root/CAServlet/C

- The \install_path/bin directory for the Web server must be added to the ncf.jvm.libpath property in the jvm.properties file and must be ahead of the /usr/lib directory.
  For example, if you are using the default install path, you would add the following directory to the ncf.jvm.classpath property ahead of the /usr/lib directory:
  \usr/lpp/internet/bin

  For information on updating the Application Server was.conf file, refer to your Application Server documentation.

Verify that you are using a compatible browser

You can use the Netscape Navigator or Microsoft Internet Explorer browser with HTTP Server CA. For more information, see “Browser requirements” on page 4.
Ensure required MIME types are in the server configuration file

For client certificates to download correctly, the following AddType directives and MIME types must be included in the server configuration file:

```
AddType .cer application/x-x509-user-cert ebcdis 0.5 #User certificate
AddType .der application/x-x509-ca-cert binary 1.0 #Browser CA certificate
```

These MIME types are included in the default httpd.conf configuration file.

Create key databases and certificates using IKEYMAN

Before using the CA utility, create your server and CA key databases, server certificate, and self-signed CA certificate using IKEYMAN. For an example, see “Setting up secure connections using self-signed certificates” on page 62.

**Note:** As the Administrator, you can choose to have CA certificate requests processed automatically after they are submitted. Automatic processing eliminates the need for you to manually process each certificate request. If you want automatic processing, create a stash file for the CA key database password when you set up your CA key database using IKEYMAN.

Copy CA key database and certificate into the CA utility directory

After you create your CA key database and self-signed CA certificate, copy the cakey.kdb file into the CA utility directory. The default directory is:

```
/usr/lpp/internet/server_root/CAServlet/language_code/
```

For `language_code`, enter C (English) or Ja_JP (Japanese).

You can optionally choose to store your encrypted CA key database password in a stash file. If the stash file is in the same directory as the CA key database file (cakey.kdb), then all certificates that are sent to be signed by the CA utility will be approved automatically. If you create a stash file (cakey.sth), copy that file into the same directory as the cakey.kdb file.

Export your CA key database into the CA utility

Your cakey.kdb file on the server must be exported into the CA utility in a format that can be used by clients and servers. For detailed instructions, see “Exporting the CA key database” on page 236.

Enable client certificates for %%%CLIENT%%% support (optional)

To enable the new client certificates for %%%CLIENT%%% support, the client certificate must also be installed (or registered) in the system security product. For the OS/390 Security Server, Version 2 Release 5 or higher (RACF FMID HRF2240), it is possible to establish a registration Web page where the client can self-register the new certificate.

For more information, see the documentation included in the RACF APAR OW31933 or go to URL http://service.software.ibm.com/390.ww/psearch??HBW=nd.
Accessing the HTTP Server CA utility

To access the HTTP Server CA utility, go to URL:

http://your.server.name/CAServlet/Welcome.html

Using HTTP Server CA to process client and server certificates

Overview of CA process

Your job as CA is to verify that a certificate should be issued for a client or server. You need to make sure that the person making the request has a legitimate claim to request the certificate. After you have verified a person’s claim, you can create signed certificates using HTTP Server CA.

The input to this process is a client or server certificate request. The output will be a certificate signed with your private key.

After you have processed the client or server certificate:

1. You will notify the client or server to download your CA certificate. Detailed steps are shown in “Downloading a CA certificate to a Web browser” on page 237 and “Downloading a CA certificate for another server” on page 239.

2. You will notify the client or server to download the certificate you have signed (CA-signed certificate) and receive it into their client or server operational key database. Detailed steps are shown in “Receiving an approved browser certificate” on page 238 and “Receiving an approved server certificate” on page 240.

After completing these steps, the client or server can use their CA-signed certificate to communicate securely with other HTTP Servers and Web browsers in your private Web network.

Administrator tasks

Exporting the CA key database

Use these steps to export the CA key database (cakey.kdb) on the server in a format that can be used by browsers and servers. You do not need to repeat these steps after the initial setup of your certificate authority.

Note: You must create the cakey.kdb file using IKEYMAN before you can use the CA utility. For an example, see “Step 2. Create your CA key database and certificate” on page 63.

To export your CA key database file for browsers and servers:

1. Go to URL:

   http://your.server.name/CAServlet/Welcome.html

2. From the Front Page of the utility, click Administration to access the Administration Tasks page.

3. Select Export CA keys for browsers and enter your CA key database password.

4. Click Process Now. This creates a file called cakey.der. You should get a message confirming that the CA key was successfully exported.

5. On the Administration Tasks page, select Export CA keys for servers and enter your CA key database password.
6. Click **Process Now**. This will create a file called cakey.txt. You should get a message confirming that the CA key was successfully exported.

**Note:** You may receive an error message when performing these operations if you are not using the following permission bit settings: CA utility directory, 757; cakey.kdb file, 666; and cakey.sth file, if created, 644.

### Processing a certificate request

**Notes:**
1. If you created a stash file for your CA key database password (cakey.sth), all browser and server certificate requests will be automatically approved.
2. Ensure that certificate requests are processed over a secure connection.

To process certificate requests from browsers and servers:
1. Go to URL: 
   `
   http://your.server.name/CAServlet/Welcome.html
   `
2. From the Front Page of the utility, click **Administration** to access the Administration Tasks page.
3. Select **Process pending requests** or **Process all requests**, then click **Process Now**.
   
   If there are waiting requests:
   - If you select **Process pending requests**, you can **Defer**, **Approve**, or **Deny** the request.
   - If you select **Process all requests**, you can **Keep** or **Delete** the request.

   You can take action on one or multiple requests.
4. Click **Process**. You should get a message confirming that the certificate database was successfully updated.

For more detailed instructions, click **Help**.

### User tasks

#### Downloading a CA certificate to a Web browser

Before you can accept a browser certificate signed by HTTP Server CA, you must first download the CA certificate that identifies HTTP Server CA as a trusted certificate authority to your browser.

To download a CA certificate:
1. Go to URL:
   `
   http://your.server.name/CAServlet/Welcome.html
   `
2. Under Browser Certificates, click **Download CA certificate from the Webserver**.
3. Follow the instructions for downloading the cakey.der file to your workstation.

After downloading the CA certificate, designate the certificate as trusted by your browser and mark the certificate as the default in your operational key database. Once the CA certificate is designated as a trusted CA by your browser, you do not need to perform this procedure again.

#### Requesting a browser certificate

Use the following steps to request a browser certificate from the CA of your private network:
1. Go to URL:
HTTP Server CA

http://your.server.name/CAServlet/Welcome.html

2. Under Browser Certificates, click **Request a browser certificate**.

3. In the Common Name field, specify a name of your choice. Make a note of this name because you will need it later when you receive your signed certificate.

4. Enter your password in the Challenge Phrase field. You will need to specify this password later when you download your approved certificate.

5. Complete all the required fields. For additional information, click **Help**.

   **Note:** The optional State field requires 3 or more characters. Blanks and periods are accepted.

6. Click **Submit Requests** to send the completed form to the CA for approval. The process of generating a private key will be started on your browser. This private key will be used with the certificate you are requesting.

7. Click **OK** on your browser to create the private key.

Depending on your Administrator’s setup, your request will either be approved automatically or after the Administrator has reviewed it. If automatic approval occurs, you will not get any messages confirming that the request has been approved. You can go directly to “Receiving an approved browser certificate” and complete that procedure.

If your Administrator has set up manual approval, you will receive a message confirming that the certificate request has been accepted. Check with your Administrator to determine when the approval will be processed. Approved certificates are sent to the HTTP Server CA database directory.

**Receiving an approved browser certificate**

Before you can receive an approved browser certificate, you must download the CA’s certificate (cakey.der). For instructions, see “Downloading a CA certificate to a Web browser” on page 237.

Use the following steps to receive an approved CA-signed browser certificate:

1. Go to URL:
   http://your.server.name/CAServlet/Welcome.html

2. Under Browser Certificates, click **Receive the approved certificate**.

3. Enter the Common Name you specified on the certificate request form.

   **Note:** If you get a message that no record is found, you may have entered your common name incorrectly. If you have forgotten the Common Name you entered, check with your Administrator.

4. Enter the password in the Challenge Phrase field, then click **Submit Request**.

   **Note:** If you get a message that no record is found, you may have entered your password incorrectly. If you have forgotten your password, check with your Administrator.

5. If the certificate is found, the Download Certificate page will display, and you can download the certificate. To start the download process, click **Click here to download your certificate**.

   If not found, check with the CA Administrator to find out when the certificate will be processed.

6. After you receive the approved certificate, your browser will be trusted by other clients and servers in your private network.
Webmaster tasks

Downloading a CA certificate for another server
Before you can process any signed HTTP Server CA certificates on another Web server, you must first store your HTTP Server CA certificate in your target Web server’s operational key database.

To download a CA certificate for another server:
1. Go to URL:
   http://your.server.name/CAServlet/Welcome.html
2. Under Server Certificates, click **Download CA certificate from the Webserver**.
   The cakey.txt file is displayed.
3. Using your 3270 emulator copy function, copy the certificate file to your clipboard.
4. Log on to the target Web server’s system.
5. Go into the OS/390 UNIX System Services shell on the target system, and create a file in the operational key database called *.txt, then use OEDIT or some other editor to open the file.
6. Paste the certificate file from your clipboard into the new *.txt file you created.
7. Store this CA certificate (*.txt file) in the operational key database of the target Web server using IKEYMAN. For detailed instructions, see "Storing a CA’s certificate" on page 418.

Requesting a server certificate
Use the following steps to request a server certificate from the CA of your private network:
1. Go to URL:
   http://your.server.name/CAServlet/Welcome.html
2. Under Server Certificates, click **Request a server certificate**. The Server Certificate Request form appears.
   Use this form to send a server certificate request to the CA of your private network so that this HTTP Server will be trusted by the browsers and other HTTP Servers in the network. If there are other Web servers in your network, you can perform this task on behalf of the remote Web servers.
3. In the Server Name field, specify a server name of your choice. Make a note of this name because you will need it later when you download your approved certificate.
4. In the Organization field, specify an organization name of your choice.
5. Enter your password in the Challenge Phrase field. You will need to specify this password later when you download your approved certificate.
6. Complete all the required fields. For additional information, click **Help**.
7. Using a 3270 emulator session, browse your Web server’s operational certificate request file (*.arm). There are several OS/390 UNIX utilities or commands you can use to display the *.arm file such as OBROWSE, cat, and pg. When the *.arm file is displayed on your screen, use your 3270 emulator to copy the contents of this file to your clipboard. Then go to your Web browser and paste this file into the section of the form that says: **Please copy your certificate requests into the following area.** You must paste your operational certificate request file (*.arm) into this space. This is the file that is sent to the HTTP Server CA to be signed.
8. Click **Submit Requests** to send the completed form to the CA for approval.
HTTP Server CA

9. If your CA administrator has set up automatic approval, you can click on the **Click here to download your certificate** option to start the process. If automatic approval occurs, you will not get any messages confirming that the request has been approved. You can go directly to "Receiving an approved server certificate" and complete that procedure.

   If your CA administrator has set up manual approval, you should receive a message on your browser confirming that the certificate request has been accepted.

   **Receiving an approved server certificate**

   Before you can receive an approved server certificate, you must download the CA’s certificate (cakey.txt). For instructions, see "Downloading a CA certificate for another server" on page 239.

   Use the following steps to receive an approved CA-signed server certificate:

   1. Go to URL:  
      `http://your.server.name/CAServlet/Welcome.html`
   2. Under Server Certificates, click **Receive the approved certificate**.
   3. Enter the Server Name you specified on the certificate request form.
      
      **Note:** If you get a message that no record is found, you may have entered your server name incorrectly. If you have forgotten the Server Name you entered, check with your Administrator.
   4. Enter the password in the Challenge Phrase field, then click **Submit Request**.
      
      **Note:** If you get a message that no record is found, you may have entered your password incorrectly. If you have forgotten your password, check with your Administrator.
   5. If the certificate is found, the Download Certificate page will display, and you can download the certificate. To start the download process, click **Click here to download your certificate**.
      
      If not found, check with the CA Administrator to find out when the certificate will be processed.
   6. After the download process is complete, a signed certificate request file will display in your Web browser. Use your copy and paste function to copy this file to the clipboard. Paste the signed certificate into a *.cert file in the HFS on your server.
   7. Receive the signed certificate into your server operational key database (*.kdb) using IKEYMAN. For detailed instructions, see "Receiving a certificate signed by a trusted CA" on page 417.
   8. After you receive the approved certificate, your server will be trusted by other HTTP Servers and Web browsers in your private network.
Appendix B. Commands

**cgiparse command**

Use the cgiparse command to parse the QUERY_STRING environment variable (in the case of GET method) or standard input (in the case of POST method) for CGI scripts. The cgiparse command can also be used to read CONTENT_LENGTH characters from standard input. All returned output is written to its standard output.

The QUERY_STRING is a string of EBCDIC text. Certain characters (+, <, >, (, ), %, *, /, ?, and space) are escaped by the client. Spaces are replaced by a plus sign (+), therefore, a real plus sign must be encoded. The other special characters are set to a percent character followed by the hexadecimal form of the character in the ASCII (codepage ISO8859-1) character set. The following examples show the hexadecimal form for each special character:

```
+ %2B
? %3F
space %20
< %3C
> %3E
( %28
) %29
% %25
" %22
/ %2F
```

**Syntax**

```
cgiparse -Flag [Modifier]
```

**Flags**

Flags have one-character equivalents: -k -f -v -r -i -s -p -c -q -N -F -P
Commands

- **keywords**
  Parses QUERY_STRING for keywords. Keywords are decoded and written to standard output, one per line. A space (+) is interpreted as the delimiter between keywords.

- **form**
  Parses QUERY_STRING as form request. Returns a string which, when evaluated by the shell, will set shell variables with the prefix FORM_ followed by a field name. Field values are the contents of the variables. Variable names on a form request are restricted to include only alphabetical or numeric characters that are in the single byte character set.

- **value** field-name
  Parses QUERY_STRING as form request. Returns only the value of field-name.

- **read**
  Reads CONTENT_LENGTH characters from standard input and writes them to standard output.

- **init**
  If QUERY_STRING is not set, reads the value of standard input and returns a SET statement that sets QUERY_STRING to this value. This can be used with both the GET and POST methods. A typical use is:
  ```
  eval 'cгiparse -init'
  ```
  This will set the QUERY_STRING environment variable, regardless of whether the GET or POST method was used.

cgiparse may be called multiple times in the same script when the GET method is used, but it should only be called once if the POST method is used. With the POST method, after standard input is read, the next cgiparse would find standard input empty and would hang.

- **sep** separator
  Specifies the string used to separate multiple values. If you are using the -value flag, the default separator is newline. If you are using the -form flag, the default separator is a comma (,).

- **prefix** prefix
  Used with -POST and -form, specifies the prefix to use when creating environment variable names. The default is "FORM_" if prefix is not specified for -POST and -form.

- **count**
  Used with -keywords, -form, and -value, returns a count of items related to these flags.

- **keywords**
  Returns the number of keywords

- **form**
  Returns the number of unique fields (multiple values are counted as one)

- **value** field-name
  Returns the number of values for field-name (if there is not a field named field-name, output is 0).

- **number**
  Used with -keywords, -form, and -value, returns the specified occurrence related to these flags.
-keywords
Returns the n’th keyword. (For example -2 -keywords outputs the second keyword.)

-form
Returns all the values of the n’th field. (For example -2 -form outputs all the values of the second field.)

-value field-name
Returns the n’th of the multiple values of field field-name. (For example -2 -value -whatsit outputs the second value of the whatsit field).

-quiet
Suppresses all error messages. (Non-zero exit status still indicates error.)

-fscp FileCodepage
The FileCodepage is the name of the file system codepage used in codepage conversion when processing text document bodies. When an unknown codepage is set, the default is used. The default is IBM-1047. Specifying -fscp on the httpd command overrides the default set in the server configuration file (httpd.conf).

-netcp NetCodepage
The NetCodepage is the network codepage name used in codepage conversion when processing text document bodies. When an unknown codepage is set, the default is used. The default is ISO8859-1 if the -netcp option is not specified. Specifying -netcp on the httpd command overrides the default set in the server configuration file (httpd.conf).

Note: The system iconv() service MUST support conversion between the pair of codepages specified as DefaultFsCp and DefaultNetCp.

Some acceptable values are:

<table>
<thead>
<tr>
<th>netcp</th>
<th>fscp</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISO8859-1</td>
<td>IBM-1047</td>
</tr>
<tr>
<td>IBM-932C</td>
<td>IBM-939</td>
</tr>
<tr>
<td>IBM-eucJC</td>
<td>IBM-939</td>
</tr>
</tbody>
</table>

-POST
Information from stdin is directly decoded and parsed into shell variables, QUERY_STRING is not used. -POST is equivalent to consecutive use of the -init and -form options.

Examples
The following examples ignore the fact that, in reality, QUERY_STRING is already set by the server. In the following examples, $ is the Bourne shell prompt.

- Keyword Search
  $ QUERY_STRING="is+2%2B2+really+four%3F"
  $ export QUERY_STRING
  $ cgiparse -keywords
  is
  2+2
  really
  four?
  $

- Parsing All Form Fields
Commands

```plaintext
$ export QUERY_STRING="name1=Value1&name2=Value2%3f+That%27s+right%21"
$ cgiparse -form
FORM_name1='Value1'; FORM_name2='Value2? That\'s right!'
$ eval 'cgiparse -form'
$ set | grep FORM
FORM_name1="Value1"
FORM_name2="Value2? That\'s right!"
$
```

- Extracting Only One Field Value

```plaintext
$ QUERY_STRING="name1=value1&name2=Second+value%3F+That\'s%27s+right%21"
$ cgiparse -value name1
value1
$ cgiparse -value name2
Second value? That\'s right!
$
```

Exit statuses

<table>
<thead>
<tr>
<th>Status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Success</td>
</tr>
<tr>
<td>1</td>
<td>Illegal command line</td>
</tr>
<tr>
<td>2</td>
<td>Environment variables not set correctly</td>
</tr>
<tr>
<td>3</td>
<td>Failed to get requested information (for example, there is no such field or QUERY_STRING contains keywords when form field values are requested).</td>
</tr>
</tbody>
</table>

**Note:** When you receive one of these error codes, you may receive additional informational messages, too. The message varies depending on the command issued.

**cgiutils command**

Use the cgiutils command in no-parse header programs to produce a full HTTP 1.0 response.

**Note:** If you want to supply your own no-parse header (nph) programs specifically to return your own return values, the name of the program must begin with `nph-`. This prevents the server header from overriding your return value with the standard server return value.

**Syntax**

```
cgiutils -Flag [Modifier]
```

If `Modifier` contains blanks, enclose it in quotes.

**Flags**

- `-version`
  - Returns version information.

- `-nodate`
  - Does not return the Date: header.

- `-noel`
  - Does not return a blank line after headers. This is useful if you want other MIME headers after the initial header lines.
-status nnn
Returns full HTTP response with status code nnn, instead of only a set of
HTTP headers. Do not use this flag if you only want the Expires: header.

-reason explanation
Specifies the reason line for the HTTP response. You can only use this flag with
the -status nnn flag.

-ct [type/subtype]
Specifies MIME Content-Type header. This example specifies a MIME content
type of text/html:

```
cgutils -ct text/html
```

If you omit the type/subtype, the MIME content type is set to the default
text/plain. This example sets the MIME content type to text/plain.

```
cgutils -ct
```

Specifies text/x-ssi-html to cause server side include processing of your
output stream.

-ce encoding
Specifies MIME Content-Encoding header. For example:

```
cgutils -ce x-compress
```

Specifies ebcDIC to force translation. All other codings prevent translation.

Note: Text encodings are expected in EBCDIC codepage IBM-1047 and will be
translated to ASCII codepage ISO8859-1 by the server. If encoding is not
explicitly specified, translation occurs if MIME type is not specified or if
type is text.

-cl language-code
Specifies MIME Content-Language header. For example:

```
cgutils -cl en_UK
```

-length nnn
Specifies MIME Content-Length header.

-expires Time-Spec
Specifies MIME Expires header. This flag specifies the time to live (the
expiration date of a document) in any combination of days, hours, minutes and
seconds. This is the length of time a document is considered valid. For
example:

```
cgutils -expires "2 days 12 hours"
```

The cgutils command adds the time you specify to the current Greenwich
Mean Time to determine the expiration date. The expiration date is put in the
Expires: header in the HTTP format.

Note: If more than one word is used for Time-Spec, you must enclose the
whole time specifier in double quotes.

-expires now
Produces an Expires: header that matches the Date: header. You should use this
to prevent caching of your output.

-uri URI
Specifies the Universal Resource Identifier (URI) for the returned document.
URI can be considered to be the same as URL.
-extra xxx: yyy

Specifies an extra header that cannot otherwise be specified for cgiutils.

Examples

- This example calculates the expiration date for the Expires: header.
  
cgiutils -expires "1 year 3 months 2 weeks 4 days 12 hours 30 mins"

- The following example specifies a status code and reason, and sets the Expire:
  header equal to the Date: header.
  
cgiutils -status 200 -reason "Virtual doc follows" -expires now

This might produce headers similar to these:

HTTP/1.0 200 Virtual doc follows
MIME-Version: 1.0
Date: Tue, 05 Jan 1996 03:43:46 GMT
Expires: Tue, 05 Jan 1996 03:43:46 GMT

The cgiutils command automatically produces the Server: header because it is
available in the CGI environment. The Date: field is also automatically generated
unless the -nodate flag is specified.

This would include a blank line after the output to mark the end of the MIME
header section. If you want to follow this with some more headers yourself, use
the -noel (NO-Empty-Line) flag as shown in the next example.

- If you do not want the blank line after the header line, use the -noel flag:
  
cgiutils -noel -expires "2 days" -nodate
  
Expires: Tue, 07 Jan 1996 03:43:46 GMT

htadm command

Use the htdadm command to control your IMWEBSRV server password files. Your
server can use password files to control access to your files. See Security
concepts on page 47 for information about controlling access to your server’s
resources. With htdadm you can add a user name to a password file, delete a user
from a password file, change a password for a user, verify a user’s password, and
create an empty password file. This command cannot be used to manage SAF
userids and passwords.

Note: When you are running htdadm as a separate line command and you are
outside the server, you must ensure that the NLSPATH run-time
environment variable is set. The NLSPATH variable should be set to:

NLSPATH=/usr/lpp/internet/%L/%N

Also, the following parameter should be set:

LANG=C
LANG=JA_JP

Syntax

htadm -Flag [Modifier]

Flags

-adduser password-file [user-name ["**"] password "real-name"]

Adds a user and password into the password file. If you enter the command
with only password-file, you are prompted for the other parameters.
password-file

The path and name of the password file to which you want to add the user.

user-name

The name of the user you want to add.

Do not use the characters $, &, ``, in the user name, because these characters are interpreted by the shells first. At present, the C, Bourne, and K shells exhibit tendencies to truncate the data or interpret the data incorrectly.

The command fails if there is already a user of the same name in the password file.

```
```

The password you want to define for the user you are adding, or the asterisk character (``*``). The asterisk allows you to enter information for real-name as part of the command but still be prompted for the password.

Note: Quotes are required when specifying the asterisk character. If you omit the quotes, OS/390 UNIX System Services will interpret the asterisk as a special OS/390 UNIX character.

Passwords can be up to 32 characters long. Use only alphabetic and numeric characters for the password; do not use special characters.

Note: Some browsers are unable to read and send passwords longer than eight characters. Because of this limitation, if you define a password longer than eight characters, the server recognizes either the complete password or just the first eight characters of the password as valid.

```
```

A comment or name you want to use to identify the user name you are adding. Whatever you enter will be written into the password file.

Note: Quotes are required if the real-name contains more than one word, for example, "Clark Kent".

-deluser password-file [user-name]

Deletes a user from the password file. If you enter the command with only password-file, you are prompted for the user-name parameter.

password-file

The path and name of the password file from which you want to delete a user.

user-name

The name of the user you want to delete. The command fails if the user name you specify does not exist in the password file.

-passwd password-file [user-name [password]]

Changes the password for a user name already defined in the password file. If you enter the command with only password-file, you are prompted for the other parameters.

password-file

The path and name of the password file that contains the user name whose password you want to change.
**Commands**

`user-name`
The user name whose password you want to change. The command fails if the user name you specify does not exist in the password file.

`password`
The new password you want to define for the user name.

Passwords can be up to 32 characters long. Use only alphabetic and numeric characters for the password; do not use special characters.

**Note:** Some browsers are unable to read and send passwords longer than eight characters. Because of this limitation, if you define a password longer than eight characters, the server recognizes either the complete password or just the first eight characters of the password as valid.

`-check` `password-file [user-name [password]]`
Verifies the password for a user name already defined in the password file and lets you know if it is correct or not. If you enter the command with only `password-file` you are prompted for the other parameters.

`password-file`
The path and name of the password file that contains the user name whose password you want to verify.

`user-name`
The user name whose password you want to verify. The command fails if the user name you specify does not exist in the password file.

`password`
The password that you want to verify. If the password you enter is the one defined for the user name, the command writes `Correct` to standard output and completes with a 0 return code. If the password you enter is not the one defined for the user name, the command writes `Incorrect` to standard output.

`-create` `password-file`
Create an empty password file.

`password-file`
The path and name of the password file that you want to create.

`-stash` `filename password`
Create or change the password stash file. This file contains the encrypted password the server uses to log into an LDAP server when reading access control lists or configuration directives. When accessing the LDAP server, the HTTP Server will log in using its Distinguished Name (from the ServerDN subdirective of the LDAPInfo directive) and the decoded password from the stash file.

**Attention:** Although the file is encrypted, use file system security measures to limit read/write access to this stash file.

`filename`
The directory and name of the stashed password file.

`password-file`
The password used to access the LDAP server. This password will be encrypted within the file.
Examples

- To add a user to a password file:
  
  ```bash
  htdadm -adduser /usr/lpp/internet/server_root/heroes.pwd clark superman "Clark Kent"
  ```

- To delete a user from a password file:
  
  ```bash
  htdadm -deluser /usr/lpp/internet/server_root/heroes.pwd clark
  ```

- To create a stashed password file for LDAP access:
  
  ```bash
  htdadm -stash ldap_server_pwd secretpassword
  ```

  The example creates a file named `ldap_server_pwd` containing the encrypted string that represents the password `secretpassword`.

httpd command

Use the httpd command to start the server.

You can set all these flags (except -r) using the directives in the server configuration file.

Syntax

```bash
httpd [-Flag [-Flag [-Flag..]]]
```

Flags

To turn tracing on, see [Tracing flags on page 251].

- **-B bounce**

  Starts the Web server with the sockets setting SO_REUSEADDR on. This will avoid a potential delay while sockets used by a previous invocation of the Web server are closed. This is especially useful if you have automation software that recognizes the Web server termination and needs to start a replacement Web server as quickly as possible.

  By default, bounce is set on. There is no option on the httpd command to override the default of bounce on. To set this flag off, you must set the Bounce directive off in the httpd.conf file and omit -B from the Web server startup command.

  If bounce is set off, the Web server will retry the bind to its listen ports for up to two minutes to allow previously used sockets to close. If the Web server is started quickly after a previous Web server shutdown and the startup option -B is not used, the listen ports may not be available. This can result in a delay in the availability of the Web server.

- **-drainsoc**

  Causes the server to drain TCP/IP sockets of all input data before closing the socket. This means that all input data still outstanding on the socket after the server has processed all the data it expects will be read and discarded. If web server tracing of TCP/IP communication is also being performed, any extra data read will be traced. This can be especially helpful in diagnosing potential browser software problems.

- **-fscp FileCodepage**

  DefaultFsCp. Specifies the name of the file system codepage used in codepage conversion when processing text document bodies. When an unknown
Commands

codepage is set, the default is used. The default is IBM-1047. Specifying `-scp` on the httpd command overrides the default set in the server configuration file (httpd.conf).

`-gc_only`
Only does garbage collection and then exits. This flag is used only for caching proxy servers.

`-h` HostName. Host names or IP addresses can be used on the template.

`-netcp NetCodepage`
DefaultNetCp. Specifies the network codepage name used in codepage conversion when decoding the URL-encoded string. When an unknown codepage is set, the default is used. The default is ISO8859-1 if the `-netcp` option is not specified. Specifying `-netcp` on the httpd command overrides the default set in the server configuration file (httpd.conf).

`-nobg`
Runs the server as a foreground process. Do not run the server as a background process.

`-nosec`
Specifies the flag to use to force a base server. Security loads are bypassed.

`-nosmf`
Turns SMF recording off.

`-nosnmp`
Turns SNMP support off.

`-p port-number`
Listens on this port number. The default port number is 80. This flag overrides the Port directive specified in the configuration file.

`-r configuration-file`
Start the server with the specified server configuration file. You must use this flag if you want to start the server with a server configuration file other than the default /etc/httpd.conf.

**Note:** If you specify the `-r` option and either the normalmode or sslmode option, you must specify normalmode or sslmode before `-r`.

`-restart`
Restarts a server that is currently running.

`-smf`
Turns SMF recording on.

`-snmp`
Turns SNMP support on.

`-ss CONNECTION_POOL`
Specify where to place a shared memory segment for a connection pool. Explicitly specify a starting point for a shared memory segment and the size of the segment. Add this command only under the direction of IBM support personnel. IBM recommends the use of defaults: either do not use this command at all or explicitly code the defaults. This command is not valid without parameters.

```
-ss CONNECTION_POOL virtual_address segment_size
```

`virtual_address`
Code 0.
Commands

segment_size
Default=20 MB.

Refer to “Chapter 1. Planning for installation” on page 3 for more information on OS/390 UNIX system services parameters that may need to be set, especially, the following:
- IPCSHMSPAGES
- IPCSHMMPPAGES
- IPCSHMNSEGS

See OS/390 UNIX System Services Planning for more information about establishing the appropriate segment size.

-sslmode [on | off]
  For a secure server, turns on the SSL protocol.

-sslport [port]
  For a secure server, sets the port used for the SSL protocol.

-version
  Returns the version number of the httpd executable and then exits.

Tracing flags

The level of tracing provided is -v trace, first level; -vv trace, second level; -mtv, third level; and -debug for maximum tracing. To trace caching, use the -vc option.

Note: We recommend that you do not turn tracing on unless instructed to do so by IBM support personnel. IBM support personnel can give you guidance on the most appropriate trace for your problem.

-v  Verbose. Turns on first level debugging messages.

-vv  Very Verbose. Turns on second level debugging messages.

-mtv  Much Too Verbose. Turns on third level debugging messages.

-debug [module_name]
  Turns on debugging for all modules or a specific module. -debug with no module name specified turns on debug for all modules.

-nodebug [module_name]
  Turns off tracing for all modules or a specific module. -nodebug with no module name specified turns off debug for all modules.

-vc  Verbose cache. Turns on cache tracing messages.

Module names

On the -debug and -nodebug options, you can enter the following values for module_name:

GROUP_PROXY
GROUP_JAVELIN
MODULE_BAGS
MODULE_CACHEACCELERATOR
MODULE_CGI
MODULE_CONFIG
MODULE_CONNECTIONS
MODULE_CREATORS
Signal handling

Following is an explanation of how the server behaves when sent the SIGKILL, SIGTERM and -SIGHUP signals.

**SIGKILL**

This causes the server to terminate immediately. This can be accomplished by sending the SIGKILL signal to the httpd process using the OS/390 UNIX shell command kill, the Web server wwwcmd command, or by cancelling the job using the OS/390 operator console CANCEL command.
SIGTERM
This causes the server to stop and exit immediately. This can be accomplished by sending the SIGTERM signal to the httpd process using the OS/390 UNIX shell command kill or the Web server wwwcmd command.

—SIGHUP
Restart a running httpd. This causes httpd to stop accepting new requests, complete current requests, and, if there are no errors, reload the configuration file and resume processing. This can be accomplished by sending the —SIGHUP signal to the httpd process using the OS/390 UNIX shell command kill or the Web server wwwcmd command. If there are errors, you must fix the configuration file and try again.

The authority to send SIGTERM, SIGKILL, and —SIGHUP signal to httpd process is the same user ID as WEBSRV or superuser. You need to know the WEBSRV process ID in order to use the OS/390 UNIX shell command, kill, or the Web server wwwcmd command to send the signal. The process ID is defined on the PidFile directive in the configuration file. Two instances of the server MUST NOT be run at the same time using the same PidFile, log files, and proxy cache to avoid corruption of those files. It is the webmasters responsibility to ensure this. For information about the kill command, see the OS/390 UNIX System Services Command Reference.

Examples

- To start the server at port 8080, using the configuration file /usr/etc/httpd.conf instead of the default, /etc/httpd.conf, enter:
  ```
  httpd -p -8080 -r /usr/etc/httpd.conf
  ```
  If the Port directive is given in the configuration file, the -p flag is not required. However, the -p flag can be used to override the value set in the configuration file.

- To start the server using the default configuration file /etc/httpd.conf with verbose tracing:
  ```
  httpd -vv
  ```

IMWHTTPD program
Use the IMWEBSRV PROC to start the server. The program is IMWHTTPD. IMWHTTPD requires either a rule file (default or -r) or a directory to export.

You can set all these flags (except -r) using the directives in the server configuration file.

It is common practice to create a file named README containing instructions or notices to be read by anyone new to the directory. IMWHTTPD, by default, imbeds any README file in the hypertext version of a directory. The README file instructions can also be set with the DirReadme configuration directive.

Syntax

```
Commands

```c
// ICSPARM ==> HTTP Server parameters
// # Standalone HTTPD
// ICSPARM='p 8080
// # WLM Queue Manager
// ICSPARM='-SN WEBSN1 -p 8080
// # WLM ApplEnv Queue Server
// ICSPARM='-SN WEBSN1 -AE WEBHTML'
//
// HTTP Server Parameters:
// -SN # WLM - subsystem name
// -AE # WLM - Application Environment
// -fscp nnn # File system codepage - EBCDIC
// -netcp nnn # net code page - ASCII
// -gc_only # clean cache & exit (garbage collect)
// -normalmode
// -p nnnn # use port nnn (default 80)
// -sslmode
// -sslport nnnn # use port nnn (default 443)
// -nosec # no security
// -nosmf # no smf processing on
// -smf # smf processing on
// -r /etc/httpd.conf # use rule file xxxx
// -restart
// -v # trace to stderr
// -vv # trace to stderr
// -vc # cache trace to stderr
// -version # show version and exit
//
//*********************************************************************
//WEBSRV EXEC PGM=IMWHTTPD,REGION=0K,TIME=NOLIMIT,
// PARM=('&LEPARM/&ICSPARM')
//*********************************************************************
```

You should note that JCL only allows 100 characters in the PARM=' ' statement.

Parameters

**LE run-time option**
The run-time options allow you to control certain aspects of your program's processing. For information about run-time options, see the *LE Programming Guide*. There are specific environment variables needed to start the server.

**Environment Variables**
The LE run-time environment variables are:

**NLSPATH**
This variable controls where the message catalog is located.
LANG
This variable controls where the message catalog is located.

PATH
This variable is the search path available to CGI programs. Set the
default path for CGI programs.

TZ
This variable must be set to use local time in log files and server-side
includes.

SHELL
This variable is the name of the default shell program to be run.

LIBPATH
This variable determines the search path for loading DLLs.

The following LE run-time option is used when starting the server:

ENVAR
The ENVAR option sets initial values for specified environment variables that
the server is started with. Using ENVAR, you can pass switches or tagged
information using standard OS/390 UNIX functions. You may set additional
environment variables using the _CEE_ENVFILE ENVAR option. The default
_CEE_ENVFILE shipped with the server is /etc/httpd.envvars, which is
compiled into the IMWHTTPD program. You can use the _CEE_ENVFILE
ENVAR option to override this default.

The format of this option is:

ENVAR("_CEE_ENVFILE=filename")

filename specifies the file containing the LE run-time environment variables.
The following is an example of the file containing the environment variables
that is shipped with the server under
/usr/lpp/internet/server_root/samples/config:

SHELL=/bin/sh
PATH=/bin:.:/usr/lpp/internet/bin
LANG=c
NLSPATH=/usr/lib/nls/msg/%L/%N:/usr/lpp/internet/%N
TZ=EST5EDT
LIBPATH=/usr/lpp/internet/bin:/usr/lpp/internet/sbin

To make the above example reflect Japanese, specify the following:

LANG=Ja_JP
LC_ALL=Ja_JP.IBM-939
NLSPATH=/usr/lib/nls/msg/%L/%N:/usr/lpp/internet/%N

Setting server environment variables
Server environment variables can be set from JCL or the OMVS shell. The
following steps provide instruction on how to do this.

1. Set variables:

From JCL: LE Runtime Parameter ENVAR For example:

// EXEC PGM=IMWHTTPD,
// PARM=(ENVAR("MVSDS_CFG=/etc/mycfg"))

From OMVS shell:

export MVSDS_CFG=/etc/mycfg
2. LE checks the environment for the variable _CEE_ENVFILE. If it is defined, it
   MUST contain the name of a file containing additional environment variables to
   add to the environment. If it is not defined, IMWHTTPD sets it to
   /etc/httpd.envvars.

3. LE adds each variable defined in the file identified in step 2 to the server’s
   environment. A sample file is shipped with the Web server.

The example shows how to set the variables to preload MVS data sets using the
MVS data set configuration file.

-**AE**

   ApplicationEnvironmentName. Identifies the workload management queue.
   This is only valid on a server address space.

-**B** bounce

   The server normally does not bind to its listen ports with the
   SO_REUSEADDR socket option. This helps to prevent running multiple
   instances of the server with the same Pid, log, and proxy cache files. When a
   server is shut down or terminates abnormally, there may be sockets remaining
   in TIMED_WAIT state in the TCP/IP stack. The Web server retries the bind to
   its listen ports for up to two minutes to allow previously used sockets to close.

   If you know that the prior instance of the server has terminated, you can use
   the -B flag to set SO_REUSEADDR on to the servers listen ports before binding
   to them. This avoids the TIMED_WAIT delay. This flag is especially useful if
   you have automation software that recognizes Web server termination and
   needs to start a replacement Web server as quickly as possible.

-**drainsoc**

   Causes the server to drain TCP/IP sockets of all input data before closing the
   socket. This means that all input data still outstanding on the socket after the
   server has processed all the data it expects, will be read and discarded. If web
   server tracing of TCP/IP communications is also being performed, any extra
   data read will be traced. This can be especially helpful in diagnosing potential
   browser software problems.

-**fscp** FileCodepage

   The FileCodepage is the name of the file system codepage used in codepage
   conversion when processing text document bodies. When an unknown
   codepage is set, the default is used. The default is IBM-1047. Specifying -**fscp**
   on the httpd command overrides the default set in the server configuration file
   (httpd.conf).

-**netcp** NetCodepage

   The NetCodepage is the network codepage name used in codepage conversion
   when processing text document bodies. When an unknown codepage is set, the
   default is used. The default is ISO8859-1 if the -netcp option is not specified.
   Specifying -**netcp** on the httpd command overrides the default set in the server
   configuration file (httpd.conf).

   **Note:** The system iconv() service MUST support conversion between the pair
   of codepages specified as DefaultFsCp and DefaultNetCp.

   Some acceptable values are:

   netcp  fscp
   ISO8859-1  IBM-1047
   IBM-932C  IBM-939
   IBM-eucJC IBM-939
-gc_only
Only does garbage collection and then exits. This flag is used only for caching proxy servers.

-h HostName. Host names or IP number addresses can be used on the template.

-lb
Listen backlog. Use this parameter to override the default parameter on listen() for master and SSL sockets. If the parameter is greater than SOMAXCONN, stack uses SOMAXCONN instead. The Internet Connection Secure Server 2.1 default is 16; for the Internet Connection Secure Server 2.2 and later versions of the Web server, the default is 128.

-nobg
Runs the server as a background process. Do not run the server as a background process.

-nosec
Specifies the flag to use to force a base server. Security loads are bypassed.

-nosmf
Turns SMF recording off.

-nosnmp
Turns SNMP support off.

-p port-number
Listens on this port number. The default port number is 80. This flag overrides the Port directive specified in the configuration file.

-r configuration-file
Specifies the file to use as the configuration file. You must use this flag if you want to start the server with a configuration file other than the default configuration file.

Note: If you specify the -r option and either the normalmode or sslmode option, you must specify normalmode or sslmode before -r.

-restart
Restarts a server that is currently running. The httpd command gets the process number of the server that is running from the PidFile and sends the server process ID a —SIGHUP signal. The server process after intercepting the —SIGHUP signal reloads its configuration file, reopens its log files, and resumes servicing client request. Two instances of the server MUST not be run at the same time using the same PidFile, log files, and proxy cache to avoid corruption.

Because the http daemon must read the configuration file the server is currently using in order to access the PidFile, you must specify the same configuration file when restarting. If you used the -r flag and a specific configuration file when you started the server, then you must specify this flag and same file with -restart.

-smf
Turns SMF recording on.

-SN
SubsysName. In a workload management environment, each scaleable server environment needs a unique name.
Commands

-snmp

Turns SNMP support on.

-sslmode [on | off]

For a secure server, turns on the SSL protocol.

-sslport [port]

For a secure server, sets the port used for the SSL protocol.

-version

Returns the version number of the httpd executable and then exits.

Tracing options

The level of tracing provided is -v trace, first level; -vv trace, second level; -mtv, third level; and -debug for maximum tracing. To trace caching, use the -vc option.

Note: We recommend that you do not turn tracing on unless instructed to do so by IBM support personnel. IBM support personnel can give you guidance on the most appropriate trace for your problem.

-v Verbose. Turns on first level debugging messages.

-vv Very Verbose. Turns on second level debugging messages.

-mtv Much Too Verbose. Turns on third level debugging messages.

-debug [module_name]

Turns on debugging for all modules or a specific module. -debug with no module name specified turns on debug for all modules.

-nodebug [module_name]

Turns off tracing for all modules or a specific module. -nodebug with no module name specified turns off debug for all modules.

-vc

Verbose cache. Turns on cache tracing messages.

Module names: On the -debug and -nodebug options, you can enter the following values for module_name:

GROUP_PROXY
GROUP_JAVELIN
MODULE_BAGS
MODULE_CACHEACCELERATOR
MODULE_CGI
MODULE_CONFIG
MODULE_CONNECTIONS
MODULE_CREATORS
MODULE_DIRBRW
MODULE_DNS
MODULE_ERRORS
MODULE_FORMAT
MODULE_FCGI
MODULE_FTP
MODULE_GC
MODULE_GOPHER
MODULE_HASH
MODULE_HTML
MODULE_HTTP
Signal handling

Following is an explanation of how the server behaves when sent the SIGTERM and —SIGHUP signals.

SIGKILL
This causes the server to terminate immediately. This can be accomplished by sending the SIGKILL signal to the httpd process using the OS/390 UNIX shell command kill, the Web server wwwcmd command, or by cancelling the job using the OS/390 operator console CANCEL command.

SIGTERM
This causes the server to stop and exit immediately. This can be accomplished by sending the SIGTERM signal to the httpd process using the OS/390 UNIX shell command kill or the Web server wwwcmd command.

—SIGHUP
Restart a running httpd. This causes httpd to stop accepting new requests, complete current requests, and, if there are no errors, reload the configuration file and resume processing. This can be accomplished by sending the —SIGHUP signal to the httpd process using the OS/390 UNIX shell command kill or the Web server wwwcmd command. If there are errors, you must fix the configuration file and try again.
The authority to send SIGTERM, SIGKILL, and —SIGHUP signals to the httpd process is the same user ID as WEBSRV or superuser. You need to know the WEBSRV process ID in order to use the OS/390 UNIX shell command, kill, or the Web server wwwcmd command to send the signal. The process ID is defined on the PidFile directive in the configuration file. For information about the kill command, see the OS/390 UNIX System Services Command Reference.

Examples

- To start the server on port 8080, using the /usr/etc/httpd.conf configuration file instead of the default, /etc/httpd.conf, the following needs to be added to your PROC:

  If the Port directive is given in the configuration file, the -p flag is not required.

  The -p flag can be used to override the value set in the configuration file.

- To start the server using the default configuration file /etc/httpd.conf with verbose tracing:

  // PARM='/-vv'

- To start IMWHTTPD using its default configuration file /etc/httpd.conf, from the MVS console, enter:

  S IMWEBSRV

IMWIWM PROC (workload management)

The workload management PROC should be placed in your system PROCLIB. This PROC correlates with any query server automatically started from WLM. The passed parameters are set with WLM panels as shown in the "Modify an Application Environment" panel on page 111. Ensure that this PROC is properly recognized by your system security.

```c
//IMWIWM PROC IWMSN=,IWMAE=
//******************************************************
/*/ ICSPARM ==> HTTP Server parameters
/*/ # WLM ApplEnv Queue Server
/*/ ICSPARM="#-SN WEBSN1 -AE WEBHTML'
/*/ HTTP Server Parameters:
/*/ -SN # WLM - subsystem name
/*/ -AE # WLM - Application Environment
/*/ ICSPARM="#-SN-WEBSN1 -AE WEBHTML'
//WEBSVR EXEC PROC=IMWEBSRV,REGION=0K,TIME=NOLIMIT,
// ICSPARM=&BQQ&SN&IWMSN&AE&IWMAE&BQQ
```

Figure 4. IMWIWM PROC Example

Parameters

-AE

ApplicationEnvironmentName. Identifies the workload management queue. This is only valid on a server address space.

-SN

SubsysName. In a workload management environment, each scalable server environment needs a unique name.
**wwwcmd command**

Use the wwwcmd command to stop, kill, or restart the server.

**Syntax**

```
wwwcmd [parameters] action
```

**Parameters**

Parameters are optional and must be one of the following:

- `-p pid`
  Specifies the process ID of the server. (For example: `-p 1111111`)

- `-f pid_file`
  Specifies the process ID file of the server.

- `-r rule_file`
  Specifies the rule file to determine the location of the process ID file of the server. The program looks for two directives `ServerRoot` and `PidFile`. If `PidFile` is found, its value is used. If `PidFile` is not found, but `ServerRoot` is found, it is assumed that a file named `httpd-pid` exists in `ServerRoot`. If neither directive is found, the program uses `/etc/httpd.conf` as the rule file to search. The default used by the program as the rule file is `/etc/httpd.conf`.

**Action**

One action must be specified to the wwwcmd command from the following list:

- `-stop`
  This action stops the server gracefully. The server stops accepting any new connections and allows all existing connections to end gracefully before terminating the server. The signal, SIGTERM, gets sent to the server.

- `-kill`
  This action stops the server immediately. The server terminates at once, with no clean up or processing of existing connections. It is not recommended that you use this action. You should attempt `-stop` before entering this command. The signal, KILL, gets sent to the server.

- `-restart`
  This action restarts the server. The server stops processing any new connections and allows all existing connections to end gracefully before re-reading the current rule file. The signal —SIGHUP gets sent to the server.

**Note:** You must have permission to send the requested signal to the server. To have permission, you must either log onto the user ID of the server or be the root. IBM recommends you enter this command as root.

---

**OS/390 MODIFY console command**

You can use the OS/390 MODIFY command to:

- Restart the Web server
- Display configuration information and statistics for the Web server
- Turn System Management Facilities (SMF) on or off
- Turn tracing on or off

For more information about the OS/390 MODIFY command, see the OS/390 System Command Reference.
Syntax

F [server_jobname],APPL=parameters

Parameters

-? Displays the parameters available when using the MODIFY command.
-?? Displays the parameters available on the MODIFY command.

- Restart parameter:

-restart
Reverts the server after configuration changes.

- Configuration information and statistics:

-d config
Displays configuration information.
-d stats
Displays server statistics.
-version
Displays the version of the Web server.

- SMF parameters:

-smf
Turns SMF recording ON for performance and configuration record data.
-smf perf
Turns SMF recording ON for performance record data only.
-smf config
Turns SMF recording ON for configuration record data only.
-? smf
Displays information about the -smf parameter on the MODIFY command.

-nosmf
Turns SMF recording OFF for performance and configuration record data.
-nosmf perf
Turns SMF recording OFF for performance record data only. This parameter does not affect configuration data recording by SMF.
-nosmf config
Turns SMF recording OFF for configuration record data only. This parameter does not affect the performance data recording by SMF.
-? nosmf
Displays information about the -nosmf parameter on the MODIFY command.

Tracing parameters

The level of tracing provided is -v trace, first level; -vv trace, second level; -mtv, third level; and -debug for maximum tracing. To trace caching, use the -vc option.

Note: We recommend that you do not turn tracing on unless instructed to do so by IBM support personnel. IBM support personnel can give you guidance on the most appropriate trace for your problem.

-v Verbose. Turns on first level tracing to stderr.
-vv
Very Verbose. Turns on second level tracing to stderr.

-mtv
Much Too Verbose. Turns on third level tracing to stderr.

-debug [module_name]
Turns on debugging for all modules or a specific module. -debug with no
module name specified turns on debug for all modules.

-? debug
Displays information about the -debug parameter on the MODIFY command.

-nodebug [module_name]
 Turns off tracing for all modules or a specific module. -nodebug with no
module name specified turns off debug for all modules.

-? nodebug
Displays information about the -nodebug parameter on the MODIFY
command.

-vc
Verbose cache. Turns on cache tracing messages.

Module names
On the -debug and -nodebug options, you can enter the following values for
module_name:

GROUP_PROXY
GROUP_JAVELIN
MODULE_BAGS
MODULE_CACHEACCELERATOR
MODULE/cgi
MODULECONFIG
MODULE_CONNECTIONS
MODULE_CREATORS
MODULE_DIRBRW
MODULE_DNS
MODULE_ERRORS
MODULE_FORMAT
MODULE_FCGI
MODULE_FTP
MODULE_GC
MODULE_GOPHER
MODULE_HASH
MODULE_HTML
MODULE_HTTP
MODULE_ICAPI
MODULE_IF
MODULE_JAVELINBASE
MODULE_JAVELINCFC
MODULE_JAVELINPICS
MODULE_JAVELINWBI
MODULE_LEX
MODULE_LOCALCACHE
MODULE_LOG
MODULE_MEMPOOL
MODULE_MPOOL
MODULE_NETMON
Examples

To turn SMF recording off, turn debugging for configuration-related functions on, and restart the Web server, enter:

F IMWEBSRV,APPL=-nosmf -debug config -restart

The following messages are displayed:

IMW03514I SMF recording has been disabled for all record types
IMW03504I Debug has been enabled for module, "config"
IMW03537I SA 1207959570 0.0.0.0:8480 * * RESTARTING
IMW03538I SA 1207959570 0.0.0.0:8480 * * RESTART SUCCESSFUL

To display information on the Web server version, configuration, and statistics, enter:

F IMWEBSRV,APPL=-version -d config -d stats

The following example shows messages displayed. For more information on the statistics in IMW03501I, see "Web server activity statistics" on page 114.

IMW03516I Version:IBM HTTP Server - North American Edition V 5R2M0
IMW03501I Config: Hostname: host52.raleigh.ibm.com, Port: 80, SSL Port:4480,
Server root:/usr/lpp/internet/server_root
Tracing all modules.
SMF recording is currently disabled.
IMW3502I Stats: Threads running: 39, Threads idle: 36,
Requests: 19, Bytes rcvd: 15417, Bytes sent: 93109,
Actv In Conns: 3, Actv Out Conns: 0.
Connections since last SMF: 11,
DNS Max: 0.000745, DNS Min: 0.000005, DNS Avg: 0.000038,
Service Plugins Max: 0.035242, Service Plugins Min: 0.019390
Service Plugins Avg: 0.027316,
CGI Max: 15.142639, CGI Min: 12.545370, CGI Avg: 13.844004,
SSL Handshake Max: 199.438877, SSL Handshake Min: 0.000061,
SSL Handshake Avg: 6.657594,
Proxy Response Max: 12.966245, Proxy Response Min: 1.120083,
Proxy Response Avg: 5.128365
Non-SSL Waiting Threads: 14, SSL Waiting Threads: 16,
Async I/O Waiting Threads: 0, Msg Queue Waiting Threads: 0
There are several OS/390 commands that are useful when you are working in a Workload Management (WLM) environment. This section describes those commands. For more information on WLM commands, see *OS/390 Programming: Workload Management Services*.

### WLM mode

If your mode is GOAL, WLM is up and running to manage your system. If your mode is COMPAT, WLM is not activated to manage your system.

**Note:** You must be in GOAL mode to use the Web server in Scalable Server mode.

To change your mode to GOAL or COMPAT, enter:

```
F WLM,MODE=GOAL
F WLM,MODE=COMPAT
```

**Examples:**

```
F WLM,MODE=GOAL
IWM007I SYSTEM MVS157 NOW IN WORKLOAD MANAGEMENT GOAL MODE

F WLM,MODE=COMPAT
SET IPS=00
SET ICS=00
IWM007I SYSTEM MVS157 NOW IN WORKLOAD MANAGEMENT COMPATIBILITY MODE
IEEE252I MEMBER IEAIPS00 FOUND IN SYS1.PARMLIB
IEEE536I IPS 00 NOW IN EFFECT
IEEE252I MEMBER IEAICS00 FOUND IN SYS1.PARMLIB
IEEE536I ICS 00 NOW IN EFFECT
```

### Application environments

To check WLM application environments, enter:

```
D WLM,APPLENV=application_environment_name
```

**Example:**

```
D WLM,APPLENV=WEBHTML
IWM029I 18.42.09 WLM DISPLAY 143

APPLICATION ENVIRONMENT NAME STATE STATE DATA
WEBHTML AVAILABLE
ATTRIBUTES: PROC=WEBWLM01 SUBSYSTEM TYPE: IWEB
```

To change the status of an Application Environment, enter:

```
VARY WLM,APPLENV=WEBHTML,RESUME
```

**Example:**

```
VARY WLM,APPLENV=WEBHTML,RESUME
IWM032I VARY RESUME FOR WEBHTML COMPLETED
```

To activate changes made to ApplEnv panels within WLM when the ApplEnv has not been stopped, enter:

```
VARY WLM,APPLENV=WEBHTML,REFRESH
```

**Example:**

```
VARY WLM,APPLENV=WEBHTML,REFRESH
IWM032I VARY REFRESH FOR WEBHTML COMPLETED
```
Commands

**WLM systems**

To check WLM systems, enter:

```
D WLM,SYSTEMS
```

**Example:**

```
D WLM,SYSTEMS
IWM025I 18.42.53 WLM DISPLAY 145
ACTIVE WORKLOAD MANAGEMENT SERVICE POLICY NAME: VICOM1
DESCRIPTION: Default VICOM policy with ResGrp
RELATED SERVICE DEFINITION NAME: COEFFS
WLM VERSION LEVEL: LEVEL003
*SYSNAME* *MODE* *POLICY* *WORKLOAD MANAGEMENT STATUS*
MVS157 GOAL VICOM1 ACTIVE
```
# Appendix C. Configuration directives

This section describes each server configuration directive and subdirective. If you are editing the Web server configuration file directly, use this section as a reference.

## Overview of directives

### Quick reference

The directive descriptions are grouped according to function. Within each group, the directives are in alphabetical order. For an alphabetical list of directives, see the Directives entry in the index.

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
<th>Page</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access control</td>
<td>Specify default protection setup for requests that match a template</td>
<td>275</td>
<td>DefProt</td>
</tr>
<tr>
<td></td>
<td>Activate protection setup for requests that match a template</td>
<td>278</td>
<td>Protect</td>
</tr>
<tr>
<td></td>
<td>Define a named protection setup within the configuration file</td>
<td>283</td>
<td>Protection</td>
</tr>
<tr>
<td></td>
<td></td>
<td>288</td>
<td>Protection subdirectives</td>
</tr>
<tr>
<td></td>
<td></td>
<td>287</td>
<td>SSL client authentication subdirectives</td>
</tr>
<tr>
<td>Basic</td>
<td>Specify if the server binds to one or all IP addresses</td>
<td>288</td>
<td>BindSpecific</td>
</tr>
<tr>
<td></td>
<td>Specify the default start option for the sockets setting</td>
<td>288</td>
<td>DNS-Lookup</td>
</tr>
<tr>
<td></td>
<td>Specify whether you want to look up host names of clients</td>
<td>289</td>
<td>DNS-Lookup</td>
</tr>
<tr>
<td></td>
<td>Specify the fully qualified domain name or IP address for the server</td>
<td>284</td>
<td>HostName</td>
</tr>
<tr>
<td></td>
<td>Specify whether server-side includes will be dynamically imbedded</td>
<td>290</td>
<td>imbeds</td>
</tr>
<tr>
<td></td>
<td>Specify an alternate directory installation path</td>
<td>291</td>
<td>InstallPath</td>
</tr>
<tr>
<td></td>
<td>Specify the location of the process ID file</td>
<td>291</td>
<td>PidFile</td>
</tr>
<tr>
<td></td>
<td>Specify the port on which you want the server to listen for requests</td>
<td>292</td>
<td>Port</td>
</tr>
<tr>
<td></td>
<td>Customize ABEND recovery performed by the Web server</td>
<td>292</td>
<td>Recovery</td>
</tr>
<tr>
<td></td>
<td>Specify the absolute directory path of the server</td>
<td>292</td>
<td>ServerRoot</td>
</tr>
<tr>
<td></td>
<td>Specify the default surrogate user ID</td>
<td>292</td>
<td>UserId</td>
</tr>
</tbody>
</table>

267
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
<th>Page</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Codepages</strong></td>
<td>Specify the codepages the server receives from the client</td>
<td>294</td>
<td>DefaultNetCp</td>
</tr>
<tr>
<td></td>
<td>Specify the codepage environment variable for the server</td>
<td>294</td>
<td>DefaultFsCp</td>
</tr>
<tr>
<td><strong>Directories and Welcome Page</strong></td>
<td>Specify the icon URL used to align the heading of directory listings</td>
<td>293</td>
<td>AddBlankIcon</td>
</tr>
<tr>
<td></td>
<td>Specify the icon URL for directories on directory listings</td>
<td>293</td>
<td>AddDirIcon</td>
</tr>
<tr>
<td></td>
<td>Bind an icon to a MIME content-type or encoding-type</td>
<td>296</td>
<td>AddIcon</td>
</tr>
<tr>
<td></td>
<td>Specify the icon URL for a parent directory on directory listings</td>
<td>297</td>
<td>AddParentIcon</td>
</tr>
<tr>
<td></td>
<td>Specify the icon URL for unknown file types on directory listings</td>
<td>297</td>
<td>AddUnknownIcon</td>
</tr>
<tr>
<td></td>
<td>Specify if a welcome file is returned for all directory requests</td>
<td>298</td>
<td>AlwaysWelcome</td>
</tr>
<tr>
<td><strong>Control directory listings</strong></td>
<td>Control directory listings</td>
<td>298</td>
<td>DirAccess</td>
</tr>
<tr>
<td></td>
<td>Control directory README files</td>
<td>299</td>
<td>DirReadme</td>
</tr>
<tr>
<td></td>
<td>Use brackets around alternative text on directory listings</td>
<td>299</td>
<td>DirShowBrackets</td>
</tr>
<tr>
<td></td>
<td>Show byte count for small files on directory listings</td>
<td>299</td>
<td>DirShowBytes</td>
</tr>
<tr>
<td></td>
<td>Use case when sorting files on directory listings</td>
<td>300</td>
<td>DirShowCase</td>
</tr>
<tr>
<td></td>
<td>Show date last modified on directory listings</td>
<td>300</td>
<td>DirShowDate</td>
</tr>
<tr>
<td></td>
<td>Show descriptions for files on directory listings</td>
<td>300</td>
<td>DirShowDescription</td>
</tr>
<tr>
<td></td>
<td>Show the group ID of files on directory listings</td>
<td>300</td>
<td>DirShowGroup</td>
</tr>
<tr>
<td></td>
<td>Show hidden files on directory listings</td>
<td>301</td>
<td>DirShowHidden</td>
</tr>
<tr>
<td></td>
<td>Show icons in directory listings</td>
<td>301</td>
<td>DirShowIcons</td>
</tr>
<tr>
<td></td>
<td>Set the maximum description length on directory listings</td>
<td>301</td>
<td>DirShowMaxDescrLength</td>
</tr>
<tr>
<td></td>
<td>Set the maximum length for file names on directory listings</td>
<td>301</td>
<td>DirShowMaxLength</td>
</tr>
<tr>
<td></td>
<td>Set the minimum length for file names on directory listings</td>
<td>301</td>
<td>DirShowMinLength</td>
</tr>
<tr>
<td></td>
<td>Show file permissions on directory listings</td>
<td>302</td>
<td>DirShowMode</td>
</tr>
<tr>
<td></td>
<td>Show file owner on directory listings</td>
<td>302</td>
<td>DirShowOwner</td>
</tr>
<tr>
<td></td>
<td>Show file size on directory listings</td>
<td>302</td>
<td>DirShowSize</td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
<td>Page</td>
<td>Name</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>------</td>
<td>--------------------</td>
</tr>
<tr>
<td>Specify the path for the directory</td>
<td>listing internal icons</td>
<td>302</td>
<td>IconPath</td>
</tr>
<tr>
<td>Specify names of welcome files</td>
<td></td>
<td>302</td>
<td>Welcome</td>
</tr>
<tr>
<td>Enable users to have private Web</td>
<td>documents</td>
<td>304</td>
<td>UserDir</td>
</tr>
<tr>
<td>Specify a customized message for</td>
<td>a particular error condition</td>
<td>305</td>
<td>ErrorPage</td>
</tr>
<tr>
<td>Identify the workstation running the</td>
<td>the Remote Debugger</td>
<td>312</td>
<td>DebugToolAddr</td>
</tr>
<tr>
<td>Customize the server initialization step</td>
<td></td>
<td>312</td>
<td>ServerInit</td>
</tr>
<tr>
<td>Customize the user pre-exit step</td>
<td></td>
<td>312</td>
<td>PreExit</td>
</tr>
<tr>
<td>Customize the authentication step</td>
<td></td>
<td>312</td>
<td>Authentication</td>
</tr>
<tr>
<td>Customize the name translation step</td>
<td></td>
<td>312</td>
<td>NameTrans</td>
</tr>
<tr>
<td>Customize the authorization step</td>
<td></td>
<td>312</td>
<td>Authorization</td>
</tr>
<tr>
<td>Customize the object type step</td>
<td></td>
<td>312</td>
<td>ObjectType</td>
</tr>
<tr>
<td>Customize the service step</td>
<td></td>
<td>312</td>
<td>Service</td>
</tr>
<tr>
<td>Customize the PICS label retrieval step</td>
<td></td>
<td>312</td>
<td>PICSDBLookup</td>
</tr>
<tr>
<td>Customize the data filter step</td>
<td></td>
<td>312</td>
<td>DataFilter</td>
</tr>
<tr>
<td>Customize the log step</td>
<td></td>
<td>312</td>
<td>Log</td>
</tr>
<tr>
<td>Customize the error step</td>
<td></td>
<td>312</td>
<td>Error</td>
</tr>
<tr>
<td>Customize the post-exit step</td>
<td></td>
<td>312</td>
<td>PostExit</td>
</tr>
<tr>
<td>Customize the server termination step</td>
<td></td>
<td>312</td>
<td>ServerTerm</td>
</tr>
<tr>
<td>Specify external LDAP server information</td>
<td></td>
<td>318</td>
<td>LDAPInfo</td>
</tr>
<tr>
<td>Retrieve configuration file information</td>
<td>from an LDAP server</td>
<td>327</td>
<td>LDAPInclude</td>
</tr>
<tr>
<td>Specify if reports are automatically</td>
<td>generated</td>
<td>335</td>
<td>DoReporting</td>
</tr>
<tr>
<td>Name the path for the access log file</td>
<td></td>
<td>324</td>
<td>AccessLog</td>
</tr>
<tr>
<td>Remove existing access log files or run</td>
<td>a user exit</td>
<td>324</td>
<td>AccessLogArchive</td>
</tr>
<tr>
<td>Suppress log entries for specific files</td>
<td>or directories</td>
<td>325</td>
<td>AccessLogExcludeURL</td>
</tr>
<tr>
<td>Suppress log entries for files or</td>
<td>directories requested by a given method</td>
<td>326</td>
<td>AccessLogExcludeMethod</td>
</tr>
<tr>
<td>Suppress log entries for specific MIME</td>
<td>types</td>
<td>327</td>
<td>AccessLogExcludeMimeType</td>
</tr>
<tr>
<td>Suppress log entries for specific</td>
<td>return codes</td>
<td>327</td>
<td>AccessLogExcludeReturnCode</td>
</tr>
</tbody>
</table>

Appendix C. Configuration directives 269
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
<th>Page</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Remove existing access log files when they reach a given age in days</td>
<td>327</td>
<td>AccessLogExpire</td>
</tr>
<tr>
<td></td>
<td>Remove existing access log files when they reach a given collective size</td>
<td>328</td>
<td>AccessLogSizeLimit</td>
</tr>
<tr>
<td></td>
<td>Give a short description of the report to be created</td>
<td>328</td>
<td>AccessReportDescription</td>
</tr>
<tr>
<td></td>
<td>Suppress from the report the log entries for specific files or directories</td>
<td>329</td>
<td>AccessReportExcludeURL</td>
</tr>
<tr>
<td></td>
<td>Include in the report only log entries for specific files directories</td>
<td>329</td>
<td>AccessReportIncludeURL</td>
</tr>
<tr>
<td></td>
<td>Suppress from the report the log entries for specific host names</td>
<td>330</td>
<td>AccessReportExcludeHostName</td>
</tr>
<tr>
<td></td>
<td>Include in the report only log entries for specific host names</td>
<td>331</td>
<td>AccessReportIncludeHostName</td>
</tr>
<tr>
<td></td>
<td>Suppress from the report the log entries of a given method type</td>
<td>331</td>
<td>AccessReportExcludeMethod</td>
</tr>
<tr>
<td></td>
<td>Suppress from the report the log entries with a given range of return codes</td>
<td>331</td>
<td>AccessReportExcludeReturnCode</td>
</tr>
<tr>
<td></td>
<td>Name the path for the root directory where access log reports are stored</td>
<td>332</td>
<td>AccessReportRoot</td>
</tr>
<tr>
<td></td>
<td>Name the report template</td>
<td>332</td>
<td>AccessReportTemplate</td>
</tr>
<tr>
<td></td>
<td>Specify the top number of items on which to report</td>
<td>332</td>
<td>AccessReportTopList</td>
</tr>
<tr>
<td></td>
<td>Specify the path for the agent log files</td>
<td>333</td>
<td>AgentLog</td>
</tr>
<tr>
<td></td>
<td>Specify the path for the cache access log files</td>
<td>333</td>
<td>CacheAccessLog</td>
</tr>
<tr>
<td></td>
<td>Specify the path for the CGI error log files</td>
<td>334</td>
<td>CgiErrorLog</td>
</tr>
<tr>
<td></td>
<td>Name the file where you want to log internal server errors</td>
<td>334</td>
<td>ErrorLog</td>
</tr>
<tr>
<td></td>
<td>Remove existing error log files or run a user exit</td>
<td>336</td>
<td>ErrorLogArchive</td>
</tr>
<tr>
<td></td>
<td>Remove existing error log files when they reach a given age in days</td>
<td>337</td>
<td>ErrorLogExpire</td>
</tr>
<tr>
<td></td>
<td>Remove existing error log files when they reach a given collective size</td>
<td>338</td>
<td>ErrorLogSizeLimit</td>
</tr>
<tr>
<td></td>
<td>Specify common or old log file format</td>
<td>339</td>
<td>LogFormat</td>
</tr>
<tr>
<td></td>
<td>Generate debug log for HTLOGREP reporting program</td>
<td>339</td>
<td>LoggingReportingDebugOutput</td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
<td>Page</td>
<td>Name</td>
</tr>
<tr>
<td>----------------------</td>
<td>------------------------------------------------------------------------------</td>
<td>------</td>
<td>-------------------------------------------</td>
</tr>
<tr>
<td><strong>Specify the reporting program to be used</strong></td>
<td>Specify the reporting program to be used</td>
<td>335</td>
<td>LoggingReportingProgram</td>
</tr>
<tr>
<td><strong>Specify reporting program options</strong></td>
<td>Specify reporting program options</td>
<td>338</td>
<td>LoggingReportingProgramOptions</td>
</tr>
<tr>
<td><strong>Specify GMT or local time stamps in log files</strong></td>
<td>Specify GMT or local time stamps in log files</td>
<td>340</td>
<td>LogTime</td>
</tr>
<tr>
<td><strong>Suppress log entries for specific hosts or domains matching a template</strong></td>
<td>Suppress log entries for specific hosts or domains matching a template</td>
<td>341</td>
<td>NoLog</td>
</tr>
<tr>
<td><strong>Specify the path for the referer access log files</strong></td>
<td>Specify the path for the referer access log files</td>
<td>342</td>
<td>RefererLog</td>
</tr>
<tr>
<td><strong>Specify path to the compression program used for the access data files</strong></td>
<td>Specify path to the compression program used for the access data files</td>
<td>342</td>
<td>ReportDataCompressionProgram</td>
</tr>
<tr>
<td><strong>Specify path to the uncompression program used for the access data files</strong></td>
<td>Specify path to the uncompression program used for the access data files</td>
<td>342</td>
<td>ReportDataUnCompressionProgram</td>
</tr>
<tr>
<td><strong>Specify the suffix appended to the compressed file</strong></td>
<td>Specify the suffix appended to the compressed file</td>
<td>343</td>
<td>ReportDataCompressionSuffix</td>
</tr>
<tr>
<td><strong>Specify how to handle old log files not originally included in a report.</strong></td>
<td>Specify how to handle old log files not originally included in a report.</td>
<td>343</td>
<td>ReportProcessOldLogs</td>
</tr>
<tr>
<td><strong>Remove existing access data files when they reach a given collective size. These access data files are named access.mmddyyyy and are used for reports.</strong></td>
<td>Remove existing access data files when they reach a given collective size. These access data files are named access.mmddyyyy and are used for reports.</td>
<td>344</td>
<td>ReportDataSizeLimit</td>
</tr>
<tr>
<td><strong>Specify whether to remove existing access data files. These access data files are named access.mmddyyyy and are used for reports.</strong></td>
<td>Specify whether to remove existing access data files. These access data files are named access.mmddyyyy and are used for reports.</td>
<td>344</td>
<td>ReportDataArchive</td>
</tr>
<tr>
<td><strong>Remove existing access data files when they reach a given age. These access data files are named access.mmddyyyy and are used for reports.</strong></td>
<td>Remove existing access data files when they reach a given age. These access data files are named access.mmddyyyy and are used for reports.</td>
<td>345</td>
<td>ReportDataExpire</td>
</tr>
<tr>
<td><strong>Write configuration and performance data to an SMF memory segment</strong></td>
<td>Write configuration and performance data to an SMF memory segment</td>
<td>346</td>
<td>SMF</td>
</tr>
<tr>
<td><strong>Define how often SMF writes performance record information to a file</strong></td>
<td>Define how often SMF writes performance record information to a file</td>
<td>346</td>
<td>SMFRecordingInterval</td>
</tr>
<tr>
<td><strong>Meta-Information</strong></td>
<td>Specify name of subdirectory for meta-information files</td>
<td>347</td>
<td>MetaDir</td>
</tr>
<tr>
<td><strong>Methods</strong></td>
<td>Specify the suffix for meta-information files</td>
<td>348</td>
<td>MetaSuffix</td>
</tr>
<tr>
<td><strong>Disable HTTP methods</strong></td>
<td>Disable HTTP methods</td>
<td>349</td>
<td>Disable</td>
</tr>
<tr>
<td><strong>Enable HTTP methods</strong></td>
<td>Enable HTTP methods</td>
<td>349</td>
<td>Enable</td>
</tr>
<tr>
<td><strong>Multi-format processing</strong></td>
<td>Specify the language of files with particular suffixes</td>
<td>352</td>
<td>AddLanguage</td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
<td>Page</td>
<td>Name</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>------------------------------------------------------------------------------</td>
<td>------</td>
<td>-----------------</td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td><strong>Description</strong></td>
<td><strong>Page</strong></td>
<td><strong>Name</strong></td>
</tr>
<tr>
<td>Specify the MIME content</td>
<td>encoding of files with particular suffixes</td>
<td>352</td>
<td>AddEncoding</td>
</tr>
<tr>
<td>Specify, for a given</td>
<td>suffixes, the codepage in which the character sets documents are stored</td>
<td>353</td>
<td>AddCharSet</td>
</tr>
<tr>
<td>Specify the data type of</td>
<td>files with particular suffixes</td>
<td>353</td>
<td>AddType</td>
</tr>
<tr>
<td>Specify whether suffix</td>
<td>definitions are case sensitive</td>
<td>357</td>
<td>SuffixCaseSense</td>
</tr>
<tr>
<td>Specify file suffixes to</td>
<td>use when responding to particular browsers</td>
<td>357</td>
<td>AddClient</td>
</tr>
<tr>
<td><strong>Resource Mapping</strong></td>
<td>Run a CGI program for matching requests</td>
<td>359</td>
<td>Exec</td>
</tr>
<tr>
<td>Controls access to</td>
<td>directories matching Exec directives</td>
<td>361</td>
<td>ExecDirPass</td>
</tr>
<tr>
<td>Reject matching requests</td>
<td></td>
<td>361</td>
<td>Fail</td>
</tr>
<tr>
<td>Change matching requests to</td>
<td>a new result string</td>
<td>362</td>
<td>Map</td>
</tr>
<tr>
<td>Accept matching requests</td>
<td></td>
<td>364</td>
<td>Pass</td>
</tr>
<tr>
<td>Send matching requests to</td>
<td>another server</td>
<td>365</td>
<td>Redirect</td>
</tr>
<tr>
<td>Specify which environment</td>
<td>variables are inherited by CGI programs</td>
<td>366</td>
<td>InheritEnv</td>
</tr>
<tr>
<td>Specify which environment</td>
<td>variables are disinherited by CGI programs</td>
<td>367</td>
<td>DisInheritEnv</td>
</tr>
<tr>
<td><strong>Security</strong></td>
<td>Set name for a key database file</td>
<td>362</td>
<td>KeyFile</td>
</tr>
<tr>
<td>Turn port on or off for</td>
<td>HTTP</td>
<td>368</td>
<td>NormalMode</td>
</tr>
<tr>
<td>Specify levels of</td>
<td>encryption supported by the server</td>
<td>369</td>
<td>SSLCipherSpec</td>
</tr>
<tr>
<td>Select the type of SSL</td>
<td>client authentication</td>
<td>370</td>
<td>SSLClientAuth</td>
</tr>
<tr>
<td>Turn port on or off for SSL</td>
<td></td>
<td>371</td>
<td>SSLMode</td>
</tr>
<tr>
<td>Set port for SSL security</td>
<td></td>
<td>371</td>
<td>SSLPort</td>
</tr>
<tr>
<td>Associate a key database</td>
<td>with an IP address</td>
<td>372</td>
<td>SSLServerCert</td>
</tr>
<tr>
<td>Set SSL V2 session</td>
<td>timeout value</td>
<td>372</td>
<td>SSLV2Timeout</td>
</tr>
<tr>
<td>Set SSL V3 session</td>
<td>timeout value</td>
<td>372</td>
<td>SSLV3Timeout</td>
</tr>
<tr>
<td>Specify location of trusted</td>
<td>CA certificates</td>
<td>373</td>
<td>SSLX500CARoot</td>
</tr>
<tr>
<td>Specify host name or IP</td>
<td>address of the X.500 directory server</td>
<td>372</td>
<td>SSLX500Host</td>
</tr>
<tr>
<td>Specify X.500 directory</td>
<td>server port number</td>
<td>372</td>
<td>SSLX500Port</td>
</tr>
</tbody>
</table>

HTTP Server Planning, Installing, and Using
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
<th>Page</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify user ID for the LDAP connection to the X.500 directory server</td>
<td>374</td>
<td>SSLX500UserID</td>
<td></td>
</tr>
<tr>
<td>Specify user ID password for the LDAP connection to the X.500 directory server</td>
<td>375</td>
<td>SSLX500Password</td>
<td></td>
</tr>
<tr>
<td><strong>System Management</strong></td>
<td>Specify the application environments and workload management transaction classes for processing URL requests matching a given template</td>
<td>377</td>
<td>ApplEnv</td>
</tr>
<tr>
<td>Specify a set of directives to tailor an application environment</td>
<td>376</td>
<td>ApplEnvConfig</td>
<td></td>
</tr>
<tr>
<td>Set default plugin action</td>
<td>378</td>
<td>PluginDefault</td>
<td></td>
</tr>
<tr>
<td>Specify plugin not to be loaded during configuration</td>
<td>378</td>
<td>PluginExclude</td>
<td></td>
</tr>
<tr>
<td>Specify plugin to be loaded during configuration</td>
<td>378</td>
<td>PluginInclude</td>
<td></td>
</tr>
<tr>
<td>Enable and disable SNMP support</td>
<td>381</td>
<td>SNMP</td>
<td></td>
</tr>
<tr>
<td>Provide a security password to control access to a particular community of monitored servers</td>
<td>381</td>
<td>SNMPCommunityName</td>
<td></td>
</tr>
<tr>
<td>Create an e-mail address to receive SNMP problem reports</td>
<td>382</td>
<td>WebMasterEmail</td>
<td></td>
</tr>
<tr>
<td><strong>Timeouts</strong></td>
<td>Specify input timeout setting</td>
<td>382</td>
<td>InputTimeout</td>
</tr>
<tr>
<td>Specify output timeout setting</td>
<td>383</td>
<td>OutputTimeout</td>
<td></td>
</tr>
<tr>
<td>Specify script timeout setting</td>
<td>383</td>
<td>ScriptTimeout</td>
<td></td>
</tr>
<tr>
<td><strong>Tuning settings</strong></td>
<td>Specify files you want to load in memory at start up</td>
<td>384</td>
<td>CacheLocalFile</td>
</tr>
<tr>
<td>Specify the maximum amount of memory allocated for file caching</td>
<td>384</td>
<td>CacheLocalMaxBytes</td>
<td></td>
</tr>
<tr>
<td>Specify the maximum number of files to be simultaneously cached</td>
<td>385</td>
<td>CacheLocalMaxFiles</td>
<td></td>
</tr>
<tr>
<td>Enable and customize dynamic caching using the Fast Response Cache Accelerator</td>
<td>385</td>
<td>EnableFRCA</td>
<td></td>
</tr>
<tr>
<td>Specify the number of listen backlog client connections you want the server to carry before sending connection refused messages to clients.</td>
<td>384</td>
<td>ListenBacklog</td>
<td></td>
</tr>
<tr>
<td>Specify whether the cache is updated when a cached file is modified</td>
<td>390</td>
<td>LiveLocalCache</td>
<td></td>
</tr>
<tr>
<td>Specify the maximum number of threads to have active</td>
<td>390</td>
<td>MaxActiveThreads</td>
<td></td>
</tr>
<tr>
<td>Specify the maximum number of requests the server receives on a persistent connection</td>
<td>390</td>
<td>MaxPersistRequest</td>
<td></td>
</tr>
</tbody>
</table>
## Directive description and syntax

Each directive description includes:

- Heading with the directive name and a brief description
- Usage instructions
- Example of how the directive might appear in the configuration file

Each configuration directive follows the same general syntax:

```
DirectiveName value
```

- Default value or values of the directive (change only the parts of the configuration that you want to be different from the default settings).
- Initial configuration file setting

This is the setting of the directive in the configuration file before it is changed.

### Specifying request-template values

The value you specify for `request-template` is used to match an incoming URL to see if the directive will be used for the request. You can use the asterisk (*) as a wildcard character in templates. For all other characters, matching is done on a character-by-character and case-sensitive basis. For example, an incoming URL of `/abc/` or `/ABC` will not match a `request-template` of `/abc`.

### Specifying a positive or negative string

Several configuration directives allow you to specify a positive or negative string.

For positive string, you can enter:

- Yes
- On
- OK
- Enable

For negative string, you can enter:

- No
- Off
- None
- Disable

### Specifying time

Several configuration directives allow you to specify an amount of time. You can specify any combination of:

- `hh` hours
- `hh:mm` hours and minutes
hh:mm:ss
  hours, minutes, and seconds
n years
  number of 365-day years
n months
  number of 30-day months
n weeks
  number of 7-day weeks
n days
  number of 24-hour days
n hours
  number of 60-minute hours
n minutes
  number of 60-second minutes
n seconds
  number of seconds

All of your entries will be converted to seconds and added together.

**Using an asterisk (*) as a wildcard character**

Some directives contain templates for requests, path names, or host names (the `value` or `request-template` field). Except where otherwise indicated, you can use the asterisk (*) as a wildcard character in templates. For the template to be matched, an asterisk can be replaced by any character string or single character.

**Using a blank or number sign in templates**

The Web server interprets blanks as delimiters and a number sign (#) as the beginning of a comment that should be ignored. For example, a number sign precedes all comment lines in the Web server configuration file.

If you want to use a blank or number sign in templates, you must include a backslash (\) before the blank or number sign.

---

**Access control - Set up access control for the server**

Use the directives described in this section to control access to your server’s resources.

You link protection setups to groups of files based on the requests that are used to access those files. Use the DefProt and Protect directives to define the requests you want to protect.

You can define the actual protection setup in a separate protection statement or directly in the configuration file. Within the configuration file, you can define and label a protection setup using the Protection directive. You can also define a protection setup directly on a DefProt or Protect directive.

This section also describes the subdirectives that define a protection setup.

**DefProt - Specify default protection setup for requests that match a template**

Use this directive to associate a default protection setup with requests that match a template.
**Access control directives**

**Attention:** For protection to work properly, you must put your DefProt and Protect directives in the configuration file before any Pass or Exec directives that match the template used on DefProt or Protect directives.

The format of the directive is:

```
DefProt request-template setup
[username [ServerIP-address or hostname]]
```

**Note:** The directive must be typed on one line, even though it is shown here on more than one line.

`request-template`

A template for requests that you want to associate with a default protection setup. The server compares incoming client requests to the template and associates a protection setup if there is a match.

Protection is not actually activated for requests matching the template unless the request also matches a template on a subsequent Protect directive. See the description of the Protect directive for an explanation of how it is used with DefProt.

`setup`

The default protection setup you want to associate with requests that match `request-template`. Protection setup is defined with protection subdirectives. See [Protection Subdirectives on page 283](#) for descriptions of the protection subdirectives. `setup` can take one of three forms:

- A full path and file name identifying a separate file that contains the protection subdirectives.
- A protection setup label name that matches a name defined earlier on a Protection directive. The Protection directive contains the protection subdirectives.
- The actual protection subdirectives. The subdirectives must be enclosed in braces {}. The left brace character must be the last character on the same line as the DefProt directive. Each subdirective follows on its own line. The right brace character must be on its own line following the last subdirective line. You cannot put any comment lines between the braces.

`username`

The access control user to which the server should change when serving the request. This allows OS/390 UNIX System Services file protection to restrict access. This parameter is optional. The `username` will be used for controlling access to MVS resources and must include an OS/390 UNIX segment containing the UID and GID to be used for controlling access to HFS files. The `username` is only meaningful when the server is permitted to access BPX.SERVER for file requests and additionally to access BPX.DAEMON for CGI script requests. If `username` is not specified, the default access control user ID is used. See [Chapter 1. Planning for installation](#) for information on access control user IDs and OS/390 UNIX authorization environments.

`Server-IP-address` or `hostname`

If you are using multiple IP addresses or virtual hosts, use this parameter to specify an IP address or a host name. (For more information on using multiple IP addresses or virtual hosts, see [Chapter 16. Running your server with multiple IP addresses or virtual hosts](#)) The server uses the directive only for requests that come to the server on this IP address or for this host. For an IP address, this is the address of the server’s network connection, not the address of the requesting client.
Access control directives

You can specify an IP address (for example, 204.146.167.72) or you can specify a host name (for example, hostA.bcd.com).

This parameter is optional. Without this parameter, the server uses the directive for all requests regardless of the IP address the requests come in on or the host name in the URL.

Notes:
1. To use this parameter, the setup parameter must be in the form of a path and file name or a protection setup label. You cannot use protection subdirectives enclosed in braces for the setup parameter.
2. To use this parameter, you must also use the username parameter.
A wildcard character cannot be specified for a server’s IP address.

Examples:

```
DefProt /secret/* /server/protect/setup1.acc webname
```

The above example identifies a separate file that contains the protection subdirectives and a user ID for controlling access. For information about access control user IDs, see "Access control with RACF and other SAF-based security products" on page 4.

```
DefProt /secret/* SECRET-PROT webname
```

The above example uses a label name to point to the protection subdirectives. The label name must match a label name on a Protection directive. The Protection directive must come before the DefProt directive. The access control user ID, webname, overrides any specified within the SECRET-PROT protection subdirective.

```
DefProt {
    AuthType Basic
    ServerID restricted
    PasswdFile /docs/etc/WWW/restrict.password
    GroupFile /docs/etc/WWW/restrict.group
    GetMask authors
    PutMask authors
}
```

The above example includes the protection subdirectives as part of the DefProt directive.

```
DefProt /secret/* CustomerA-PROT webname 9.67.106.79
DefProt /secret/* CustomerB-PROT webname 9.83.100.45
```

The above examples use the optional IP address parameter. If your server receives requests that begin with /secret/, it associates a different default protection setup with the request based on the IP address of the network connection the request comes in on. For requests coming in on 9.67.106.79, the server associates the request with default protection defined on a Protection directive with a label of CustomerA-PROT. For requests coming in on 9.83.100.45, the server associates the request with default protection defined on a Protection directive with a label of CustomerB-PROT.

```
DefProt /secret/* CustomerA-PROT webname hostA.bcd.com
DefProt /secret/* CustomerB-PROT webname hostB.bcd.com
```
Access control directives

The above examples use the optional *hostname* parameter. If your server receives requests that begin with `/secret/`, it associates a different default protection setup with the request based on the host name in the URL. For requests coming in for hostA, the server associates the request with default protection defined on a Protection directive with a label of CustomerA-PROT. For requests coming in on hostB, the server associates the request with default protection defined on a Protection directive with a label of CustomerB-PROT.

**Program default setting**
None.

**Initial configuration file setting**
None.

**Protect - Activate protection setup for requests that match a template**

Use this directive to activate protection setup rules for requests that match a template.

**Attention:** For protection to work properly, you must put your DefProt and Protect directives before any Pass or Exec directives in your configuration file.

The format of the directive is different depending upon whether you want to point to a label or file containing the protection subdirectives or you want to include the protection subdirectives as part of the Protect directive.

If you want to point to a label or file containing the protection subdirectives, the format is as follows:

```
Protect request-template [setup-file/label
[username [IP-address or hostname]]]
```

**Note:** The directive must be typed on one line, even though it is shown here on two lines.

If you want to include the protection subdirectives as part of the Protect directive, the format is as follows:

```
Protect request-template [username [IP-address or hostname]]{
  subdirective value
  subdirective value
  ...
}
```

**request-template**
A template for requests that you want to activate protection for. The server compares incoming client requests to the template and activates protection if there is a match.

**Note:** Use caution when specifying this value to ensure that resources are protected. When the Web server compares an incoming request to a *request-template*, it matches on a character-by-character and case-sensitive basis, not an implied resource basis.

For example, a Protect directive with a *request-template* of `/* .conf` will not match an incoming URL of `/server .conf/` or `/SERVER .CONF`, resulting in the resource not being protected by this Protect directive. In this
example, even though the incoming URL /server.conf/ would match
on a file name and case basis, the match fails because a forward slash
follows the file name.

**setup-file/label**

If you are pointing to a label or file containing the protection subdirectives, use
this parameter to identify the protection setup you want to activate for
requests that match request-template.

This parameter is optional. If it is omitted, the protection setup is defined by
the most recent DefProt directive that contains a matching template.

**Protection setup is defined** with protection subdirectives. See "Protection
Subdirectives" on page 283 for descriptions of the protection subdirectives. If
present, this parameter can take one of three forms:

- A full path and file name identifying a separate file that contains the
  protection subdirectives.
- A protection setup label name that matches a name defined earlier on a
  Protection directive. The Protection directive contains the protection
  subdirectives.
- The actual protection subdirectives. The subdirectives must be enclosed in
  braces {}. The left brace character must be the last character on the same line
  as the DefProt directive. Each subdirective follows on its own line. The right
  brace character must be on its own line following the last subdirective line.
  You cannot put any comment lines between the braces.

**subdirective value**

If you want to include the protection subdirectives as part of the Protect
directive, use this parameter. The left brace character must be the last character
on the same line as the Protect directive. Each subdirective follows on its own
line. The right brace character must be on its own line following the last
subdirective line. You cannot put any comment lines between the braces.

See "Protection Subdirectives" on page 283 for descriptions of the protection
subdirectives.

**username**

The access control user to which the server should change when serving the
request. This allows OS/390 UNIX file protection to restrict access. This
parameter is optional, but if you want to use it you must also use the setup
parameter. The username will be used for controlling access to MVS resources
and must include an OS/390 UNIX segment containing the UID and GID to be
used for controlling access to HFS files. The username is only meaningful when
the server is permitted to access BPX.SERVER for file requests and additionally
to access BPX.DAEMON for CGI script requests. If username is not specified,
the default access control user ID is used. See "Chapter 1. Planning for
installation" on page 3 for information on user IDs and OS/390 UNIX
authorization environments.

**Note:** username is inherited from a DefProt directive to a Protect directive only
when the protection setup is also inherited. If you use the setup
parameter without the username parameter on a Protect directive,
username defaults to the access control user ID regardless of any
previous matching DefProt directives. Because of this default, the server
will not run with the wrong user name for a given protection setup.
Access control directives

Server-IP-address or hostname

If you are using multiple IP addresses or virtual hosts, use this parameter to specify an IP address or a host name. (For more information on using multiple IP addresses or virtual hosts, see "Chapter 16. Running your server with multiple IP addresses or virtual hosts" on page 165.) The server uses the directive only for requests that come to the server on this IP address or for this host. For an IP address, this is the address of the server’s network connection, not the address of the requesting client.

You can specify an IP address (for example, 204.146.167.72) or you can specify a host name (for example, hostA.bcd.com).

This parameter is optional. Without this parameter, the server uses the directive for all requests regardless of the IP address the requests come in on or the host name in the URL.

Notes:
1. To use this parameter, you must also use the setup-file/label or subdirective value parameters and the username parameter.

A wildcard character cannot be specified for a server’s IP address.

Examples:

UserID anybody
Protection BUS-PROT {
  UserID busybody
  AuthType Basic
  ServerID restricted
  PasswdFile /docs/WWW/restrict.pwd
  GroupFile /docs/WWW/restrict.grp
  GetMask authors
  PutMask authors
}
DefProt /secret/* /server/protect/setup1.acc webname
Protect /secret/scoop/*
Protect /secret/business/* BUS-PROT
Protect /topsecret/* {
  AuthType Basic
  ServerID restricted
  PasswdFile /docs/WWW/restrict.pwd
  GroupFile /docs/WWW/restrict.grp
  GetMask topbrass
  PutMask topbrass
}
Pass /secret/scoop/* /WWW/restricted/*
Pass /secret/business/* /WWW/confidential/*
Pass /topsecret/* /WWW/topsecret/*

In the above example, the server would activate protection as follows:

- Requests that start with /secret/scoop/ activate protection. The protection setup is defined in the /server/protect/setup1.acc protection setup file. Since the Protect directive does not specify a protection setup, the protection setup on the previously matching DefProt directive is used. Also, the server changes to the OS/390 UNIX user of webname as defined on the DefProt directive.
- Requests beginning with /secret/business/ activate protection. The protection setup is defined on the Protection directive that has a label of BUS-PROT. Also, the server changes to the OS/390 UNIX user of busybody as defined in the BUS-PROT protection setup.
Access control directives

- Requests beginning with /topsecret/ activate protection. The protection setup is included directly on the Protect directive. The OS/390 UNIX user defaults to ANYBODY.

Note: The user ID ANYBODY must be defined and the server must have permission to use it as a surrogate.

Protect /secret/* CustomerA-PROT webname 9.67.106.79
Protect /secret/* CustomerB-PROT webname 9.83.100.45
Protect /topsecret/* webname 9.67.106.79 {
  AuthType Basic
  ServerID restricted
  PasswdFile /docs/WWW/customer-A.pwd
  GroupFile /docs/WWW/customer-A.grp
  GetMask A-brass
  PutMask A-brass
}
Protect /topsecret/* webname 9.83.100.45 {
  AuthType Basic
  ServerID restricted
  PasswdFile /docs/WWW/customer-B.pwd
  GroupFile /docs/WWW/customer-B.grp
  GetMask B-brass
  PutMask B-brass
}

The above examples use IP addresses. If your server receives requests that begin with /secret/ or /topsecret/, it activates a different protection setup for the request based on the IP address of the network connection the request comes in on.

For /secret/ requests coming in on 9.67.106.79, the server activates the protection setup defined on a Protection directive with a label of CustomerA-PROT. For /topsecret/ requests coming in on 9.67.106.79, the server activates the protection setup defined inline on the Protect directive for /topsecret/.

For /secret/ requests coming in on 9.83.100.45, the server activates the protection setup defined on a Protection directive with a label of CustomerB-PROT. For /topsecret/ requests coming in on 9.83.100.45, the server activates the protection setup defined inline on the Protect directive for /topsecret/.

Protect /secret/* CustomerA-PROT webname hostA.bcd.com
Protect /secret/* CustomerB-PROT webname hostB.bck.com
Protect /topsecret/* webname hostA.bcd.com {
  AuthType Basic
  ServerID restricted
  PasswdFile /docs/WWW/customer-A.pwd
  GroupFile /docs/WWW/customer-A.grp
  GetMask A-brass
  PutMask A-brass
}
Protect /topsecret/* webname hostB.bcd.com {
  AuthType Basic
  ServerID restricted
  PasswdFile /docs/WWW/customer-B.pwd
  GroupFile /docs/WWW/customer-B.grp
  GetMask B-brass
  PutMask B-brass
}

The above examples use virtual hosts. If your server receives requests that begin with /secret/ or /topsecret/, it activates a different protection setup for the request based on the host name in the URL.
For /secret/ requests coming in for hostA.bcd.com, the server activates the protection setup defined on a Protection directive with a label of CustomerA-PROT. For /topsecret/ requests coming in for hostA.bcd.com, the server activates the protection setup defined inline on the Protect directive for /topsecret/.

For /secret/ requests coming in for hostB.bcd.com, the server activates the protection setup defined on a Protection directive with a label of CustomerB-PROT. For /topsecret/ requests coming in for hostB.bcd.com, the server activates the protection setup defined inline on the Protect directive for /topsecret/.

**Initial configuration file setting**

Protection is provided for the Configuration and Administration forms by a Protect directive with a request template of /admin-bin/*.

**Protection - Define a named protection setup within the configuration file**

Use this directive to define a protection setup within the configuration file. You give the protection setup a name and define the type of protection using protection subdirectives.

**Note:** In the configuration file, you must place Protection directives before any DefProt or Protect directives that point to them.

The format of the directive is:

```plaintext
Protection label-name { 
 subdirective value 
 subdirective value 
 . 
 . 
 . 
}
```

**label-name**

The name you want to associate with this protection setup. The name can then be used by subsequent DefProt and Protect directives to point to this protection setup.

**subdirective value**

Put a protection subdirective and its value on each line between the left brace and the right brace. You cannot put any comment lines between the braces.

See ["Protection Subdirectives" on page 283](#) for descriptions of the protection subdirectives.

**Example**

```plaintext
Protection NAME-ME { 
 AuthType Basic 
 ServerID restricted 
 PasswdFile /WWW/password.pwd 
 GroupFile /WWW/group.grp 
 GetMask groupname 
 PutMask groupname 
}
```
**Initial configuration file setting**

```plaintext
Protection IMW_Admin {
    ServerId IMWEBSRV_Administration
    AuthType Basic
    PasswdFile %%SAF%%
    Mask WEBADM, WEBADM
}
```

### Protection Subdirectives

Following are descriptions of each of the protection subdirectives that can be used in a protection setup. The subdirectives are listed in alphabetical order, but do not need to be in any particular order in your configuration file.

Protection setups can either be in separate files or within the configuration file as part of DefProt, Protect, or Protection directives.

See the previous descriptions of the DefProt, Protect, and Protection directives for examples of using protection setups.

**ACLOverride - Specify that ACL files override protection setups**

Use this subdirective with a value of On if you want Access Control List files (ACL) to override the masks specified in the protection setup. If a directory being protected by the protection setup has an ACL file, the mask subdirectives in the protection setup are ignored. (The mask subdirectives are DeleteMask, GetMask, Mask, PostMask, and PutMask.)

**Example:**

```
ACLOverride On
```

**AuthType - Specify authentication type**

Use this subdirective when limiting access based on user names and passwords. Specify the type of authentication to use when the client sends a password to the server. With basic authentication (AuthType Basic), passwords are sent to the server as plain text. They are encoded, but not encrypted.

**Example:**

```
AuthType Basic
```

**DeleteMask - Specify the user names, groups, and addresses allowed to delete files**

Use this subdirective to specify user names, groups, and addresses templates authorized to make DELETE requests to a protected directory.

**Example:**

```
DeleteMask authors,(niceguy,goodie)@96.96.3.1,128.141.*.*
```

**GetMask - Specify the user names, groups, and addresses allowed to get files**

Use this subdirective to specify user names, groups, and address templates authorized to make GET requests to a protected directory.

**Example:**

```
GetMask authors,(niceguy,goodie)@96.96.3.1,128.141.*.*
```
Access control directives

**GroupFile - Specify the location of the associated group file**

Use this subdirective to specify the path and file name of the server group file that you want this protection setup to use. The groups defined within the server group file can then be used by:

- Any mask subdirectives that are part of the protection setup. (The mask subdirectives are DeleteMask, GetMask, Mask, PostMask, and PutMask.)
- Any ACL file on a directory that is protected by the protection setup.

**Example:**

```
GroupFile /docs/etc/WWW/restrict.group
GroupFile %%LDAP[: PrimaryLdapServer[, SecondaryLdapServer]]%%
```

PrimaryLdapServer and SecondaryLdapServer match LDAP servers defined in LDAPInfo directives.

**Attention:** The LDAPInfo directive must precede any %%LDAP%% references in the configuration file.

The server first looks to the PrimaryLdapServer for group information. If there is no response, the request is made to the SecondaryLdapServer. The server with whom a connection is made will be used to look up the password information.

**Mask - Specify the user names, groups, and addresses allowed to make HTTP requests**

Use this subdirective to specify user names, groups, and address templates authorized to make HTTP requests not covered by other mask subdirectives. See "Methods - Set method acceptance" on page 348 for descriptions of the HTTP methods supported by the server.

**Example:**

```
Mask authors,(niceguy,goodie)@96.96.3.1,128.141.*.*
```

**Note:** When you use the Mask directive, it is important that you keep in mind that Masks are case sensitive. The following is an example of how you would issue Mask protection on a user ID:

**Example:**

```
MASK WEBADM, webadm
```

**PasswdFile - Specify the location of the associated password file**

Use this subdirective when limiting access based on user names and passwords. Specify the path name of the password file that you want this protection setup to use, or that you want to use the SAF interface to your system security subsystem to validate user names and passwords.

Because some browsers such as NetScape cache userid/password by security realm (ServerID) within host, follow these guidelines when specifying ServerID and password files:

- Protection setups that use the same password file should use the same ServerID.
- Protection setups that use different password files should use different ServerIDs.

**Examples:**

```
PasswdFile %%SAF%%
PasswdFile %%LDAP[: PrimaryLdapServer[, SecondaryLdapServer]]%%
PasswdFile /usr/lpp/internet/server_root/Admin/heroes.pwd
```
In the above example, note that %%SAF%% lets the server know that password verification is done with a SAF controlled operating system (for example, RACF).

**Note:** If the access control UserID resolves to %%CLIENT%% (from either the UserID subdirective or the UserID directive) SAF will always be used to verify the UserID and password, instead of the given password file.

PrimaryLdapServer and SecondaryLdapServer match LDAP servers defined in LDAPInfo directives.

**Attention:** The LDAPInfo directive must precede any %%LDAP%% references in the configuration file.

The server first looks to the PrimaryLdapServer for password information. If there is no response, the request is made to the SecondaryLdapServer. The server with whom a connection is made will be used to look up the group information.

**PostMask - Specify the user names, groups, and addresses allowed to post files**
For a secure server, use this subdirective to specify users, groups, and address templates authorized to make POST requests to a protected directory.

Example:

```
PostMask authors,(niceguy,goodie)@96.96.3.1,128.141.*.*
```

**PutMask - Specify the users names, groups, and addresses allowed to put files**
Use this subdirective to specify users, groups, and address templates authorized to make PUT requests to a protected directory.

Example:

```
PutMask authors,(niceguy,goodie)@96.96.3.1,128.141.*.*
```

**ServerID - Specify a name to associate with the password file**
Use this subdirective when limiting access based on user names and passwords. Specify a name you want to associate with the password file being used. The name does not need to be a real machine name.

The name is used as an identifier to the requester. Since different protection setups can use different password files, having a name associated with the protection setup can help the client decide which password to send. Most clients display this name when prompting for a user name and password.

Because some browsers such as NetScape cache userid/password by security realm (ServerID) within host, follow these guidelines when specifying ServerID and password files:

- Protection setups that use the same password file should use the same ServerID.
- Protection setups that use different password files should use different ServerIDs.

Example:

```
ServerID restricted
```
Access control directives

**UserID - Specify the Access Control user ID that the server should use**

Use this subdirective to specify the access control user ID that the server should change to when serving a request. You can use this subdirective only in a protection setup within the configuration file, not in a separate protection setup file. If you do not specify the access control user ID, the default access control user ID is used.

Web servers frequently need to process requests from users that do not have user IDs on the system running the server. The server uses access control User IDs to access resources for these requests. The HTTP Server uses four types of access control user IDs. There are special user IDs, %%CLIENT%%, %%SERVER%%, %%CERTIF%%, and surrogate.

**%%CLIENT%%**

The Web server requires that the requester have a local OS/390 user ID and password. The requester's user ID is used to access the data. The user is prompted for a valid password.

**%%SERVER%%**

The Web server uses its own user ID to access data.

*Note:* Be extremely cautious when using %%SERVER%%. If your server is running as a superuser, this gives all users superuser authority.

**%%CERTIF%%**

The Web server treats SSL connection certificate data in a special way. When presented with an SSL session with client certificate data present, the Web server attempts to map the client certificate to a local MVS User ID and password. The request is treated as if %%CLIENT%% has been specified in the following situations:

- The session is not an SSL session.
- There is no certificate present or the certificate cannot be mapped.
- The underlying support is not available.

Note that SSLClientAuth must be set on in order to get client certificate data.

**Related information:**

- "Access control with RACF and other SAF-based security products" on page 4
- "Surrogate user IDs" on page 20
- The ability to change to a different OS/390 UNIX identity is controlled on MVS. The server must be given surrogate authority for the specified user ID and might be permitted to access the BPX.SERVER and BPX.DAEMON facilities. For more information about the BPX facilities, see "Chapter 1. Planning for installation" on page 3 and the OS/390 UNIX System Services Planning book.

**Example:**

```
UserID WWW
UserID %%SERVER%%
```

Use %%SERVER%% if you are sharing information on an external LDAP server. See "Chapter 14. Retrieving LDAP information" on page 153 for more information.
SSL client authentication subdirectives

If you implement SSL client authentication, the server requests the client’s certificate when the client makes an HTTPS request. The server establishes a secure connection whether or not the client has a valid certificate.

You can restrict who can access documents by using password files and/or user or group authentication in protection setups. For more information, see “Setting up protection for server resources” on page 58. You can further restrict who can access documents by coding SSL client authentication parameters on protection setups, ACL files, or both.

Using SSL client authentication parameters as subdirectives, you can specify that the client certificate is valid or you can specify all or part of the Distinguished Name (DN) of a client or of the certification authority (CA) who issued the client’s certificate.

When you use SSL client authentication parameters, the server first checks to see if the client certificate is valid, as is. If not, it compares any DN information in a protection setup and then compares any DN information in an ACL file with the DN information in the client’s certificate. If the DN information matches, the server serves the document.

The following can be specified on the Protection or Protect directive:

- The validity of the client certificate.
  - SSL_ClientAuth client - indicates that the client certificate is valid without verifying any of the Distinguished Name information in the client certificate.
    Only the keyword client is valid with this parameter.
- All or any of the following parameters that make up a client’s Distinguished Name in the client’s certificate:
  - CommonName - the client’s common name
  - Country - the country in which the client resides
  - Locality - the locality in which the client resides
  - StateOrProvince - the state or province in which the client resides
  - Organization - the organization of the client
  - OrgUnit - the organizational unit of the client
- All or any of the following subdirectives that make up the CA’s Distinguished Name in the client’s certificate:
  - IssuerCommonName - the CA’s common name
  - IssuerCountry - the country in which the CA resides
  - IssuerLocality - the locality in which the CA resides
  - IssuerStateOrProvince - the state or province in which the CA resides
  - IssuerOrganization - the organization of the CA
  - IssuerOrgUnit - the organizational unit of the CA

Example

```
Protect /topsecret/* {  
CommonName "Dr Sheila A. Jones"  
Organization "RTP Quick Care Center"  
Mask Anybody@(*)  
}
```

In the above example of an inline Protect directive, any HTTPS request beginning with /topsecret/ causes the server to request the client’s certificate. If the client’s common name is Dr Sheila A. Jones and the client’s organization is RTP Quick Care Center, then the server returns the document to the client.
Hints and tips for coding SSL client authentication parameters

- Specify any or all of a client or CA’s DN
- Enclose DN information that contains blanks in double quotes (as shown in the above example).
- Make sure the DN information matches the DN information in the client’s certificate. This information is case sensitive and must have the same punctuation.
- Do not use wildcard characters for any of the parameters.

Basic - Specify required settings

Use the directives described in this section to control your server's basic configuration settings.

BindSpecific - Specify if the server binds to one or all IP addresses

Use this directive on a multi-networking system to run a different server on each IP address. All the servers may listen on the same port.

If you specify On, the server binds to the IP address specified in the HostName directive only, instead of binding to all local IP addresses.

Note: Beginning with OS/390 V2R5, if you are configuring your Web server to use virtual hosting with BindSpecific on and using the same port for multiple Web servers, you must use the SHAREPORT option on the PORT statement in your TCP/IP profile.

If this directive has not been specified, the server binds to the default HostName.

If you change this directive, you must stop the server and then start it again. The server will not pick up the change if you only restart it.

Example

BindSpecific On

Initial configuration file setting
None

Program default setting
BindSpecific Off

Bounce — Specify the default start option for the sockets setting SO_REUSEADDR

By default, the Web server starts with the sockets setting SO_REUSEADDR on. This will avoid a potential delay while sockets used by a previous invocation of the Web server are closed. This is especially useful if you have automation software that recognizes the Web server termination and needs to start a replacement Web server as quickly as possible.

You must use this directive to set SO_REUSEADDR off (Bounce off). To set Bounce off, you must also omit the -B flag from your Web server startup command. The startup command will override the directive setting. See "httpd command" on page 249 for information on the startup command.
If bounce is set off, the Web server will retry the bind to its listen ports for up to two minutes to allow previously used sockets to close. If the Web server is started quickly after a previous Web server shutdown and Bounce is set off, the listen ports may not be available. This can result in a delay in the availability of the Web server.

If you change this directive, you must stop the server and then start it again. The server will not pick up the change if you only restart it.

Example

Bounce Off

Initial configuration file setting

None

Program default setting

Bounce On

DNS-Lookup - Specify whether you want to look up host names of clients

Use this directive to specify whether you want your server to look up the host name of requesting clients.

The value you use affects the following things about how your server works:

- The performance of the server. Using the default value of Off improves the performance and response time of the server because it does not use resources to perform the host name lookup.
- The information your server records about clients when writing to log files.
  - Off - Clients identified by IP address
  - On - Clients identified by host name
- Whether you can use host names on address templates in protection setups, server group files, and ACL files.
  - Off - Cannot use host names on address templates; must use IP addresses
  - On - Can use host names on address templates; cannot use IP addresses

Example

DNS-Lookup On

Initial configuration file setting

DNS-Lookup Off

HostName - Specify the fully qualified domain name or IP address for the server

Use this directive to specify the domain name or an IP address returned to clients from document requests. If you specify a domain name, a domain name server must be able to resolve the name into an IP address. If you specify an IP address, the domain name server is not needed or accessed.

Important migration note: In previous releases, the HostName directive bound the server to only the IP address specified, instead of binding to all local IP addresses. Currently, you must also specify BindSpecific On if you want the server to bind to only the IP address specified for this directive.
Basic directives

Example

HostName name or IP address

Initial configuration file setting
None.

Default
local node returned from system call: gethostname()

**imbeds - Specify whether server-side includes will be dynamically imbedded**

Use this directive to specify if you want server-side include processing to be performed for documents served from the file system, CGI programs, or both. Server-side include processing is done on documents with a content type of text/x-ssi-html. Optionally, you can specify that server-side include processing also be done for documents with a content type of text/html. For more information about content types, see "AddType - Specify the data type of files with particular suffixes" on page 353.

You can use server-side includes to dynamically insert information, such as the date, the size of a file, the last change date of a file, CGI or server-side include environment variables, or text documents into the document being returned. For more information on using server-side includes, see "Using server-side includes to insert information into CGI programs and HTML documents" on page 93.

Server-side include processing causes the server to search your documents for special commands each time they are served. This can affect the server's performance and slow down response time to clients.

The format of this directive is:

```
imbeds source [type]
```

*source* can be:

- **on** Server-side include processing is done for documents from the file system and from CGI programs.
- **files** Server-side include processing is only done for documents from the file system.
- **cgi** Server-side include processing is only done for documents returned by CGI programs.
- **off** Server-side include processing is not done for any documents.

The server checks the content type of each file it retrieves and the output of each CGI program it processes.

Server-side include processing is normally done only for documents having a content type of text/x-ssi-html. However, you can specify that documents with a content type of text/html be processed for server-side includes.

*type* can be:

- **SSIOnly** Server-side include processing is done for documents with a content type of text/x-ssi-html.
html  Server-side include processing is done for documents with a content type of text/html and a content type of text/x-ssi-html.

Note: The server treats html, .html, and .htm as html. Anything else is treated as SSIOnly.

Initial configuration file setting
imbeds on SSIOnly

Program default setting
imbeds off SSIOnly

InstallPath - Specify an alternate directory installation path
Use this directive to specify an alternate installation directory path for the HTTP Server. Installing the server in an alternate directory path allows you to install a new release of the server without disrupting the server you currently have running. You can sufficiently test the new release of the server while keeping the previous release of the server in production.

The server is installed into a directory other than the default directory of /usr/lpp/internet and is executed from the alternate directory. It is recommended that you install the new release of the server in a nonstandard place on a system with the previous release in the standard place. When the HTTP Server is installed in an alternate directory, the default configuration files created are automatically tailored to the alternate installation location. When you finish testing, you should be able to unmount the new server from its original location and remount it over /usr/lpp/internet to move it from test to production. Your current configuration files (httpd.conf, httpd.envvars, ics_pics.conf,lgw_fcgi.conf, mvsds.conf, javelin.conf, and socks.conf) may not be compatible with the new level of the server. You need to manually modify the configuration files to the new path, use the configuration files from the previously installed server, or use the default configuration files shipped in the installpath/etc directory.

Example
InstallPath /service/usr/lpp/internet

Initial configuration file setting
InstallPath /usr/lpp/internet

PidFile - Specify the location of the process ID file
Use this directive to specify the full path and file name of the file that you want the server to write its process ID to when you start the server.

If you change this directive, you must stop your server and then start it again for the change to take effect. The server will not pick up the change if you only restart it.

Example
PidFile /other/ibm_server.pid

Initial configuration file setting
/usr/lpp/internet/server_root/httpd.pid
Basic directives

**Port - Specify the port on which you want the server to listen for requests**

Use this directive to specify the port number the server should listen to for requests. The standard port number for HTTP is 80. Other port numbers less than 1024 are reserved for other TCP/IP applications and should not be used. Common ports used for proxy Web servers are 8080 and 8008.

When a port other than 80 is used, clients are required to include a specific port number on requests to the server. The port number is preceded by a colon and placed after the host name on the URL. For example, from the browser, the URL

```
http://www.turfco.com:8008/
```

requests the default welcome page from a host named www.turfco.com that is listening on port 8008.

You can use the `-p` option on the `httpd` command to override this setting when starting the server.

If you change this directive, you must stop your server and then start it again for the change to take effect. The server will not pick up the change if you only restart it.

**Example**

*Port 8080*

**Initial configuration file setting**

*Port 80*

**Recovery — Customize ABEND recovery performed by the Web server**

Use this directive to control the action the Web server will take when it detects that an ABEND has occurred in its address space. For planning considerations, see "ABEND recovery performed by the Web server" on page 3.

The format of this directive is:

```
Recovery type
```

type can be:

- **None**  No ABEND recovery is performed.
- **MsgOnly**  No ABEND recovery is performed. The Web server issues messages, then terminates. No dump is taken.
- **Msg/Dump**  No ABEND recovery is performed. The Web server issues messages, takes a dump, then terminates.
- **Normal**  ABEND recovery will be attempted by the Web server. The Web server issues messages, takes a dump, then recovers if possible. If the Web server is running in Scalable Server mode, no ABEND recovery will be attempted for WLM queue servers.
- **Full**  ABEND recovery will be attempted by the Web server. The Web server
issues messages, takes a dump, then recovers if possible. If the Web server
is running in Scalable Server mode, ABEND recovery will be attempted for
WLM queue servers.

For more information on error conditions, see the descriptions of messages
IMW0085E and IMW0162E in “Appendix H. Messages” on page 421.

Program default setting

Recovery Normal

ServerRoot - Specify the current working directory of the server

Use this directive to specify the current working directory of the server. By default,
this directory is located in the install path specified by the InstallPath directive.
This directive is relative to the InstallPath directive. Logging directives use this
current working directory as the default root when relative path names are used.

Example

ServerRoot /usr/lpp/internet/server_root

Initial configuration file setting

ServerRoot /usr/lpp/internet/server_root

Note: PASS and EXEC rules may be independent of this directory.

Userid - Specify the Default Access Control user ID

Use this directive to specify the user name the server changes to before accessing
files. The ability to change to a different OS/390 UNIX identity is controlled on
MVS. The server must be given surrogate authority for any specified surrogate
user ID and might need to be permitted to access the BPX.SERVER and
BPX.DAEMON facilities. For more information about BPX facilities, see “Chapter 1.
Planning for installation” on page 3 and OS/390 UNIX System Services Planning.

The server does not normally serve data from its own WEBSRV user ID because of
its level of authority. The server uses access control user IDs to access resources for
requests. Every request must have an access control user ID using the Protect,
DefProt, or UserId directive. Therefore, any pass rules that do not invoke a protect
will use this user ID. The HTTP Server uses four types of access control user IDs.
There are special user IDs, %%CLIENT%%, %%SERVER%%, %%CERTIF%%, and
surrogate.

%%CLIENT%%
This is not really a user ID. Instead, it tells the server to require that the
requester have a local OS/390 user ID and password. The requester’s user
ID is used to access the data. The user is prompted for a valid password.

%%SERVER%%
This is not really a user ID. Instead, it tells the server to use its own user
ID to access data.

Note: Be extremely cautious when using %%SERVER%%. If your server is
running as a superuser, all users have superuser authority.

%%CERTIF%%
This is not really a user ID. Instead, it tells the server to treat SSL
connection certificate data in a special way. The web server, when
presented with an SSL session with client certificate data present, attempts to map the client certificate to a local MVS User ID and password. If the session isn’t SSL, or there isn’t a certificate present, or the underlying support is not available, or the certificate can’t be mapped, then the request is treated as if %%CLIENT%% has been specified. Note that SSLClientAuth must be ON in order to get client certificate data.

For more information on access control user IDs, see “Chapter 1. Planning for installation” on page 3.

Example

UserId webmaster

Initial configuration file setting

UserId %%CLIENT%%

Codepages - Specify default codepage environment

Use the directives described in this section for codepage translation, primarily between ASCII and EBCDIC. Each directive specifies one codepage. The two directives together define a pair that must be supported by the iconv() service.

**DefaultFsCp - Specify server codepage**

Use this directive to specify the default file system code page on the server. This is the EBCDIC codepage for local text files and text streams from applications (gateway). For example, acceptable values are IBM-1047 (Open Systems Latin 1 codepage) and IBM-939 (Japanese (Latin) Extended codepage). For information about the iconv(s) service, see the Code Page Conversion Table, “Code Set Conversion Utilities” chapter in the C/MVS Programming Guide. You might find the C/C++ for OS/390 Library Reference helpful also.

You can use the -fscp option on the httpd command to override the DefaultFsCp directive when you start the server.

If you change this directive, you must stop the server and start it again to make it effective.

**Example**

DefaultFsCp IBM-939

**Initial configuration file setting**

None.

**Program default setting**

DefaultFsCp IBM-1047

**DefaultNetCp - Specify codepage**

Use this directive to specify the default network codepage. This is the ASCII codepage used for text bodies on the network. Some acceptable values are:

- ISO8859-1 (8bit single-byte coded graphic character sets)
- IBM-932C (Shift JIS)
- IBM-eucJC (Japanese version of Extended Unix Code)
Codepages directives

For information about the iconv(s) service, see the Code Page Conversion Table, “Code Set Conversion Utilities” chapter in the C/MVS Programming Guide. You might find the C/C++ for OS/390 Library Reference helpful also.

You can use the -netcp option on the httpd command to override the DefaultNetCp directive when you start the server.

If you change this directive, you must stop the server and start it again to make it effective.

Example

    DefaultNetCp IBM-932C

Initial configuration file setting
None.

Program default setting

    DefaultNetCp ISO8859-1

Directories and Welcome Page - Set viewing options

Use the directives described in this section to control how your server responds to requests containing a directory name. You can have the server search the directory for a welcome file to return, or you can have the server generate a directory listing.

By default, the server first looks for a welcome file. If no welcome file is present, the server displays a directory listing. Configuration settings control how directory listings appear and the icons that the listings use.

The server provides a set of default icons to use for directory listings. You can replace these icons with others using some of the directives described in this section.

AddBlankIcon - Specify the icon URL used to align the heading of directory listings

Use this directive to specify an icon to use for aligning the heading on directory listings. This can either be a blank icon or another icon you want to appear on the headings of your directory listings. For proper alignment, the icon you use must be the same size as the other icons you are using on your directory listings. The format of the directive is:

    AddBlankIcon icon-URL alternate-text

icon-URL

The URL for the icon. The URL is translated through the mapping directives. For the icon to be retrieved, the mapping directives must allow the URL to be passed.

If you are using the server as a proxy, you must specify a fully qualified URL pointing to your server.

alternate-text

The alternate text to use for the icon if the requesting browser is not displaying graphics.
Directories/Welcome Page directives

Example
AddBlankIcon /icons/logo.gif logo

Initial configuration file setting
AddBlankIcon /icons/blank.gif

The default does not specify alternative text since the icon is blank.

AddDirIcon - Specify the icon URL for directories on directory listings

Use this directive to specify an icon for representing a directory on a directory listing. The format of the directive is:
AddDirIcon icon-URL alternate-text

icon-URL

The URL for the icon. The URL is translated through the mapping directives. For the icon to be retrieved, the mapping directives must allow the URL to be passed.

If you are using the server as a proxy, you must specify a fully qualified URL pointing to your server. You must map the URL to a local file and make sure that the mapping directives allow the URL to be passed.

alternate-text

The alternate text to use for the icon if the requesting browser is not displaying graphics.

Example
AddDirIcon /icons/direct.gif DIR

Initial configuration file setting
AddDirIcon /icons/dir.gif DIR

AddIcon - Bind an icon to a MIME content-type or encoding-type

Use this directive to specify icons for representing files with a specific MIME content-type or encoding-type. The server uses the icons on directory listings. The format of the directive is:
AddIcon icon-URL alternate-text
type-template
icon-URL

The URL for the icon. The URL is translated through the mapping directives. For the icon to be retrieved, the mapping directives must allow the URL to be passed.

If you are using the server as a proxy, you must specify a fully qualified URL pointing to your server. You must map the URL to a local file and make sure that the mapping directives allow the URL to be passed.

alternate-text

The alternate text to use for the icon if the requesting browser is not displaying graphics.
Directories/Welcome Page directives

**type-template**
Either a MIME content-type or encoding-type template. Content-type templates always contain a slash. Encoding-type templates never have a slash.

**Example**

```
AddIcon /icons/movie.gif video video/*
```

**Initial configuration file setting**

```
AddIcon /internet/server_root/icons/binary.gif BIN binary
AddIcon /internet/server_root/icons/text.gif TXT text/*
AddIcon /internet/server_root/icons/image.gif IMG image/*
AddIcon /internet/server_root/icons/movie.gif MOV video/*
AddIcon /internet/server_root/icons/sound.gif AU audio/*
AddIcon /internet/server_root/icons/tar.gif TAR multipart/*tar
AddIcon /internet/server_root/icons/compress.gif CMP x-compress x-gzip
```

**AddParentIcon - Specify the icon URL for a parent directory on directory listings**

Use this directive to specify an icon for representing a parent directory on a directory listing. The format of the directive is:

```
AddParentIcon icon-URL alternate-text
```

**icon-URL**

The URL for the icon. The URL is translated through the mapping directives. For the icon to be retrieved, the mapping directives must allow the URL to be passed.

If you are using the server as a proxy, you must specify a fully qualified URL pointing to your server. You must map the URL to a local file and make sure that the mapping directives allow the URL to be passed.

**alternate-text**
The alternate text to use for the icon if the requesting browser is not displaying graphics.

**Example**

```
AddParentIcon /icons/parent.gif UP
```

**Initial configuration file setting**

```
AddParentIcon /icons/back.gif UP
```

**AddUnknownIcon - Specify the icon URL for unknown file types on directory listings**

Use this directive to specify an icon for representing files with an unknown file type on a directory listing. The format of the directive is:

```
AddUnknownIcon icon-URL alternate-text
```

**icon-URL**

The URL for the icon. The URL is translated through the mapping directives. For the icon to be retrieved, the mapping directives must allow the URL to be passed.

If you are using the server as a proxy, you must specify a fully qualified URL pointing to your server. You must map the URL to a local file and make sure that the mapping directives allow the URL to be passed.
Directories/Welcome Page directives

*alternate-text*

The alternate text to use for the icon if the requesting browser is not displaying graphics.

**Example**

```
AddUnknownIcon /icons/saywhat.gif huh
```

**Initial configuration file setting**

```
AddUnknownIcon /icons/unknown.gif ???
```

**AlwaysWelcome - Specify if a welcome file is returned for all directory requests**

Use this directive to specify if you want your server to always handle directory requests by first searching the directory for a welcome file.

The default value is **On**, which means that the server always searches the directory for a welcome file. The Welcome directive specifies the names of the files that the server recognizes as welcome files.

If you change the value to **Off**, the server first checks the last character of directory requests for the slash (/) character. If a directory request ends with a slash, the server searches the directory for a welcome file. If a directory request does not end with a slash, the server attempts to return a directory listing.

If the server does not find a welcome file, or AlwaysWelcome is set to Off and the directory request does not end in a slash, the DirAccess directive controls whether or not the server responds to the request with a directory listing.

**Note:** Setting AlwaysWelcome to Off does not affect requests that contain only your server name without a directory name. The server will always handle these requests by looking in your document root directory for a welcome file. The server cannot generate a directory listing for the document root directory.

**Example**

```
AlwaysWelcome Off
```

**Initial configuration file setting**

```
AlwaysWelcome On
```

**DirAccess - Control directory listings**

Use this directive to specify whether you want your server to return directory listings when requested. The values on the Welcome and AlwaysWelcome directives determine when a request is interpreted as a request for a directory listing.

The default value is **Off**, which means that the server does not return directory listings for all directories and subdirectories. If you want to control which directories and subdirectories the server can return directory listings for, use:

```
DirAccess Selective
```

If you change the value to **On**, the server will return directory listings.
Directories/Welcome Page directives

If you change the value to Selective, the server will return directory listings for any directory that contains a file named .www_browsable. The contents of the .www_browsable file are not important; the server only checks for its existence.

Examples:
DirAccess Off
DirAccess Selective

Initial configuration file setting
DirAccess Off

DirReadme - Control directory README files

Use this directive to specify if and where you want your server to display directory listing README files.

The default value is Top, which means that when the server returns a directory listing, it searches the directory for a file named README. If README is found, the server puts the contents of the file at the top of the directory listing.

If you change the value to Bottom, the server searches for a README file, but puts the contents at the bottom of the directory listing.

If you change the value to Off, the server does not search the directory for a README file.

Examples:
DirReadme Bottom
DirReadme Off

Initial configuration file setting
DirReadme Top

DirShowBrackets - Use brackets around alternative text on directory listings

Use this directive to specify whether you want the server to put brackets around alternative text on directory listings. The directives that specify directory listing icons also contain alternate text. The alternate text is used in place of an icon if the requesting browser is not displaying graphics.

Example
DirShowBrackets Off

Initial configuration file setting
DirShowBrackets On

DirShowBytes - Show byte count for small files on directory listings

Use this directive to specify whether directory listings should include the exact byte count for files smaller than 1 KB.

A value of Off means the directory listing shows a size of 1 KB for all files that are 1 KB or smaller.

Example
DirShowBytes On
Directories/Welcome Page directives

Initial configuration file setting
DirShowBytes Off

DirShowCase - Use case when sorting files on directory listings
Use this directive to specify whether directory listings should distinguish between uppercase and lowercase letters when sorting file names.

A value of On means uppercase letters are placed after lowercase letters.

Example
DirShowCase Off

Initial configuration file setting
DirShowCase On

DirShowDate - Show date last modified on directory listings
Use this directive to specify whether directory listings should include the last modification date for each file.

Example
DirShowDate Off

Initial configuration file setting
DirShowDate On

DirShowDescription - Show descriptions for files on directory listings
Use this directive to specify whether directory listings should include descriptions for HTML files. The descriptions are taken from the files’ HTML <title> tags.

Example
DirShowDescription Off

Initial configuration file setting
DirShowDescription On

DirShowGroup - Show the group ID of files on directory listings
Use this directive to specify whether directory listings should include the Group ID of the files.

Example
DirShowGroup On

Initial configuration file setting
None.

Program default setting
Off
DirShowHidden - Show hidden files on directory listings

Use this directive to specify whether directory listings should include any hidden files on the directory.

The server considers any file that has a name beginning with a period (.) to be a hidden file.

Example

```
DirShowHidden Off
```

Initial configuration file setting

```
DirShowHidden On
```

DirShowIcons - Show icons in directory listings

Use this directive to specify whether you want your server to include icons in directory listings. Icons can be used to provide a graphic representation of the content type of the files in the listing. The icons themselves are defined by the AddBlankIcon, AddDirIcon, AddIcon, AddParentIcon, and AddUnknownIcon directives.

Example

```
DirShowIcons Off
```

Initial configuration file setting

```
DirShowIcons On
```

DirShowMaxDescrLength - Set the maximum description length on directory listings

Use this directive to set the maximum number of characters to show in the description field on directory listings.

Example

```
DirShowMaxDescrLength 30
```

Initial configuration file setting

```
DirShowMaxDescrLength 25
```

DirShowMaxLength - Set the maximum length for file names on directory listings

Use this directive to set the maximum number of characters that will be used for file names on directory listings.

Example

```
DirShowMaxLength 30
```

Initial configuration file setting

```
DirShowMaxLength 25
```

DirShowMinLength - Set the minimum length for file names on directory listings

Use this directive to set the minimum number of characters that will always be reserved for file names on directory listings. Lengths of fully-qualified file names
in the directory can exceed this number. However, file names cannot be longer than the number specified on the DirShowMaxLength directive.

**Example**

\[ \text{DirShowMinLength 10} \]

**Initial configuration file setting**

\[ \text{DirShowMinLength 15} \]

**DirShowMode - Show file permissions on directory listings**

Use this directive to specify whether directory listings should include the permission bits for each file.

**Example**

\[ \text{DirShowMode On} \]

**Initial configuration file setting**

None.

**Program default setting**

\[ \text{DirShowMode Off} \]

**DirShowOwner - Show file owner on directory listings**

Use this directive to specify whether directory listings should include the owner ID for each file.

**Example**

\[ \text{DirShowOwner On} \]

**Initial configuration file setting**

None.

**Program default setting**

\[ \text{DirShowOwner Off} \]

**DirShowSize - Show file size on directory listings**

Use this directive to specify whether directory listings should include the size of each file.

**Example**

\[ \text{DirShowSize Off} \]

**Initial configuration file setting**

\[ \text{DirShowSize On} \]

**IconPath - Specify the path for the directory listing internal icons**

Use this directive to specify the path where the icons you want to use on directory listings are stored. You can use this directive if you have a group of servers that you want to share the same set of icons.

**Attention:** This directive must be before any of the other icon directives (AddBlankIcon, AddDirIcon, AddParentIcon, AddUnknownIcon, and AddIcon).
Directories/Welcome Page directives

Example

IconPath http://icon.server.com:8080/httpd-internal-icons/

In the above example, each request for a directory list icon generates a request to a server named icon.server.com.

Initial configuration file setting

IconPath server_root/icons/*

Program default setting

server_root /icons/*

Welcome - Specify names of welcome files

Use this directive to specify the name of a welcome file the server should look for to respond to requests that do not contain a specific file name. You can build a list of welcome files by putting multiple occurrences of this directive in the configuration file.

For requests that do not contain a file name or a directory name, the server always looks in the document root directory for a file that matches a name specified on a Welcome directive. If a match is found, the file is returned to the requester.

For requests that contain a directory name but not a file name, the AlwaysWelcome directive controls whether the server looks in the directory for a welcome file to return. By default, AlwaysWelcome is set to a value of On. This means the server always looks in the requested directory for a file matching a name specified on a Welcome directive. If a match is found, the file is returned to the requester.

If the server finds more than one match between files in a directory and file names on Welcome directives, the order of the Welcome directives determines which file is returned. The server uses the Welcome directive closest to the top of the configuration file.

If the server does not find a welcome file in the directory, the DirAccess directive controls whether or not the server responds to the request with a directory listing.

The format of the Welcome directive is:

Welcome file-name [Server-IP-address or hostname]

file-name

A file name you want to define as being a welcome file.

Server-IP-address or hostname

If you are using multiple IP addresses or virtual hosts, use this parameter to specify an IP address or a host name. (For more information on using multiple IP addresses or virtual hosts, see Chapter 16, Running your server with multiple IP addresses or virtual hosts on page 163.) The server uses the directive only for requests that come to the server on this IP address or for this host. For an IP address, this is the address of the server’s network connection, not the address of the requesting client.

You can specify an IP address (for example, 204.146.167.72) or you can specify a host name (for example, hostA.bcd.com).
Directories/Welcome Page directives

This parameter is optional. Without this parameter, the server uses the directive for all requests regardless of the IP address the requests come in on or the host name in the URLs.

A wildcard character cannot be specified for a server’s IP address.

Examples:

Welcome letsgo.html
Welcome Welcome.html

The above example defines two welcome pages and assumes the AlwaysWelcome directive is set to its default of On. For requests that do not contain a file name, the server would try to return a welcome file from the directory specified on the request (or document root directory if the request does not specify a file name or a directory). The server would first look for a file named letsgo.html. If the directory does not have a letsgo.html file, the server would look for a file named Welcome.html.

Welcome CustomerA.html 9.67.106.79
Welcome CustomerB.html 9.83.100.45

Your server would look for different welcome files based on the IP address of the network connection the request comes in on. For requests coming in on 9.67.106.79 the server would look for welcome files named CustomerA.html. For requests coming in on 9.83.100.45, the server would look for welcome files named CustomerB.html. If the request comes in on a different IP address, the server looks for the default address.

Welcome CustomerA.html hostA.bcd.com
Welcome CustomerB.html hostB.bcd.com

Your server would look for different welcome files based on the host name in the URL. For requests coming in for hostA, the server would look for welcome files named CustomerA.html. For requests coming in for hostB, the server would look for welcome files named CustomerB.html. If the request comes in for a different host, the server looks for the default host name.

Initial configuration file setting

Welcome Welcome.html
Welcome welcome.html
Welcome index.html
Welcome Frntpage.html

The above default values are shown in the order used by the default configuration.

User directories

Use the directive described in this section to control whether individual users of your server can have their own private Web documents.

UserDir - Enable users to have private Web documents

Use this directive to allow individual users of your server to have their own private Web documents. The user name must reference a defined OS/390 UNIX user ID that includes a home directory definition. The value on the directive specifies the name of a subdirectory within each user’s home directory. When the server receives a URL request that begins with /UserName/, the server looks for the requested object in the subdirectory name specified on the UserDir directive of the user’s home directory.
Before allowing local users to have their own private documents on the Web, you may want to consider the issues of security and corporate image.

**Example**

```
UserDir public
```

In this example, if the server receives the request http://MyServer/joe, the server looks in the joe/public directory and returns the appropriate welcome document (for example, index.html).

**Default**

The default configuration does not allow user-supported directories.

---

**Error messages - Customize Web server error messages**

Use this directive to customize the messages your server sends to the requesting client when it encounters an error condition. For example, you can change a message to include more information about the cause of the problem and suggest possible solutions to fix it. For internal networks, you might provide a contact person for your users to call.

Each error condition is identified by a keyword. To decide which error messages you want to customize, first review the list of error conditions, their causes, and the default message that the server sends. Then, for each error message you want to change:

- Create an individual HTML file with the desired text.
- Add an ErrorPage directive to your configuration file that associates the error condition keyword with the HTML file you want to serve.

**Note:** The server does not parse your error files for imbeds, regardless of the file extensions or use of the Imbeds directive.

---

**ErrorPage - Specify a customized message for a particular error condition**

Use this directive to specify the name of a file that you want to send when the server encounters a particular error condition.

You can place this directive anywhere in the configuration file. When the error occurs, the file will be processed according to the mapping rules defined in your configuration file. Therefore, the file you want to send must be in a location that can be reached through the mapping rules as defined by the Fail, Map, NameTrans, Pass, Redirect, Service directives. At a minimum, you need a Pass directive that would allow the server to pass the error message file.

The format of this directive is:

```
ErrorPage keyword /path/filename.html
```

- **keyword**
  One of the keywords associated with an error condition. See "Error Conditions, Causes, and Default Messages" on page 306 for a list of keywords.

- **/path/filename.html**
  This is the fully qualified Web name of your error file, as viewed by a client on the Web.
Error message customization directives

Example

```
ErrorPage scriptstart /errors/html/scriptstart.html
```

In the above example, when a `scriptstart` condition is encountered, the server will send the `scriptstart.html` file found in the `/errors/html` directory to the client.

This file might contain the following HTML text:

```
<HTML>
<HEAD>
<TITLE>Message for SCRIPTSTART condition</TITLE>
</HEAD>
<BODY>
The CGI program could not be started.
</BODY>
</HTML>
```

If the directive that matches the above path in the server’s configuration file is `PASS /* /wwwhome/*`, then the full path for this message file would be `/wwwhome/errors/html/scriptstart.html`.

Default

If you do not specify an `ErrorPage` directive for an error condition, the server’s default error message for that condition will be sent.

Error Conditions, Causes, and Default Messages

The following list shows the HTTP status code and keyword for each error condition, followed by the probable cause, and the default message the server sends.

<table>
<thead>
<tr>
<th>Code and Keyword</th>
<th>Cause and Default Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>302 okredirect</td>
<td>Cause: The requested file is on another recognized server. The name of the server is sent back to the requesting client along with a message. The client can connect to the correct server or display the message that is sent. Default message: Found.</td>
</tr>
<tr>
<td>400 badrequest</td>
<td>Cause: Either there is a network problem, such as a time-out, or the request was indecipherable. Default message: Invalid request - completely unable to parse it.</td>
</tr>
<tr>
<td>400 badscript</td>
<td>Cause: The server could determine that the requested file was a CGI script but it could not process it; the request was invalid in some way. Default message: The script execution request is not valid.</td>
</tr>
<tr>
<td>400 connectfail</td>
<td>Cause: On a tunneled request, the server could not connect to the requested partner on the requested port. Default message: Host not found or not responding.</td>
</tr>
<tr>
<td>400 nopartner</td>
<td>Cause: On a tunneled request, the server could not connect to the requested hostname due to bad syntax or an unknown host.</td>
</tr>
</tbody>
</table>
Error message customization directives

Default message: Host not found or not responding.

400 proxyfail
Cause: The client is trying to use the server as a proxy, and although this is allowed, it did not work. Possibly the destination server doesn’t exist or is busy.

Default message: Proxy load failed.

400 unknownmethod
Cause: The request did not include a recognized method, such as GET, POST, PUT, or DELETE.

Default message: The request is not valid or not recognized.

401 notauthorized
Cause: The request requires a user ID and password. Either the user ID and password sent by the client are not valid for this request or the client did not send a user ID and password.


401 notmember
Cause: The requested file has a protection rule listing valid user IDs and passwords and the user ID of the requesting client is not included in that list.

Default message: Not authorized to access the document.

401 pwchanged
Cause: The user ID has been changed to use the new password you entered. Enter the new password again to correct your browser’s password cache.

Default message: Password changed. Enter newpw to continue.

401 pwexpired
Cause: The password for the MVS user ID has expired.

Default message: Access denied - password expired. Enter oldpw/newpw/newpw to change your password.

401 pwnewinv
Cause: The password you entered did not meet the password format defined in the installation rules.

Default message: New password format not valid, try again. Enter oldpw/newpw/newpw to change your password.

401 pwnewneq
Cause: The two passwords you entered for newpw do not match.

Default message: New passwords are not equal, try again. Enter oldpw/newpw/newpw to change your password.

403 baduser
Cause: The client requested a user’s home directory that does not exist.

Default message: The user directory is not valid.

403 badredirect
Cause: The server is trying to redirect the request and the Redirect directive is invalid (possibly missing a destination) or contains a loop.

Default message: The redirection in the configuration file is not valid.
Error message customization directives

403 byrule
   Cause: Either the file requested is specifically blocked by a Fail directive or it does not match any of the files that are allowed to be accessed according to other request mapping directives.
   Default message: Forbidden by rule.

403 dirbrowse
   Cause: The client specified a directory (rather than a file name) in the URL that does not have a welcome page and the administrator has turned off directory browsing (either for this directory or for the entire server).
   Default message: Directory browsing failed - access forbidden.

403 dotdot
   Cause: The client request contains an instruction (/../) to navigate above the document directory root and this is not allowed.
   Default message: Forbidden - URL containing .. forbidden (don’t try to break in).

403 ipmask
   Cause: The file requested has a protection rule that includes a list of valid IP addresses and the client’s address is not included in the list.
   Default message: Server will not serve to your IP address.

403 ipmaskproxy
   Cause: The client is trying to use the server as a proxy and the client is not included in the list of host names or IP addresses that are allowed to do so.
   Default message: Proxy server will not serve to your IP address (at least with this HTTP method).

403 methoddisabled
   Cause: The client requested a method (such as GET, POST, PUT, DELETE) that is specifically not allowed by the Disable directive.
   Default message: Method method is disabled on this server.

403 noacl
   Cause: The directory has a protection rule but does not have an Access Control List (ACL) defined and the protection setup does not have a GetMask subdirective. The administrator needs to remove the protection rule or add an ACL.
   Default message: Access to this file is not allowed ‘no ACL file’.

403 noentry
   Cause: The directory is protected by an Access Control List (ACL) and the user is not included in the ACL.
   Default message: Access to this file is not allowed (no ACL entry).

403 notallowed
   Cause: The requested file was found but the server’s protection setup prevented access. This is commonly generated for URLs that point to CGI programs.
   Default message: The PUT and DELETE methods must be specified in the server’s protection setup.

403 openfailed
   Cause: After passing the protection rules, the server determined that the
Error message customization directives

client should have read access to the file but the operating system will not allow the server to access it. Possibly the user ID running the server does not have read permission to the file it is trying to serve or the file system may be encountering problems.

Default message: Can’t browse selected file.

403 setuperror
Cause: The directory has an Access Control List (ACL) defined but does not have a protection rule. The administrator needs to add a protection rule or remove the ACL.

Default message: Server protection setup error occurred. Probably, the protection setup file was not found or it contained a syntax error.

404 multifail
Cause: The requested file could not be found on the server. The server tried to match the file name exactly as specified and with every known file extension appended.

Default message: The file was not found, even after searching on any extensions to the file name.

406 notacceptable
Cause: A request was submitted that matched one or more files found on the server but the accept headers sent with the request did not match exactly. For example, the accept language header asked for English files, but the matching file was French.

Default message: Not Acceptable - no file exists that matches the accept headers.

407 proxypwchanged
Cause: The user ID has been changed to use the new password you entered. Enter the new password again to correct your browser’s password cache.

Default message: Password changed. Enter newpw to continue.

407 proxypwexpired
Cause: The proxy password for the MVS user ID has expired. The password for the MVS proxy user ID has expired.

Default message: Access denied - password expired. Enter oldpw/newpw/newpw to change your password.

407 proxynotauth
Cause: The proxy request requires a user ID and password. Either the user ID and password sent by the client are not valid for this request or the client did not send a user ID and password. Note that some Web browsers do not support the PROXY-AUTHENTICATE function.

Default message: Not authorized. Proxy-Authentication failed (or your browser does not support it).

407 proxynotmember
Cause: The proxy request has a protection rule listing valid user IDs and the user ID of the requesting client is not included in that list.

Default message: Not authorized for proxy access to the document.

407 proxypwnewinv
Cause: The password you entered did not meet the password format defined in the installation rules.
Error message customization directives

Default message: New password format not valid, try again. Enter oldpw/newpw/newpw to change your password.

407 proxypwnewneq
Cause: The new password you entered is not correct. The two passwords you entered for newpw do not match.

Default message: New passwords are not equal, try again. Enter oldpw/newpw/newpw to change your password.

412 preconfail
Cause: A precondition specified by the client on this request was not met. For example, this could result from an HTTP/1.1 request with a condition "if-not-modified-since xxx".

Default message: Precondition failed: could not match entity tags.

416 badrange
Cause: A PUT request either has an invalid content range header or it has incorrect information in the content range header for the file being processed. For example, the starting byte range of the associated content exceeds the existing file size. Note that a HTTP response code of 501 is returned if the content header cannot be parsed.

Default message: Invalid request - Content range is incorrect.

417 expectfailed
Cause: A request was submitted with an expect header but the server could not understand or honor the expectation sent in the header.

Default message: Expectation failed.

500 addrspacedirty
Cause: The BPX.SERVER FACILITY or BPX.DAEMON FACILITY is defined and a program or DLL has been loaded into the server’s address space that is not under PROGRAM CONTROL. The server’s authority to check passwords and set access control user IDs has been temporarily revoked. You must stop the server, correct the problem, and start the server again. For information about controlling programs used with the server, see “Chapter 1. Planning for installation” on page 3.

Default message: Access denied - unauthorized program loaded.

500 setupsurrogate
Cause: A surrogate user ID is defined in the configuration file, but the server does not have permission to use this user ID as a surrogate. For information about creating surrogate user IDs, see “Chapter 1. Planning for installation” on page 3.

Default message: Access denied - surrogate user setup error.

500 scriptio
Cause: The client requested a CGI script; the server can find it and start it but cannot get it to process input or output. The script may contain invalid code.

Default message: Cannot read script output pipe.

500 scriptnotfound
Cause: The client requested a CGI script that cannot be found.

Default message: The script request is not valid; none of <program> and <program>.pp is executable.
Error message customization directives

500 scriptstart
Cause: The client requested a CGI script; the server can find it but cannot start it. The script may contain invalid code.
Default message: Starting the CGI program failed. Could not communicate with the CGI program.

500 systemerror
Cause: An internal MVS error occurred using SAF services. See trace table error information.
Default message: Access denied - system error using SAF.

501 noformat
Cause: The server has encountered an internal error and cannot interpret the format of the file it is trying to serve. The file may be corrupted or have an unknown or invalid file extension.
Default message: Sorry, can’t convert from mime-type-1 to mime-type-2.

503 service unavailable
A servlet request thread is unavailable at the time of the servlet request.

GWAPI - Specify GWAPI applications for processing

The Go Webserver Application Programming Interface (GWAPI) allows you to extend the HTTP Server’s base functions with your own customized processing routines. Use the directives described in this section to have the server call the application functions in your program at various points in its request processing cycle.

For detailed information on writing the application functions and compiling your program, see “Chapter 18. Writing GWAPI programs” on page 197.

Except for Service and NameTrans, these directives can be in any order in the configuration file and you do not need to include every directive. If you do not have a customized application function for a particular step, just omit the corresponding directive. The normal processing for that step will execute by default.

The Service and NameTrans directives behave like the other mapping directives and are sensitive to their placement in the configuration file.

You can also have more than one configuration directive for a step. For example, you could include two NameTrans directives, each pointing to a different application function. When the server performs the name translation step, it will process your name translation functions in the order in which they appear within the configuration file.

Your application functions do not have to be executed for every request:
• By specifying a URL with some directives, you can indicate that you want the application function called only for URLs that match a certain pattern or mask.
• By specifying an authentication scheme with the Authentication directive, you can indicate that you want the application function called only for certain types of authentication.
GWAPI application processing directives

**DebugToolAddr - Identify the workstation running the Remote Debugger**

Use this directive to identify the workstation running the OS/390 Debug Tool graphical user interface (Remote Debugger). The Debug Tool is a Language Environment (LE) tool that can be used to debug your C/C++ GWAPI programs.

For information on using the Debug Tool with the Web server, see [Debugging C/C++ GWAPI programs](#) on page 226.

The format of the directive is:

```
DebugToolAddr IP_address or hostname [port_number]
```

For `IP_address or hostname`, enter the IP address or hostname of the workstation that is running the Remote Debugger. The default port number is 8000; specify a `port_number` if you are using a port other than the default.

**Example**

```
DebugToolAddr 127.20.5.3 8040
```

**Initial configuration file setting**

None.

**ServerInit - Customize the Server Initialization step**

Use this directive to specify a customized application function you want the server to call during its initialization routines. This code will be executed before any client requests are read and whenever the server is restarted.

If you use this directive to preload frequently accessed data sets or PDS members, see [Appendix F. GWAPI MVSDS DLL Service](#) on page 405 for more information.

The format of the directive is:

```
ServerInit /path/file:function_name
```

`/path/file`

The fully qualified file name of your compiled program, including the extension.

`function_name`

The name you gave your application function within your program.

**Example**

```
ServerInit /ics/api/bin/icsext05.so:svr_init
```

**Initial configuration file setting**

None.

**PreExit - Customize the PreExit step**

Use this directive to specify a customized application function you want the server to call during the PreExit step. This code will be executed after a client request has been read but before any other processing occurs.

The format of the directive is:

```
PreExit /path/file:function_name
```

`/path/file`

The fully qualified file name of your compiled DLL, including the extension.
function_name

The name you gave your application function within your program.

Example

PreExit /ics/api/bin/icsext05.so:pre_exit

Initial configuration file setting
None.

Authentication - Customize the Authentication step

Use this directive to specify a customized application function you want the server to call during the Authentication step. This code will be executed based on the authentication scheme. Currently, only Basic authentication is supported.

Note: Authentication is part of the authorization process; it only occurs when authorization is required.

The format of the directive is:

Authentication type /path/file:function_name

type
Specifies an authentication scheme which further determine if your application function is called. Both an asterisk (*) and Basic are accepted values.

/path/file
The fully qualified file name of your compiled program, including the extension.

function_name
The name you gave your application function within your program.

Example

Authentication BASIC /ics/api/bin/icsextpgm.so:basic_authentication

Initial configuration file setting
None.

NameTrans - Customize the Name Translation step

Use this directive to specify a customized application function you want the server to call during the Name Translation step. This code would supply the mechanism for translating the virtual path in the request to the physical path on the server, mapping URLs to specific objects.

Note: This is not a terminal mapping rule. The transformed URL still has to match one of the terminal mapping rule directives, such as Exec, Fail, Map, Pass, Redirect, and Service.

The format of the directive is:

NameTrans request-template /path/file:function_name

[Server-IP-address or hostname]

Note: The directive must be typed on one line, even though it is shown here on two lines.

request-template
A template for requests that further determine if your application function is called. The specification can include the protocol, domain and host, can be
GWAPI application processing directives

preceded by a slash (/), and can use an asterisk (*) as a wildcard. For example, /front_page.html, http://www.ics.raleigh.ibm.com, /pub*, /*, and * are all valid.

/path/file
The fully qualified file name of your compiled program, including the extension.

function_name
The name you gave your application function within your program.

Server-IP-address or hostname
If you are using multiple IP addresses or virtual hosts, determines if your application function will be called only for requests coming in on a specific IP address or for a specific host.

A wildcard character cannot be specified for a server's IP address.

Example
NameTrans /index.html /api/bin/icsextpgm.so:trans_url

Initial configuration file setting
None.

Authorization - Customize the Authorization step

Use this directive to specify a customized application function you want the server to call during the Authorization step. This code would verify that the requested object can be served to the client.

The format of the directive is:

Authorization request-template /path/file:function_name

request-template
A template for requests that further determine if your application function is called. The specification can include the protocol, domain and host, can be preceded by a slash (/), and can use an asterisk (*) as a wildcard. For example, /front_page.html, http://www.ics.raleigh.ibm.com, /pub*, /*, and * are all valid.

/path/file
The fully qualified file name of your compiled program, including the extension.

function_name
The name you gave your application function within your program.

Example
Authorization /index.html /api/bin/icsextpgm.so:auth_url

Initial configuration file setting
None.

ObjectType - Customize the Object Type step

Use this directive to specify a customized application function you want the server to call during the Object Type step. This code would locate the requested object in the file system and identify its MIME type.

The format of the directive is:
GWAPI application processing directives

Object Type request-template /path/file:function_name

**request-template**
A template for requests that further determine if your application function is called. The specification can include the protocol, domain and host, can be preceded by a slash (/), and can use an asterisk (*) as a wildcard. For example, /front_page.html, http://www.ics.raleigh.ibm.com, /pub*, /*, and * are all valid.

**/path/file**
The fully qualified file name of your compiled program, including the extension.

**function_name**
The name you gave your application function within your program.

**Example**
Object Type /index.html /api/bin/icsextpgm.so:obj_type

**Initial configuration file setting**
None.

**Service - Customize the Service step**
Use this directive to specify a customized application function you want the server to call during the Service step. This code would service the client request. For example, it sends the file or runs the CGI program.

Once a request matches a template on a Service directive, the request is not compared to request templates on any subsequent directives.

There is no default for this directive. If the request matches a Service rule (an application function specified on a Service directive is executed) but it returns HTTP_NOACTION, the server will generate an error and the request will fail.

The format of the directive is:

```
Service request-template
/path/file:function_name [Server-IP_address or hostname]
```

**Note:** The directive must be typed on one line, even though it is shown here on more than one line.

**request-template**
A template for requests that further determine if your application function is called. The specification can include the protocol, domain and host, can be preceded by a slash (/), and can use an asterisk (*) as a wildcard. For example, /front_page.html, http://www.ics.raleigh.ibm.com, /pub*, /*, and * are all valid.

**/path/file**
The fully qualified file name of your compiled program, including the extension.

**function_name**
The name you gave your application function within your program.

**Server-IP_address or hostname**
If you use multiple IP addresses or virtual hosts, determines if your application function will be called only for requests coming in on a specific IP address or for a specific host.
GWAPI application processing directives

A wildcard character cannot be specified for a server’s IP address.

**Note:** If you want full path translation, including *query_string*, you must have an asterisk (*) in both the *request-template* and in the */file/path:function_name* as shown in the second example.

**Example**

Service /index.html /ics/api/bin/icsext05.so:serve_req

Service /cgi-bin/hexcalc* /ics/api/calculator:HEXcalc*

**Initial configuration file setting**

None.

**PICSDBLookup - Customize the PICS label retrieval step**

Use this directive to specify a customized application function you want the server to call to retrieve PICS labels for a specified URL. Your function can either dynamically create a PICS label for the requested document or search for a PICS label in an alternative file or database.

The format of the directive is:

```
PICSDBLookup /path/file:function_name
```

* /path/file
  The fully qualified file name of your compiled program, including the extension.

* function_name
  The names you gave your application functions within your program.

**Example**

PICSDBLookup /api/bin/icsext05.so:get_pics

**Initial configuration file setting**

None.

**DataFilter - Customize the Data Filter step**

Use this directive to specify a customized application function you want the server to call during the Data Filter step. This code would provide three application functions:

- An *open* function to perform any initialization prior to processing the data
- A *write* function to process the data
- A *close* function to perform any clean up activities

You can only have one DataFilter active for each instance of the server.

The format of the directive is:

```
DataFilter /path/file:function_name:function_name:function_name
```

* /path/file
  The fully qualified file name of your compiled program, including the extension.

* function_names
  The names you gave your application functions within your program. You will need to supply the name of the open, write, and close functions.
GWAPI application processing directives

Example

DataFilter /ics/bin/icsext05.so:open_data:write_data:close_data

Initial configuration file setting
None.

Log - Customize the Log step

Use this directive to specify a customized application function you want the server to call during the Log step. This code would supply logging and other processing you want performed after the connection has been closed.

The format of the directive is:

Log request-template /path/file:function_name

request-template
A template for requests that further determine if your application function is called. The specification can include the protocol, domain and host, can be preceded by a slash (/), and can use an asterisk (*) as a wildcard. For example, /front_page.html, http://www.ics.raleigh.ibm.com, /pub*, /*, and * are all valid.

/path/file
The fully qualified file name of your compiled program, including the extension.

function_name
The name you gave your application function within your program. You must supply the names of the open, write, and close functions.

Example

Log /index.html /api/bin/icsextpgm.so:log_url

Initial configuration file setting
None.

Error - Customize the Error step

Use this directive to specify a customized application function you want the server to call during the Error step. This code would execute only when an error is encountered, to provide customized error routines.

The format of the directive is:

Error request-template /path/file:function_name

request-template
A template for requests that further determine if your application function is called. The specification can include the protocol, domain and host, can be preceded by a slash (/), and can use an asterisk (*) as a wildcard. For example, /front_page.html, http://www.ics.raleigh.ibm.com, /pub*, /*, and * are all valid.

/path/file
The fully qualified file name of your compiled program, including the extension.

function_name
The name you gave your application function within your program.
GWAPI application processing directives

Example

```
Error /index.html /ics/api/bin/icsext05.so: error_rtns
```

Initial configuration file setting

None.

PostExit - Customize the PostExit step

Use this directive to specify a customized application function you want the server to call during the PostExit step. This code will be executed regardless of the return codes from previous steps or other PostExit handlers. It allows you to clean up any resources allocated to process the request.

The format of the directive is:

```
PostExit /path/file:function_name
```

- **/path/file**
  - The fully qualified file name of your compiled program, including the extension.

- **function_name**
  - The name you gave your application function within your program.

Examples:

```
PostExit /ics/api/bin/icsext05.so: post_exit
```

Initial configuration file setting

None.

ServerTerm - Customize the Server Termination step

Use this directive to specify a customized application function you want the server to call during the Server Termination step. This code would execute when an orderly shutdown occurs and whenever the server is restarted. It allows you to release resources allocated by a PreExit application function.

The format of the directive is:

```
ServerTerm /path/file:function_name
```

- **/path/file**
  - The fully qualified file name of your compiled program, including the extension.

- **function_name**
  - The name you gave your application function within your program.

Example

```
ServerTerm /ics/api/bin/icsext05.so: shut_down
```

Initial configuration file setting

None.

LDAP - Set up shared configuration for the server

LDAPInfo - Define an external LDAP server

Use the LDAPInfo directive to provide the server with information about the LDAP servers in which to store information. Storing information on an external LDAP server allows applications to share the same information.
LDAP directives

Note: In the configuration file, you must place LDAPInfo directives before any LDAPInclude directives or protection setups.

The format of the directive is:

```plaintext
LDAPInfo label-name { 
subdirective value 
subdirective value 
subdirective value 
} 

label-name
The name you want to associate with this LDAP server setup. The name can then be used by subsequent LDAPInclude and Protection directives to point to this LDAP setup.

subdirective
Put an LDAPInfo subdirective and its value on each line between the left and right brace. You cannot put any comment lines between the braces.

The LDAPInfo subdirectives are described below.

Example

```plaintext
LDAPInfo PrimaryLdapServer {
 Host ldap.ibm.com
 Transport TCP
 ClientAuthType Basic
 ServerAuthType Basic
 ServerDN "cn=HTTP Server, o=IBM c=US"
 ServerPasswordStashFile "StashFileName"
 UserSearchBase "o=IBM c=US"
 GroupSearchBase "o=IBM c=US"
}
```

LDAPInfo Subdirectives

Following are descriptions of the LDAPInfo subdirectives that can be used in an LDAP server setup. Four groups comprise the LDAPInfo sub-directives: general, timeout, server connection, and client connection.

General subdirectives
Use the following subdirectives to provide general information about the LDAP servers:

Host - Specify LDAP server hostname: Specify the hostname of the LDAP server.

Example:

```plaintext
Host ldap.mycompany.com
```

Transport - Specify LDAP connection protocol: Protocol to use to connect to the LDAP server. Possibilities are TCP(default) and SSL.

Example:

```plaintext
Transport SSL
```
LDAP directives

**Port - Specify LDAP server port number:** Specify the port number the LDAP server listens on. The default port is 389 for TCP-transported connections, and 636 for SSL-transported connections. See "Transport - Specify LDAP connection protocol" on page 319 for more information.

Example:
Port 636

**Timeout subdirectives**

When attempting to reach an external server, there is always a possibility the connection will take a long time to complete. The following subdirectives provide the server with timeout settings for connections to the LDAP server:

**IdleConnTimeout - Specify how long to keep an idle LDAP connection:**
optional - Time during which to leave an idle LDAP connection open. The default is 10 minutes

Example:
IdleConnTimeout 10 minutes

**WaitToRetryConnTime - Specify how long to wait before retrying LDAP connection:** Amount of time to wait between unsuccessful attempts to connect to a server

Example:
WaitToRetryConn 5 minutes

**SearchTimeout - Specify maximum LDAP search time:** Time limit to wait for an LDAP search request to complete. The default is 10 seconds

Example:
SearchTimeout 2 minutes

**CacheTimeout - Specify LDAP expiry time:** Time-out of a cached entry. To reduce query time, the LDAP server caches responses. Use this directive to specify how long the server will return the cached copy as opposed to generating a new query. The default is ten minutes.

Example:
CacheTimeout 5 minutes

**Server connection subdirectives**

The following subdirectives provide the HTTP Server with parameters for establishing LDAP connections in order to read access control information.

**ServerAuthType - Specify server LDAP authentication type:** Authentication type for the server’s connection to the LDAP server. Values are:
1. **None**, if the LDAP server will allow anonymous access
2. **Basic**, if the HTTP Server logs in to the LDAP server. It uses its distinguished name from the **ServerDN** subdirective, and the password provided in the password stash file.

Example:
ServerAuthType Basic
ServerDN - Specify distinguished name of the Web server: Distinguished name of the HTTP Server. This name is used when accessing the LDAP server when ServerAuthType is set to Basic.

Example:
ServerDN MyLdapServer

ServerPasswordStashFile - Specify LDAP password stash file: The file containing the encrypted password to access the LDAP server. The password is only used if ServerAuthType is Basic. Create the stash file using the htadm command or using the Configuration and Administration forms. See "htadm command" on page 246 for more information.

Example:
ServerPasswordStashFile /usr/lpp/internet_base/server/passwordFile

Client connection subdirectives
The following subdirectives provide the HTTP Server with parameters for establishing an LDAP connection on behalf of the client.

ClientAuthType - Specify client LDAP authentication type: Specify the authentication type for connections made on behalf of the client. Values are:
1. Basic - the client must provide a userid and password in order to authenticate
2. Cert - the client’s certificate is used to authenticate.

Note: In order for the client to successfully authenticate, the client must have used the https protocol, SSLClientAuth must be turned on, SSL client authentication by the HTTP Server must be successful, and a search for the client’s entry (using UserCertFilter) must be successful.
3. BasicIfNoCert - client’s certificate is used for authentication. However, if https is not used, SSLClientAuth is not on, or if SSL client authentication by the HTTP Server fails, then basic authentication is used.

Example:
ClientAuthType BasicIfNoCert

UserSearchBase - Specify LDAP user search root: Specify the starting point for the LDAP server to search for user names.

Example:
UserSearchBase "o=IBM, c=US"

UserNameFilter - Specify LDAP user search filter: Specify the filter used to convert the username as input by the user to a search filter for an LDAP entry. The default is "(&(objectclass=person)(cn=%v1* %v2*))" where %v1 and %v2 are the words typed by the user.

For example, if the user types "Pa Kel", the resulting search filter would be "(cn=Pa* Kel*)". Search filters are described in "LDAP search filters" on page 157.

However, if multiple matching entries are returned, the HTTP Server does not know which to use, and fails authentication. If there were entries (cn=Paul Kelsey) AND (cn=Paula Kelly), the above search string will return both entries and fail to authenticate.

Example:
LDAP directives

**UserNameFilter** "(cn=%v1* %v2*)"

**UserNameFieldSep - Specify delimiters for user data:** Specify the set of characters used to separate the user's input into fields. The default value and the example below delimits data using a space, a comma, and the tab character.

*Example:
 UserNameFieldSep " \t,"*

**UserCertFilter - Specify LDAP certificate search filter:** Specify the certificate filter which converts the information in the client certificate passed over SSL to a search filter for an LDAP entry. The default is
"(&(objectclass=person)(cn=%v1)(ou=%v2)(o=%v3)(c=%v4))". SSL certificates include the following fields, all of which can be converted to a search filter:
1. common name
2. organizational unit
3. organization
4. country
5. locality
6. state or country
7. serial number

*Example:
 UserCertFilter "(cn=%v1)"*

**GroupSearchBase - Specify LDAP group search root:** Specify the starting point for the LDAP server to search for group entries.

*Example:
 GroupSearchBase "o=IBM, c=US"*

**GroupNameFilter - Specify LDAP group search filter:** Specify the filter LDAP uses to search for group names.

*Example:
 GroupNameFilter "(&(objectclass=groupNames)(objectclass=groupOfUniqueNames))"*

**GroupMemberAttrs - Specify attributes returned for group members:** Specify the attributes of the group which contain member information. Many attributes may be specified separated by a comma. The default value and the example below show that member information is available in both the member and the uniqueMember attributes.

*Example:
 GroupMemberAttrs "member,uniqueMember"*

**KeyFileName - Specify LDAP key database:** Specify the key database to use if the transport is SSL. The default value is the file specified by the KeyFile directive in the httpd configuration file.

*Example:
 KeyFileName /usr/lpp/internet_server/base/key.kdb*
**KeyLabel - Specify label of the certificate used for SSL connections:** Specify a label for the certificate that the HTTP Server uses to authenticate with the LDAP server. This field is required if the transport is SSL and there is no default value.

*Example:*

```
KeyLabel "My Server's Certificate"
```

**LDAPInclude - Retrieve configuration file information from the LDAP server**

Use the `LDAPInclude` directive to provide the server with locations of configuration file information stored on the LDAP server.

The `LDAPInfo` directive must precede the `LDAPInclude` directive.

**Format**

```
LDAPInclude label filter attribute
```

- **label**
  - name of the LDAP server setup defined in the `LDAPInfo` directive. The label tells the HTTP Server which LDAP server to use to locate configuration information related to the `attributes`.

- **filter**
  - LDAP search filter described in "LDAP search filters” on page 157, and RFC 1960.

- **attribute**
  - the name of the attribute whose value is some arbitrary part of the configuration file.

*Example*

```
LDAPInclude PrimaryLdapServer "(cn=web config)" description
```

The example will retrieve the value of the `description` attribute from a single entry with the common name `web config` on the server associated with `PrimaryLdapServer`.

**Initial configuration file setting**

None.

---

**Logging and Reporting - Customize logs and generate reports**

Use the directives described in this section to control your server’s logs. You can have the server log incoming requests and server errors. If your server is running as a caching proxy server, you can log access requests for files from the proxy server’s cache.

You can specify the path and file where you want these logs to be kept, how requests should be logged, and which requests you do not want to log.

You most likely will want to use the common log format. This is the default format and it is the same format used by most other types of Web servers. If you plan to use the access log report function, you are required to use the common log format.

*Note:* To use a third-party reporting program, set the DoReporting directive on, then use the `LoggingReportingProgram` and
Logging and Reporting directives

LoggingReportingProgramOptions directives. For more information, see "LoggingReportingProgram — Specify the reporting program to be used" on page 339.

AccessLog - Name the path for the access log file

Use this directive to specify the place where the server logs all requests made by the client.

The server starts a new log file each day at midnight if it is running. Otherwise, the server starts a new log file the first time you start it on a given day. When creating the file, the server uses the file name you specify and appends a date suffix. The date suffix is in the format \textit{Mmmddyyyy}, where \textit{Mmm} is the first three letters of the month; \textit{dd} is the day of the month; and \textit{yyyy} is the year.

It is a good idea to remove old log files, because they can take up a significant amount of space. For information about removing old log files, refer to "AccessLogArchive - Remove existing access, agent, or referer log files or run a user exit".

Example

\begin{verbatim}
AccessLog logs/accesslog
\end{verbatim}

Initial configuration file setting

\begin{verbatim}
AccessLog /usr/lpp/internet/server_root/logs/httpd-log
\end{verbatim}

The example gives you \texttt{ServerRoot/logs/httpd-log.date_suffix.file_extension}. In other words, if \texttt{ServerRoot} is \texttt{/usr/lpp/internet/server_root/} and the server is started on January 15, 1996 the result is:

\begin{verbatim}
\end{verbatim}

where \texttt{extension} is the web server generated file extension.

Program default setting
None.

AccessLogArchive - Remove existing access, agent, or referer log files or run a user exit

Values specified on the AccessLogArchive directive apply to access, agent, and referer logs. The collective size includes the size of all access logs or all agent logs or all referer logs, not the collective size of all types of logs.

At midnight each night, the server closes the current log and creates a new log file for the coming day. You can choose to do one of the following actions with the closed logs:

- Remove log files of a given age or when a given amount of storage is used by the collection of log files
- Allow closed logs to remain on your file system
- Branch to a user exit

To remove access, agent, or referer logs of a given age, specify this directive, in addition to the AccessLogExpire directive. To remove logs when their collective size exceeds a certain amount of storage, specify this directive, in addition to the AccessLogSizeLimit directive.
To allow closed logs to remain on your file system, you can accept the default, which is `AccessLogArchive none`.

To branch to a user exit, specify the path to the user exit and any parameters for the user exit on the `AccessLogArchive` directive. The server will append to this directive the path to the access, agent, or referer log.

The `AccessLogArchive` directive can be specified in any of the following formats:

- `AccessLogArchive purge`
- `AccessLogArchive none`
- `AccessLogArchive userexit path_to_the_user-exit_program` [parameters for the user-exit]

**Note:** The directive must be typed on one line, even though it is shown here on two lines.

- **purge**
  Remove access log files of a given age or when their collective size exceeds a given amount of storage.

- **none**
  Do not remove access log files. `none` is the default.

- **userexit**
  Specifies the path of the user-exit program you want to branch to. You can optionally specify the parameters for your user-exit program, as shown in the following examples. The server appends the path to the access log to the directive.

**Examples:**

```
AccessLogArchive purge
AccessLogArchive none
AccessLogArchive userexit /u/userxyz/bin/backup/backup.rexx -d -a
```

For the `AccessLogArchive userexit` example, the user exit invocation is:

```
AccessLogArchive userexit /u/userxyz/bin/backup/backup.rexx -d -a /www/logs/httpd-log
```

**Initial configuration file setting**

`AccessLogArchive none`

**Program default setting**

None.

**AccessLogExcludeURL - Suppress log entries for specific files or directories**

Use this directive to specify that you do not want to log access requests made for specific files or directories that match a given URL template. For example, you might not want to log access requests for GIF files or you might not want to log access requests to a particular file or directory on your server.

You can have multiple occurrences of this directive in your configuration file. You can also put multiple entries for the same directive if you separate them by one or more spaces.
Logging and Reporting directives

Example
AccessLogExcludeURL *.gif

AccessLogExcludeURL /Freebies/*

Initial configuration file setting
None. The server includes in the access log requests for all files and directories.

Program default setting
None.

AccessLogExcludeMethod - Suppress log entries for files or directories requested by a given method
Use this directive to specify that you do not want to log access requests made for files or directories by using a specific method. For example, you might not want to log DELETE requests for files or directories.

You can have multiple occurrences of this directive in your configuration file. You can also put multiple methods on the same directive if you separate them by one or more spaces.

Example
AccessLogExcludeMethod GET
AccessLogExcludeMethod PUT
AccessLogExcludeMethod POST
AccessLogExcludeMethod DELETE

Initial configuration file setting
None. The server includes in the access log the files and directories requested by all types of methods.

Program default setting
None.

AccessLogExcludeMimeType - Suppress log entries for specific MIME types
Use this directive to specify that you do not want to log access requests made for directories or files of a given MIME type. (Examples of MIME types are text/html, image/gif, and image/jpeg.) For example, you might not want to log access requests for GIF images.

You can have multiple occurrences of this directive in your configuration file. You can also put multiple MIME types on the same directive if you separate them by one or more spaces.

Example
AccessLogExcludeMimeType image/gif

Initial configuration file setting
None. The access log includes requests to the server for files and directories of all MIME types.

Program default setting
None.
**AccessLogExcludeReturnCode - Suppress log entries for specific return codes**

Use this directive to specify that you do not want to log access requests that fall within a given range of error code numbers. These error code numbers are HTTP status codes. You cannot specify individual codes. For example, specifying 300 indicates that you want to exclude all access requests that fall within the range 300-399. To exclude these requests, you would specify:

```
AccessLogExcludeReturnCode 300
```

You can have multiple occurrences of this directive in your configuration file. You can also put multiple return codes on the same directive if you separate them by one or more spaces.

**Example**

```
AccessLogExcludeReturnCode 300
```

**Initial configuration file setting**

None. The access log includes all requests to the server, regardless of the code.

**Program default setting**

None.

---

**AccessLogExpire - Remove existing access log files when they reach a given age in days**

Use this directive to specify that you want to remove access log files when they reach a certain age (in days).

This directive requires that you also specify the `AccessLogArchive` directive, described under “AccessLogArchive - Remove existing access, agent, or referer log files or run a user exit” on page 324. You can have only one occurrence of this directive in your configuration file.

The format of the AccessLogExpire directive is:

```
AccessLogExpire number-of-days
```

*number-of-days*

Specifies that access logs older than this value are to be removed. *

*number-of-days* must be an integer; decimal values such as 1.5 are not valid. The default is 0, a value that indicates that no expiration date exists.

The file creation date, as reported by the operating system, is used to determine the age of the access log file. The suffix of the filename, such as `httpd-log.Mar221996.extension`, is not used to determine file age. (`extension` is the file extension.)

**Example**

```
AccessLogExpire 10
```

**Initial configuration file setting**

```
AccessLogExpire 0
```

**Program default setting**

None.
Logging and Reporting directives

**AccessLogSizeLimit - Remove existing access log files when they reach a given collective size**

Use this directive to specify that you want to remove access log files when they reach a collective size (in megabytes).

This directive requires that you also specify the AccessLogArchive directive, described under [AccessLogArchive - Remove existing access, agent, or referer log files or run a user exit” on page 324]. You can have only one occurrence of this directive in your configuration file.

The format of the AccessLogSizeLimit directive is:

```
AccessLogSizeLimit number-of-megabytes
```

*number-of-megabytes* specifies that when the combined size of the access log files exceeds this value, files are deleted starting with the oldest file, until the collective size is within the limit specified on the AccessLogSizeLimit directive. *number-of-megabytes* must be an integer. The default is 0, a value that indicates that no access log files are to be removed.

This directive takes effect after the AccessLogExpire directive has taken effect.

**Example**

```
AccessLogSizeLimit 4
```

**Initial configuration file setting**

```
AccessLogSizeLimit 0
```

**Program default setting**

None.

**AccessReportDescription - Give a short description of the HTLOGREP report to be created**

Use this directive to include a short description of the report to be created with this template.

**Note:** Use this directive only if you are using the HTLOGREP program to produce your reports.

**Example**

```
AccessReportDescription Report on Web page accesses
```

**Initial configuration file setting**

None.

**Program default setting**

None.

**AccessReportDoDnsLookup - Display client hostnames in HTLOGREP access reports**

Use this directive to display client hostnames in access reports created by HTLOGREP.

**Note:** Use this directive only if you are using the HTLOGREP program to produce your reports.
When access reports are created, a list of the client hostnames is generated and DNS is called. This results in improved performance since the server does not need to do a DNS lookup for each client access.

**Example**

```
AccessReportDoDnsLookup On
```

**Initial configuration file setting**

```
AccessReportDoDnsLookup Off
```

**Program default setting**

None.

**AccessReportExcludeURL** - Suppress log entries for specific files or directories from the HTLOGREP report

Use this directive to specify that you do not want the HTLOGREP program to include specific files or directories that match a given URL template. For example, you might not want to include requests for GIF files or access requests to a particular file or directory on your server.

**Note:** Use this directive only if you are using the HTLOGREP program to produce your reports.

**Examples**

```
AccessReportExcludeURL *.gif
AccessReportExcludeURL oldfiles*
```

You can only specify one file or directory on a directive. However, you can have multiple occurrences of this directive in your configuration file.

The following rules are used when attempting to match the template against a URL in the report being generated:

- If the template has no slash, any directories in the URL are ignored, and only the filename part of the URL is used for matching.
- When matching filenames, any number of asterisks can be used as wildcards, which can span dots (.), but not slashes (/).
- If the template contains any slash, it is considered to be path-specific. Therefore, the number of directories in the URL must match the number of directories in the template, and each directory name must match. When matching directory names, any number of asterisks can be used as wildcards, which can span dots (.)

**Initial configuration file setting**

None.

**Program default setting**

None.

**AccessReportIncludeURL** - Include only log entries for specific files or directories in the HTLOGREP report

Use this directive to specify that you want the HTLOGREP program to include only access requests made for specific files or directories that match a given URL template. For example, you might want to include only access requests for HTML files or you might want to include access requests to a particular file or directory on your server.
Logging and Reporting directives

Note: Use this directive only if you are using the HTLOGREP program to produce your reports.

Example

AccessReportIncludeURL /*.html

You can only specify one file name or directory on a directive. However, you can have multiple occurrences of this directive in your configuration file.

The following rules are used when attempting to match the template against a URL in the report being generated:

- If the template has no slash, any directories in the URL are ignored, and only the filename part of the URL is used for matching.
- When matching filenames, any number of asterisks can be used as wildcards, which can span dots (.), but no slashes (/).
- If the template contains any slash, it is considered to be path-specific. Therefore, the number of directories in the URL must match the number of directories in the template, and each directory name must match. When matching directory names, any number of asterisks can be used as wildcards, which can span dots ().

Initial configuration file setting
None.

Program default setting
None.

AccessReportExcludeHostName - Suppress the log entries for specific host names from the HTLOGREP report

Use this directive to specify that the HTLOGREP program should not include in the access report requests made by host names or IP addresses that match a given template.

Note: Use this directive only if you are using the HTLOGREP program to produce your reports.

You can only specify one host name or IP address on a directive. However, you can have multiple occurrences of this directive in your configuration file.

Note: To exclude host names, you must set the DNS-Lookup directive to On. If the DNS-Lookup directive is set to Off (the default), you can exclude IP addresses only.

Example

AccessReportExcludeHostName 9.85.*.*
AccessReportExcludeHostName *.edu

Initial configuration file setting
None.

Program default setting
None.
AccessReportIncludeHostName - Include only log entries for specific host names in the HTLOGREP report

Use this directive to specify that the HTLOGREP program should include in the access report requests made by host names or IP addresses that match a given template.

Note: Use this directive only if you are using the HTLOGREP program to produce your reports.

You can only specify one host name or IP address on a directive. However, you can have multiple occurrences of this directive in your configuration file.

Note: To include host names, you must set the DNS-Lookup directive to On. If the DNS-Lookup directive is set to Off (the default), you can include IP addresses only.

Example

AccessReportIncludeHostName 9.9.99.*
AccessReportIncludeHostName *.com

Initial configuration file setting
None.

Program default setting
None.

AccessReportExcludeMethod - Suppress the log entries of a given method type from the HTLOGREP report

Use this directive to specify that the HTLOGREP program should not include in the access report requests of a given method type.

Note: Use this directive only if you are using the HTLOGREP program to produce your reports.

You can specify only one method type on a directive. However, you can have multiple occurrences of this directive in your configuration file.

Example

AccessReportExcludeMethod GET
AccessReportExcludeMethod PUT
AccessReportExcludeMethod POST

Initial configuration file setting
None.

Program default setting
None.

AccessReportExcludeReturnCode - Suppress the log entries with a given return code from the HTLOGREP report

Use this directive to specify that the HTLOGREP program should not include in the access report requests that fall within a given set of error code numbers.
Logging and Reporting directives

Notes:

1. The error code numbers are http status codes. You cannot specify individual
   return codes. Specifying 300 indicates that you want to exclude from the report
   access requests with redirection return codes (301, 302, 303, and 304).

2. You can only specify one set of return codes on a directive. However, you can
   have multiple occurrences of this directive in your configuration file.

3. Use this directive only if you are using the HTLOGREP program to produce
   your reports.

Example

AccessReportExcludeReturnCode 200
AccessReportExcludeReturnCode 400

Initial configuration file setting
None.

Program default setting
None.

AccessReportRoot - Name the path for the root directory
where HTLOGREP access log reports are stored

Use this directive to specify the path and file name where you want the server to
store HTLOGREP access log reports and summary databases.

Note: Use this directive only if you are using the HTLOGREP program to produce
your reports.

We recommend that you accept the default path. If you choose to specify a
different path, you will need to create the new directory with all the appropriate
permissions and add a PASS directive to enable the server to honor requests for
reports in that directory.

If you are running with workload management enabled, you should have unique
AccessReportRoot directives for each instance of the server based on the subsystem
name. If you specify -SN system1, you should have AccessReportRoot
/usr/lpp/internet/server_root/pub/reports/system1 and a corresponding Pass
directive.

Example

AccessReportRoot WWW/reports

Initial configuration file setting
/usr/lpp/internet/server_root/pub/reports

Program default setting
None.

AccessReportTemplate - Name the HTLOGREP report template

Use this directive to specify the name of the HTLOGREP report template. The
default template is named “Top50”.

Note: Use this directive only if you are using the HTLOGREP program to produce
your reports.

The format of the AccessReportTemplate is:
Logging and Reporting directives

AccessReportTemplate *report_title*

*report_title*
  The name of the report. The name cannot include any blanks.

Example
  AccessReportTemplate Page_Hits

Initial configuration file setting
  AccessReportTemplate Top50 {
  AccessReportDescription Top 50 files and visitors
  AccessReportTopList 50
  }

Program default setting
  None.

AccessReportTopList - Specify the top number of items on which HTLOGREP is to report

Use this directive to specify the top number of items on which the HTLOGREP program is to report.

Note: Use this directive only if you are using the HTLOGREP program to produce your reports.

The format of the AccessReportTopList is:
  AccessReportTopList *top_number*|*all*

*top_number*  Specifies that the report is to include the *top_number* most frequently occurring entries in the access log. This must be an integer value.

Note: When you are using Web usage mining, the value for the AccessReportTopList directive must be an integer value up to 1000.

*all*  Specifies that the report is to include all entries in the report.

Example
  AccessReportTopList 10

Initial configuration file setting
  None.

Program default setting
  None.

AgentLog - Name the path for the agent log file

Use this directive to specify the place where the server logs statistics about which Web browser was used to access a Web page. By default the server writes an entry to this log each time a client sends the server a request. For every entry made in the access log, the agent log has a corresponding entry that indicates the browser used to display the page or file requested by the client.

The server starts a new agent log file each day at midnight if it is running. Otherwise, the server starts a new log file the first time you start it on a given day. When creating the file, the server uses the file name you specify and appends a date suffix. The date suffix is in the format *Mmmddyyyy*, where *Mmm* is the first three letters of the month; *dd* is the day of the month; and *yyyy* is the year.
Logging and Reporting directives

Example

AgentLog  logs/agent-log

Initial configuration file setting

AgentLog /usr/lpp/internet/server_root/logs/agent-log

This gives you ServerRoot/logs/agent-log.datesuffix.file_extension. In other words, if ServerRoot is /usr/lpp/internet/server_root/ and the server is started on January 15, 1996 the result is:


where extension is the web server generated file extension.

Program default setting

None.

CacheAccessLog - Specify the path for the cache access log files

If the server is running as a proxy, you can log requests to the cache separately from other requests. Use the CacheAccessLog directive to specify the path and file name where you want the server to put access requests for cached files. To enable logging of requests to the proxy cache, the following directives must be defined:

• Caching must be turned ON (default is OFF)
• CacheRoot (by default, no CacheRoot is defined)
• CacheAccessLog

The value of CacheAccessLog can either be an absolute path or a path relative to ServerRoot (one example is shown of each).

Note: If you choose to use CacheAccessLog, access requests for cached files are logged, but they are not included in the access reports. Access reports contain only information from access logs, not from cache access logs. Therefore, if you want access reports to contain access requests for cached files, do not specify the CacheAccessLog directive.

The server starts a new log file each day at midnight if it is running. Otherwise, the server starts a new log file the first time you start it on a given day. When creating the file, the server uses the file name you specify and appends a date suffix. The date suffix is in the format Mmmddyyyy, where Mmm is the first three letters of the month; dd is the day of the month; and yyyy is the year.

The format of this directive is

CacheAccessLog <file_path>

Example

CacheAccessLog /absolute/path/logfile
CacheAccessLog logs/logfile

Initial configuration file setting

None. The server does not log cache access requests if you include this directive in your configuration file.

Program default setting

None.
CgiErrorLog - Name the path for the CGI error log file

Use this directive to specify the place where the server logs standard error output (stderr) from CGI programs.

The server starts a new CGI error file each day at midnight if it is running. Otherwise, the server starts a new log file the first time you start it on a given day. When creating the file, the server uses the file name you specify and appends a date suffix. The date suffix is in the format **Mmmddyyyy**, where **Mmm** is the first three letters of the month; **dd** is the day of the month; and **yyyy** is the year.

**Example**

```
CgiErrorLog logs/cgi-error
```

**Initial configuration file setting**

```
CgiErrorLog /usr/lpp/internet/server_root/logs/cgi-error
```

This gives you **ServerRoot/logs/cgi-error**. In other words, if **ServerRoot** is **/usr/lpp/internet/server_root/** and the server is started on January 15, 1996 the result is:

```
```

where **extension** is the web server generated file extension.

**Program default setting**

None.

DoReporting — Specify if reports are automatically generated

Use this directive to specify whether you want the Web server to automatically generate reports. When this directive is set on, the server will use the default reporting program, HTLOGREP, to automatically generate reports, unless you specify a third-party reporting program.

**Notes:**

1. HTLOGREP uses your default configuration file options unless you change the default options. To change default options, use the **LoggingReportingProgram** and **LoggingReportingProgramOptions** directives.
2. To use a third-party reporting program instead of HTLOGREP, use the **LoggingReportingProgram** and **LoggingReportingProgramOptions** directives.
3. For an overview of reporting options, see [“Tailoring the reports your server creates” on page 78](#).

**Example**

```
DoReporting Off
```

**Initial configuration file setting**

```
DoReporting On
```

ErrorLog - Name the file where you want to log internal server errors

Use this directive to specify the path and file name where you want the server to log internal errors.

The server starts a new log file each day at midnight if it is running. Otherwise, the server starts a new log file the first time you start it on a given day.
Creating the file, the server uses the file name you specify and appends a date suffix. The date suffix is in the format `Mmmddyyyy`, where `Mmm` is the first three letters of the month; `dd` is the day of the month; and `yyyy` is the year.

**Example**

```text
ErrorLog logs/errorlog
```

**Initial configuration file setting**

```text
ErrorLog /usr/lpp/internet/server_root/logs/httpd-error
```

**Program default setting**

None.

**ErrorLogArchive - Remove existing error or CGI error log files or run a user exit**

Values specified on the `ErrorLogArchive` directive apply to error and CGI error logs. The collective size includes the size of all error logs or all CGI logs, not the collective size of both types of logs.

At midnight each night, the server closes the current error and CGI error logs and creates new log files for the coming day. You can choose to do one of the following actions with the closed error logs:

- Remove log files of a given age or when a given amount of storage is used by the collection of error log files
- Allow closed logs to remain in your file system
- Branch to a user exit.

To remove logs of a given age, specify this directive, in addition to the `ErrorLogExpire` directive. To remove logs when their collective size exceeds a certain amount of storage, specify this directive, in addition to the `ErrorLogSizeLimit` directive.

To allow closed logs to remain on your file system, you can accept the default, which is `ErrorLogArchive none`.

To branch to a user exit, specify the path to the user exit and any parameters for the user exit on the `ErrorLogArchive` directive. The server will append to this directive the path to the error or CGI error log.

The `ErrorLogArchive` directive can be specified in any of the following formats:

- `ErrorLogArchive purge`
- `ErrorLogArchive none`
- `ErrorLogArchive userexit path_to_the_user-exit_program [parameters_for_the_user-exit]`

**Note:** The directive must be typed on one line, even though it is shown here on two lines.

- `purge`
  Remove log files of a given age or when their collective size exceeds a given amount of storage.

- `none`
  Do not remove log files. `none` is the default.

- `userexit`
  Specifies the path of the user-exit program you want to branch to. You can
optionally specify the parameters for your user-exit program, as shown in the following examples. The server appends the path to the error log to the directive.

**Examples:**
- `ErrorLogArchive purge`
- `ErrorLogArchive none`
- `ErrorLogArchive userexit /u/userxyz/bin/errback/backup.rexx -d -a`

For the `ErrorLogArchive userexit` example, the user exit invocation is:
- `ErrorLogArchive userexit /u/userxyz/bin/errback/backup.rexx
  -d -a /www/logs/httpd-error`

**Note:** The directive must be typed on one line, even though it is shown here on two lines.

**Initial configuration file setting**
- `ErrorLogArchive none`

**Program default setting**
- None.

**ErrorLogExpire - Remove existing error log files when they reach a given age in days**

Use this directive to specify that you want to remove error log files when they reach a certain age (in days).

This directive requires that you also specify the `ErrorLogArchive` directive, described under “`ErrorLogArchive - Remove existing error or CGI error log files or run a user exit`” on page 336. You can have only one occurrence of this directive in your configuration file.

The format of the `ErrorLogExpire` directive is:
- `ErrorLogExpire number-of-days`

*number-of-days*
  - Specifies that error logs older than this value are to be removed. *number-of-days* must be an integer; decimal values such as 1.5 are not valid. The default is 0, a value that indicates that no expiration date exists.

The file creation date, as reported by the operating system, is used to determine the age of the error log file. The suffix of the filename, such as `httpd-log.Mar221996`, is not used to determine file age.

**Example**
- `ErrorLogExpire 10`

**Initial configuration file setting**
- `ErrorLogExpire 0`

**Program default setting**
- None.
Logging and Reporting directives

**ErrorLogSizeLimit - Remove existing error log files when they reach a given collective size**

Use this directive to specify that you want to remove error log files when they reach a collective size (in megabytes).

This directive requires that you also specify the ErrorLogArchive directive, described under "ErrorLogArchive - Remove existing error or CGI error log files or run a user exit" on page 336. You can have only one occurrence of this directive in your configuration file.

The format of the ErrorLogSizeLimit directive is:

```
ErrorLogSizeLimit number-of-megabytes
```

Specifies that when the sum total size of the error log files exceeds this value, files are deleted starting with the oldest file, until the collective size is within the limit specified on the ErrorLogSizeLimit directive. `number-of-megabytes` must be an integer. The default is 0, a value that indicates that no error log files are to be removed.

This directive takes effect after the ErrorLogExpire directive has taken effect.

**Example**

```
ErrorLogSizeLimit 4
```

**Initial configuration file setting**

```
ErrorLogSizeLimit 0
```

**Program default setting**

None.

**LogFormat - Specify common or old log file format**

Use this directive to specify whether you want your server to write log files in the common format or old format.

If you plan to use the reporting functions described under "Tailoring the reports your server creates" on page 78, you must accept the default file format, common.

The common format is the one used by most Web servers. The old format was used with early versions of Web servers from CERN. You most likely will want to use the common format, which is the default.

**Example**

```
LogFormat Old
```

**Initial configuration file setting**

```
LogFormat Common
```

**Program default setting**

Common.
Logging and Reporting directives

LoggingReportingDebugOutput— Generate debug log for HTLOGREP reporting program

Use this directive to help you troubleshoot problems with the HTLOGREP reporting program. If you specify a debug log file name using this directive, an HTLOGREP debug log will be created at midnight. To use this option, the DoReporting directive must be set on.

Example

LoggingReportingDebugOutput /temp/mydebug

Initial configuration file setting

None

For examples of HTLOGREP debug logs, see the Logging and Reporting hints and tips in the Web-based WebSphere Troubleshooter for OS/390. To access the Troubleshooter, go to URL:


LoggingReportingProgram — Specify the reporting program to be used

Use this directive if you want to use a third-party reporting program or change the default options for the HTLOGREP reporting program:

- To use a third-party reporting program, such as Analog, instead of HTLOGREP:
  1. Enter the complete file path of the third-party reporting program.
  2. Specify reporting program options using the LoggingReportingProgramOptions directive.

- If you are using the default reporting program, HTLOGREP, but want to change the default reporting options:
  1. Enter the complete file path of the HTLOGREP program. The default path is /usr/lpp/internet/sbin/htlogrep.
  2. Specify HTLOGREP options using the LoggingReportingProgramOptions directive.

For an overview of reporting options, see “Tailoring the reports your server creates” on page 78.

Example if using third-party program

LoggingReportingProgram /usr/lpp/internet/sbin/analog

Example if using HTLOGREP and changing default options

LoggingReportingProgram /usr/lpp/internet/sbin/htlogrep

Initial configuration file setting

None

LoggingReportingProgramOptions — Specify reporting program options

Use this directive to:

- Specify options for a third-party reporting program you specified on the LoggingReportingProgram directive. See your program documentation for information on possible options.
Logging and Reporting directives

- Change the default reporting options for the HTLOGREP reporting program. If you specify options for HTLOGREP using this directive, you must also specify the complete file path of the HTLOGREP program on the `LoggingReportingProgram` directive.

  Possible options are:
  - Use a configuration file other than the default configuration file (`-c`).
  - Create a debug log (`-d`).
  - Specify a log file name other than the default (`-l`).

  For an overview of reporting options, see "Tailoring the reports your server creates" on page 78.

  For examples of HTLOGREP debug logs, see the Logging and Reporting hints and tips in the Web-based WebSphere Troubleshooter for OS/390. To access the Troubleshooter, go to URL: http://www.ibm.com/software/websphere/httpservers/troubleshooter.html

  **Example if using HTLOGREP and changing default options**
  
  `LoggingReportingProgramOptions -c /etc/myhttpd.conf -d /temp/mydebug -l /temp/mylog`

  **Initial configuration file setting**
  
  None

  **LogTime - Specify GMT or local time stamps in log files**
  
  Use this directive to specify whether your logs should record entries using local time or Greenwich Mean Time (GMT).

  **Example**
  
  `LogTime GMT`

  **Initial configuration file setting**
  
  LogTime LocalTime

  **Program default setting**
  
  LocalTime.

  **LogToSyslog - Log access information to the syslog, in addition to or instead of the error log**
  
  Use this directive to specify whether you want your server to log access requests and errors to the MVS syslog daemon in addition to the access and error log files. If you want to have access and error log information sent to the syslog file in addition to sending it to the log files, you must change the default.

  The MVS syslog daemon must be up and running on your system before you specify that error log information be written to it. You can choose whether to log only access or error information or to log both.

  For information about how to configure the syslog daemon, refer to IBM TCP/IP for OS/390 UNIX System Services MVS Application Feature Guide, SC31-8069.

  **Example**
  
  `LogToSyslog On`
Initial configuration file setting
LogToSyslog Off

Program default setting
LogToSyslog Off

NoLog - Suppress log entries for specific hosts or domains matching a template

Use this directive to specify that you do not want to log access requests made from specific hosts or domains that match a given template. For example, you may not want to log access requests from local hosts.

You can have multiple occurrences of this directive in your configuration file. You can also put multiple templates on the same directive if you separate them by one or more spaces. You can use host names or IP addresses on the templates.

Note: To use host name templates, you must set the DNS-Lookup directive to On. If the DNS-Lookup directive is set to Off (the default), you can use IP address templates only.

Example
NoLog 128.141.* *.edu localhost.*

Initial configuration file setting
None

Program default setting
None.

ProxyAccessLog - Name the path for the proxy access log file

Use this directive to specify the place where the server logs all requests which it handles through its proxy services.

The server starts a new log file each day at midnight if it is running. Otherwise, the server starts a new log file the first time you start it on a given day. When creating the file, the server uses the file name you specify and appends a date suffix. The date suffix is in the format Mmmddyyyy, where Mmm is the first three letters of the month; dd is the day of the month; and yyyy is the year.

It is a good idea to remove old log files, because they can take up a significant amount of space on your file system.

Example
ProxyAccessLog logs/proxylog

Initial configuration file setting
ProxyAccessLog /usr/lpp/internet/server_root/logs/proxy-log

This gives you ServerRoot/logs/proxy-log date suffix file extension. In other words, if ServerRoot is /usr/lpp/internet/server_root/ and the server is started on January 15, 1996 the result is:

where extension is the web server generated file extension.
RefererLog - Name the path for the referer log file

Use this directive to specify the place where you want the server log the identity of the Web page that referred to (linked to) the requested Web page. By default the server writes an entry to this log each time a client sends the server a request. For every entry made in the access log, the referer log has a corresponding entry that indicates which page referred to the page that was requested by the client. If no page referred to the requested page, the entry is two quotation marks (" ").

The server starts a new referer log file each day at midnight if it is running. Otherwise, the server starts a new log file the first time you start it on a given day. When creating the file, the server uses the file name you specify and appends a date suffix. The date suffix is in the format Mmmddyyyy, where Mmm is the first three letters of the month; dd is the day of the month; and yyyy is the year.

Example
RefererLog logs/referer-log

Initial configuration file setting
RefererLog /usr/lpp/internet/server_root/logs/referer-log

This gives you ServerRoot/logs/referer-log,datesuffix.file_extension. In other words, if ServerRoot is /usr/lpp/internet/server_root/ and the server is started on January 15, 1996 the result is:

where extension is the web server generated file extension.

Program default setting
None.

ReportDataCompressionProgram - Specify path to the compression program

Use this directive to specify the path to the compression program (such as PKZIP2, GZIP, or compress) and any program parameters for the compression program. Include any command line parameters on the same line. This compression program is to be used to compress access data log files.

Example
ReportDataCompressionProgram /bin/compress

Initial configuration file setting
None

Program default setting
None

ReportDataUnCompressionProgram - Specify path to the uncompression program

Use this directive to specify the path to the uncompression program (such as UNZIP, GZIP, or uncompress) and any program parameters for the uncompression program. Include any command line parameters on the same line. This uncompression program is to be used to uncompress access data log files.
Logging and Reporting directives

Example

ReportDataUnCompressionProgram /bin/uncompress

Initial configuration file setting
None

Program default setting
None

ReportDataCompressionSuffix - Specify the suffix appended to the compressed report data files

Use this directive to specify the suffix appended to the compressed report data files.

Example

ReportDataCompressionSuffix .Z

Initial configuration file setting
None

Program default setting
None

ReportProcessOldLogs - Check for old logs in the log directory

Use this directive to indicate that you want the server to check for old access logs in the log directory that are not listed in the list of log files that have been processed into reports. With this directive, you can process old access log files by:

• Appending the data from the old access log files to existing reports
• Creating reports for all access log files and overwriting existing reports
• Creating a report for the most recently created access log file.

The format of the ReportProcessOldLogs directive is:

ReportProcessOldLogs append|force|last

append
Add to existing access log reports data from log files that were not originally included in the reports.

force
Overwrite existing access log reports with reports based on data from all access log files, regardless of whether they were originally included in the reports.

Note: The only way to erase reports named access.mmdy yyyy files is to archive them with the ReportDataArchive directive, described under “ReportDataArchive - Specify whether to remove existing accessdata files” on page 344.

last
Create reports based on data from the most recently created access log file.

Examples

ReportProcessOldLogs append
ReportProcessOldLogs force
ReportProcessOldLogs last
Logging and Reporting directives

Initial configuration file setting
None.

Program default setting
None.

ReportDataSizeLimit - Remove existing access data files when they reach a given collective size

Use this directive to specify that you want to remove access data files when they reach a collective size (in megabytes).

This directive requires that you also specify the ReportDataArchive directive, described under “AccessLogArchive - Remove existing access, agent, or referer log files or run a user exit” on page 324. You can have only one occurrence of this directive in your configuration file.

The format of the ReportDataSizeLimit directive is:

ReportDataSizeLimit number-of-megabytes

number-of-megabytes
Specifies that when the sum total size of the access data files exceeds this value, files are deleted starting with the oldest file, until the collective size is within the limit specified on the ReportDataSizeLimit directive.

number-of-megabytes must be an integer. The default is 0, a value that indicates that no access data files are to be removed.

This directive takes effect after the ReportDataExpire directive has taken effect.

Example

ReportDataSizeLimit 4

Initial configuration file setting
None.

Program default setting
None.

ReportDataArchive - Specify whether to remove existing access data files

Use this directive to specify whether you want to remove existing access data log files.

If you want to remove access data files, you also need to specify the ReportDataExpire directive, described under “ReportDataExpire - Remove existing access data files when they reach a given age in days” on page 345. You can have only one occurrence of this directive in your configuration file.

Even after you remove access data files, the data from these files is still available for reports to use, until you specify the ReportProcessOldLogs directive with the force option.

The ReportDataArchive directive can be specified in any of the following formats:

ReportDataArchive purge
ReportDataArchive none
ReportDataArchive userexit path_to_the_user-exit_program [parameters_for_the_user-exit]
Logging and Reporting directives

**Note:** The directive must be typed on one line, even though it is shown here on two lines.

**purge**
Remove access data files of a given age or when their collective size exceeds a given amount of storage.

**none**
Do not remove access data files. none is the default.

**userexit**
Specifies the path of the user-exit program you want to branch to. You can optionally specify the parameters for your user-exit program, as shown in the following examples. The server appends the path to the access log, to the directive.

**Examples:**

```
ReportDataArchive purge
ReportDataArchive none
ReportDataArchive userexit /u/userxyz/bin/backup/backup.rexx -d -a
```

For the `ReportDataArchive userexit` example, the user exit invocation is:

```
ReportDataArchive userexit /u/userxyz/bin/backup/backup.rexx
   -d -a /www/logs/httpd-log
```

**Note:** The directive must be typed on one line, even though it is shown here on two lines.

**Initial configuration file setting**
None.

**Program default setting**
None.

**ReportDataExpire - Remove existing access data files when they reach a given age in days**

Use this directive to specify that you want to remove access log data files when they reach a certain age (in days).

This directive requires that you also specify the `ReportDataArchive` directive, described under [“ReportDataArchive - Specify whether to remove existing accessdata files” on page 344](#). You can have only one occurrence of this directive in your configuration file.

```
ReportDataExpire number-of-days
```

**number-of-days**
Specifies that reports older than this value are to be removed. *number-of-days* must be an integer; decimal values such as 1.5 are not valid. The default is 0, a value that indicates that no expiration date exists.

The file creation date, as reported by the operating system, is used to determine the age of the error log file. The suffix of the filename, such as `httpd-log.Mar221996`, is not used to determine file age.

**Example**

```
ReportDataExpire 10
```
Initial configuration file setting
None.

Program default setting
None.

SMF - Specify the type of information that SMF records
Use this directive to write configuration and performance data to a memory segment in preparation for analysis. The format of the directive is:

```
SMF settings
```

where `settings` can be `all`, `config`, `perf`, or `none`.

- `all` Record both configuration and performance data to SMF.
- `config` Record only configuration data (record type 103, subtype 01) to SMF.
- `perf` Record only performance data (record type 103, subtype 02) to SMF.
- `none` Record no configuration or performance data to SMF.

Example
```
SMF perf
```

Initial configuration file setting
None.

Program default setting
None.

SMFRecordingInterval - Specify how often performance record information is recorded
Use this directive to define how often SMF writes performance record information to file. Performance record data accumulates continuously. The logging thread writes SMF records only when the logging queue is full. If the level of activity on the server is low, the logging queue fills more slowly and the SMF records are written less frequently than the interval specified in this directive. The format of the directive is:

```
SMFRecordingInterval hh:mm
```

- `hh:mm` Indicates the time that elapses between recordings of performance information.

Example
```
SMFRecordingInterval 00:20
```

The recording interval for performance information is 20 minutes.

Initial configuration file setting
```
SMFRecordingInterval 00:15
```

Program default setting
```
SMFRecordingInterval 00:15
```
**Meta-Information** - Name meta-information files and directories

Use the directives described in this section to control where your server looks for meta-information files.

You can use a separate set of files to store meta-information about your server’s documents. The server can include the meta-information with its HTTP responses. Meta-information describes the file containing a document, not the contents of the document. For example, meta-information for a file might give the date the file was created and the date it was last modified. You can include any valid response headers as described in the HTTP 1.1 specification.

HTTP recognizes MIME headers. Information that MIME header fields can include are the file type, subtype, encoding, and content length.

Each line of a meta-information file contains a header field, followed by a colon, and the value of the field. For example:

- Last-Modified: Wednesday, 05-Apr-96 20:51:35 GMT
- Expires: Friday, 30-Jun-96 24:00:00 GMT
- MIME-Version: 1.0

**MetaDir** - Specify name of subdirectory for meta-information files

Use this directive to specify the name you want to use for subdirectories that contain meta-information files. You can only have one instance of this directive, which means all your meta-information subdirectories have the same name. The stated directory is a subdirectory of the directory in the document which "meta" is located in.

Any directory from which your server retrieves files can have a subdirectory with the name specified on this directive. The files on the meta-information subdirectory contain meta-information about the files being retrieved. The meta-information files have the same file name and extension as the file they describe, plus an added extension. The name of the added suffix is specified on the MetaSuffix directive.

For example, you might have the following two directives in your configuration file:

```
MetaDir look_here
MetaSuffix .file_desc
```

If your server goes to retrieve this file:
```
/html/realcool/coolindex.html
```

it looks for meta information to include with the response in this file:
```
/html/realcool/look_here/coolindex.html.file_desc
```

**Example**

```
MetaDir mimeinfo
```

**Initial configuration file setting**

```
MetaDir .web
```

**Note:** The dot character (.) at the beginning of the default value is used as part of the subdirectory name.
Meta-Information directives

MetaSuffix - Specify the suffix for meta-information files

Use this directive to specify the suffix you want to use for meta-information files. You can only have one instance of this directive, which means all meta-information files end with the same suffix. You must include the period character (.) as part of the value.

Any file your server retrieves can have a meta-information file associated with it. A meta-information file has the same file name and suffix as the file it describes, plus the additional suffix specified on the MetaSuffix directive. The suffix is an extension of the complete "meta" file name. A meta-information file must be located on a subdirectory of the directory that contains the file being described. The name of the subdirectory must be the name specified on the MetaDir directive.

See the description of the MetaDir directive to see an example of how MetaDir and MetaSuffix work together.

Example

| MetaSuffix | .head |

Initial configuration file setting

| MetaSuffix | .meta |

Methods - Set method acceptance

Use the directives described in this section to control which HTTP methods are enabled for your server.

Client requests to the server include a method field that indicates the action the server is to perform on the specified object. The request identifies the object with a Uniform Resource Locator (URL).

Following is a list of methods that the server supports and a description of how the server would respond to a client request containing the method. The description assumes the method is enabled.

- **CONNECT** - This method is used to establish an SSL tunneling session between a client, such as NetScape Navigator, and a remote server through a proxy server. The sessions between the client and the proxy and between the proxy and the remote server are secure. The proxy does not have access to the data being sent. The proxy server can be a base or secure server.

- **DELETE** - The server deletes the object identified by the URL. After the object is deleted, the URL is not valid. Because delete typically lets clients delete information from your server, you must use protection setups to define who can use this method and which files can be deleted.

- **GET** - The server returns whatever data is identified by the URL. If the URL refers to an executable program, the server returns the output of the program.

- **HEAD** - The server returns only HTTP document headers without the document body.

- **OPTIONS** - The request returns information about the communications options on the request/response chain identified by the URL. This method allows a client to determine the options and requirements associated with an object, or the capabilities of a server, without having to act on or retrieve the object.

- **POST** - The request contains data and a URL. The server accepts the data enclosed in the request as a new subordinate of the resource identified in the URL. The resource, which may be a data-accepting program, a gateway to some
other protocol, or a separate program that accepts annotations, processes the enclosed data. The POST method is designed to handle annotation of existing resources; posting of a message to a bulleted boards, newsgroup, mailing list, or similar group of articles; providing a block of data, such as data from a form to a data-handling program; or extending a database through an append operation. In the HTTP Server, the POST method is used to process the Configuration and Administration forms.

- **PUT** - The request contains data and a URL. The server stores the resource identified in the URL. If the resource already exists, PUT replaces it. If the resource does not exist, PUT creates it. Because PUT typically lets clients add or replace information on your server, you must use protection setups to define who can use this method for which files.

- **TRACE** - The server echos the request message sent by the client. This method allows the client to see what is being received at the other end of the request chain and use that data for testing or diagnostic information. The content type of the response is message/http.

### Disable - Disable HTTP methods

Use this directive to specify which HTTP methods you do not want your server to accept.

In the default configuration file, the GET, HEAD, OPTIONS, POST, and TRACE methods are enabled and all other supported HTTP methods are disabled. To disable a method that is currently enabled, change the Enable directive for the method to a Disable directive.

**Note:** The Configuration and Administration forms use the POST method to make updates to your server configuration. If you disable the POST method you will not be able to use the Configuration and Administration forms.

**Example**

Disable HEAD

**Initial configuration file setting**

Disable PUT
Disable DELETE

### Enable - Enable HTTP methods

Use this directive to specify which HTTP methods you want your server to accept.

You can enable as many of the HTTP methods as you need. For each method you want the server to accept, enter a separate Enable directive followed by the name of the method.

**Example**

Enable DELETE

If no Service directive exists for a particular URL, you can use the Enable directive to perform customized programming for any HTTP method. The program you specify on this directive will override the standard processing for that method.

The format is:

```
Enable method /path/fileDLL:function_name
```
Methods directives

Example
Enable

Initial configuration file setting
Enable GET
Enable HEAD
Enable POST
Enable TRACE
Enable OPTIONS

Multi-format processing - Define file extensions for multi-format processing

Use the directives described in this section to associate files with particular extensions to the meta-information found in the headers of incoming requests. Based on the file suffix (suffixes) specified in these directives, the server binds files to a content type, content encoding, content language, character set, or to a browser sending a request.

Multi-Format Processing

Multi-format processing is only enabled when the requesting URL contains the .multi suffix or does not have a suffix (and a file with that name and no suffix does not exist).

csl1*

The resource mapping directives, AddType, AddEncoding, AddLanguage, AddCharSet, and AddClient, are used to associate meta-information from request headers with file suffixes or extensions. Meta-information can consist of MIME type, encoding, quality, charset, language, and browser (agent) type. When processing a file for which a suffix or extension is not defined by an AddEncoding or AddType directive, the server uses the default value of binary for the content encoding.

The following list identifies meta-information that is associated with each directive:

- AddType directive
  - Suffix
  - MIME
  - Encoding
  - Quality (optional)
  - Charset (optional)
- AddEncoding directive
  - Suffix
  - Encoding
  - Quality (optional)
- AddLanguage directive
  - Suffix
  - Quality (optional)
  - Language
- AddCharSet directive
  - Suffix
Multi-format processing directives

- Quality (optional)
- Charset
- AddClient directive
  - Suffix
  - Quality (2.0 implied)
  - Agent

There are two pieces to multi-format processing:

- Request headers from the browser
  The browser sends accept-headers containing acceptable values (content-type, content-encoding, language, charset) that you can associate to file suffixes with the configuration directives. The browser also sends a user-agent header that identifies its browser type that you can associate with file suffixes in the same manner.

- The requested URL
  The server finds all files with any extension that matches the directory and file name and uses multi-format processing to choose the best file to return.

Computing file quality

Multi-format processing computes the quality of a file based on the set of suffixes in the directory for the requested file name and sends the highest quality it finds. A perfect match is always the highest quality.

Quality is a floating point number between 0.0 and 1.0 that represents the relative desirability of a file. For example, if the file abc.html is requested and cannot be found, the server searches the directory for all files that match abc.*. Multi-format processing adds the * and finds all that may qualify.

When computing file quality, each file is given the value of 1.0 and this is multiplied by the quality of each suffix. The quality of a suffix is based on the quality value specified in the directive which is multiplied by the quality value specified in the accept-header. This can be shown as:

\[
\text{fileQ} = (1.0 \times ((\text{suffixQ1}) \times (\text{suffixQ2})...))
\]

or

\[
\text{fileQ} = (1.0 \times ((\text{directiveQ} \times \text{headerQ}) \times (\text{directiveQ} \times \text{headerQ})...))
\]

The quality value is optional on many of the directives. If one is not specified, then the value of 1.0 is assumed. Also, if the accept-header does not specify a quality value, 1.0 is assumed.

If the server is processing a file and suffix that is defined in the list of AddClient directives, and the requester agent matches the defined string, the server doubles the file quality (implying a value of 2.0). If no match is found, this does not happen.

Example: If you have HTML source files in different languages,

```
myfile.du.html
myfile.uk.html
```

you could use these directives for multi-format processing:

```
AddLanguage .du du
AddLanguage .uk en_UK
AddType .html text/html ebcDIC
```
Multi-format processing directives

Then, if a browser sends a request for myfile.multi and sends the header Accept-Language: du, the server returns myfile.du.html. (Both files are stored in EBCDIC charset IBM-1047 and translated to ASCII charset ISO8859-1 when served.) The configuration file shipped with the server contains specifications for most commonly used suffixes. Use the suffix definition directives only if you need to add new definitions or change the sample definitions found in the configuration file.

AddLanguage - Specify the language of files with particular suffixes

Use this directive to bind files with a particular suffix to a language. The format of the directive is:

```
AddLanguage .extension language
```

---

**Examples**

```
AddLanguage .en en_US
```

This example defines files with a .en suffix as being in American English.

```
AddLanguage .uk en_UK
```

This example defines files with a .uk suffix as being in United Kingdom English.

**Initial configuration file setting**

None.

AddEncoding - Specify the MIME content encoding of files with particular suffixes

Use this directive to bind files with a particular suffix to a MIME-encoding type.

The format of the directive is:

```
AddEncoding .extension encoding
```

---

**Example**

```
AddEncoding .qp quoted_printable
```

**Initial configuration file setting**

```
AddEncoding .Z x-compress
AddEncoding .gz x-gzip
AddEncoding .ascii 8bit
AddEncoding .asc8 8bit
AddEncoding .asc7 7bit
AddEncoding .ebcdic ebc dic
```
AddCharSet - Specify the character set documents are encoded in

Use this directive to specify the character set (code page) documents are stored in.
(You can also specify character set on the AddType directive.)

The format of the directive is:

\texttt{AddCharSet .extension \ character-set}

\texttt{.extension}

The file suffix pattern.

\texttt{character-set}

The character set you want to associate with text documents. This should indicate the character set transmitted. If encoding is specified as ebcdic, the character set is ISO8859-1. For the documents that you assign a character set to, the server tells client browsers what character set to use when displaying the document. If you want to use the character-set field, you must also include a value for the quality field. If you use DefaultFsCp or DefaultNetCp, you might want to use \texttt{character-set} to indicate displayed or stored binary pages.

Example

\texttt{AddCharSet .932 IBM-932}

Initial configuration file setting

None

AddType - Specify the data type of files with particular suffixes

Use this directive to bind files with a particular suffix to a MIME type/subtype. You can have multiple occurrences of this directive in your configuration file. The format of the directive is:

\texttt{AddType .extension type/subtype encoding [quality[ character-set]]}

\texttt{.extension}

The file suffix pattern. You can use the wildcard character (*) only on the following two special suffix patterns:

\texttt{**} Matches all file names that contain a dot character (.) and have not been matched by other rules

\texttt{*} Matches all file names that do not contain a dot character (.) and have not been matched by other rules

\texttt{type/subtype}

The MIME type and subtype you want to bind to files that match the corresponding suffix pattern.

\texttt{encoding}

The MIME content encoding to which the data has been converted. On MVS, text data is transmitted in 7 bit or 8 bit ASCII and stored in the HFS as EBCDIC. The iconv() service is used to translate between ASCII codepage ISO8859-1 and EBCDIC codepage IBM-1047. Only files with ebcdic encoding are translated. In most cases, the appropriate encoding is 7bit, 8bit, binary, or ebcdic and is determined as follows:

\texttt{7bit} All data is represented as short lines (less than 1000 characters) of 8859-1 ASCII data (less than 1000 characters). Source code or plain text
files usually fall into this category. Exceptions would be files containing line-drawing characters or accented characters.

8bit  Data is represented as short lines, but may contain characters with the high bit set (for example, line-drawing characters or accented characters). PostScript files and text files from European sites usually fall into this category.

binary  This encoding can be used for all data types. Data may contain not only non-ASCII characters, but also long (greater than 1000 characters) lines. Almost every file of type image/*, audio/*, and video/* falls into this category, as do binary data files of type application/*.

ebcdic  Data is represented as lines of EBCDIC text (IBM-1047) and data is translated to ASCII text (ISO8859-1) when served. Note that codepages can be overridden by the DefaultFsCp and DefaultNetCp directives.

other  Any other value of encoding receives the same treatment as binary and is passed in MIME headers as a content encoding MIME header. Encoding values of 7bit, 8bit, and ebcdic are not sent in MIME headers.

quality  An optional indicator of relative value (on a scale of 0.0 to 1.0) for the content type. The quality value is used if multiple representations of a file are matched by a request. The server selects the file that is associated with the highest quality value. For example, if the file internet.ps is requested, and the server has the following AddType directives:

```
AddType .ps application/postscript 8bit 1.0
AddType *.ps application/binary binary 0.3
```

the server would use the application/postscript line because its quality number is higher.

class-set  An optional indicator of the character set you want to associate with text documents. This should indicate the character set transmitted. If encoding is specified as ebcdic, the character set is ISO8859-1. For the documents that you assign a character set to, the server tells client browsers what character set to use when displaying the document. If you want to use the character-set field, you must also include a value for the quality field. If you use DefaultFsCp or DefaultNetCp, you might want to use character-set to indicate displayed or stored binary pages.

You can also use the AddCharSet directive to specify character set.

**Example**

```
AddType .html text/html ebcdic 1.0
```

**Initial configuration file settings**

<table>
<thead>
<tr>
<th>AddType</th>
<th>Type</th>
<th>encoding</th>
<th>quality</th>
</tr>
</thead>
<tbody>
<tr>
<td>.cer</td>
<td>application/x-x509-user-cert</td>
<td>ebcdic</td>
<td>0.5</td>
</tr>
<tr>
<td>.der</td>
<td>application/x-x509-ca-cert</td>
<td>binary</td>
<td>1.0</td>
</tr>
<tr>
<td>.mime</td>
<td>www/mime</td>
<td>binary</td>
<td>1.0</td>
</tr>
<tr>
<td>.bin</td>
<td>application/octet-stream</td>
<td>binary</td>
<td>1.0</td>
</tr>
<tr>
<td>.class</td>
<td>application/octet-stream</td>
<td>binary</td>
<td>1.0</td>
</tr>
<tr>
<td>.oda</td>
<td>application/oda</td>
<td>binary</td>
<td>1.0</td>
</tr>
<tr>
<td>.pdf</td>
<td>application/pdf</td>
<td>binary</td>
<td>1.0</td>
</tr>
<tr>
<td>.ai</td>
<td>application/postscript</td>
<td>ebcdic</td>
<td>0.5</td>
</tr>
<tr>
<td>.PS</td>
<td>application/postscript</td>
<td>ebcdic</td>
<td>0.8</td>
</tr>
<tr>
<td>.eps</td>
<td>application/postscript</td>
<td>ebcdic</td>
<td>0.8</td>
</tr>
<tr>
<td>.ps</td>
<td>application/postscript</td>
<td>ebcdic</td>
<td>0.8</td>
</tr>
<tr>
<td>.rtf</td>
<td>application/x-rtf</td>
<td>ebcdic</td>
<td>1.0</td>
</tr>
<tr>
<td>File Extension</td>
<td>MIME Type</td>
<td>Encoding</td>
<td>Comments</td>
</tr>
<tr>
<td>----------------</td>
<td>-----------</td>
<td>----------</td>
<td>----------</td>
</tr>
<tr>
<td>.csh</td>
<td>application/x-csh</td>
<td>ebcdic</td>
<td>0.5</td>
</tr>
<tr>
<td>.dvi</td>
<td>application/x-dvi</td>
<td>binary</td>
<td>1.0</td>
</tr>
<tr>
<td>.hdf</td>
<td>application/x-hdf</td>
<td>binary</td>
<td>1.0</td>
</tr>
<tr>
<td>.latex</td>
<td>application/x-latex</td>
<td>ebcdic</td>
<td>1.0</td>
</tr>
<tr>
<td>.nc</td>
<td>application/x-netcdf</td>
<td>binary</td>
<td>1.0</td>
</tr>
<tr>
<td>.cdf</td>
<td>application/x-cdf</td>
<td>ebcdic</td>
<td>1.0</td>
</tr>
<tr>
<td>.sh</td>
<td>application/x-sh</td>
<td>ebcdic</td>
<td>0.5</td>
</tr>
<tr>
<td>.tcl</td>
<td>application/x-tcl</td>
<td>ebcdic</td>
<td>0.5</td>
</tr>
<tr>
<td>.tex</td>
<td>application/x-tex</td>
<td>ebcdic</td>
<td>1.0</td>
</tr>
<tr>
<td>.texinfo</td>
<td>application/x-texinfo</td>
<td>ebcdic</td>
<td>1.0</td>
</tr>
<tr>
<td>.t</td>
<td>application/x-troff</td>
<td>ebcdic</td>
<td>0.5</td>
</tr>
<tr>
<td>.roff</td>
<td>application/x-troff</td>
<td>ebcdic</td>
<td>0.5</td>
</tr>
<tr>
<td>.man</td>
<td>application/x-troff-man</td>
<td>ebcdic</td>
<td>0.5</td>
</tr>
<tr>
<td>.me</td>
<td>application/x-troff-me</td>
<td>ebcdic</td>
<td>0.5</td>
</tr>
<tr>
<td>.ms</td>
<td>application/x-troff-ms</td>
<td>ebcdic</td>
<td>0.5</td>
</tr>
<tr>
<td>.src</td>
<td>application/x-wais-source</td>
<td>ebcdic</td>
<td>1.0</td>
</tr>
<tr>
<td>.bcpio</td>
<td>application/x-bcpio</td>
<td>binary</td>
<td>1.0</td>
</tr>
<tr>
<td>.cpio</td>
<td>application/x-cpio</td>
<td>binary</td>
<td>1.0</td>
</tr>
<tr>
<td>.gtar</td>
<td>application/x-gtar</td>
<td>binary</td>
<td>1.0</td>
</tr>
<tr>
<td>.sv4cpio</td>
<td>application/x-sv4cpio</td>
<td>binary</td>
<td>1.0</td>
</tr>
<tr>
<td>.sv4crc</td>
<td>application/x-sv4crc</td>
<td>binary</td>
<td>1.0</td>
</tr>
<tr>
<td>.wrl</td>
<td>x-world/x-vrml</td>
<td>binary</td>
<td>1.0</td>
</tr>
</tbody>
</table>

The following are neutral CAE formats:

<table>
<thead>
<tr>
<th>File Extension</th>
<th>MIME Type</th>
<th>Encoding</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>.igs</td>
<td>application/iges</td>
<td>binary</td>
<td>1.0</td>
</tr>
<tr>
<td>.iges</td>
<td>application/iges</td>
<td>binary</td>
<td>1.0</td>
</tr>
<tr>
<td>.IGS</td>
<td>application/iges</td>
<td>binary</td>
<td>1.0</td>
</tr>
<tr>
<td>.IGES</td>
<td>application/iges</td>
<td>binary</td>
<td>1.0</td>
</tr>
<tr>
<td>.stp</td>
<td>application/STEP</td>
<td>ebcdic</td>
<td>1.0</td>
</tr>
<tr>
<td>.STP</td>
<td>application/STEP</td>
<td>ebcdic</td>
<td>1.0</td>
</tr>
<tr>
<td>.STEP</td>
<td>application/STEP</td>
<td>ebcdic</td>
<td>1.0</td>
</tr>
<tr>
<td>.dxf</td>
<td>application/dxf</td>
<td>binary</td>
<td>1.0</td>
</tr>
<tr>
<td>.DXF</td>
<td>application/dxf</td>
<td>binary</td>
<td>1.0</td>
</tr>
<tr>
<td>.vda</td>
<td>application/vda</td>
<td>binary</td>
<td>1.0</td>
</tr>
<tr>
<td>.VDA</td>
<td>application/vda</td>
<td>binary</td>
<td>1.0</td>
</tr>
<tr>
<td>.set</td>
<td>application/set</td>
<td>ebcdic</td>
<td>1.0</td>
</tr>
<tr>
<td>.SET</td>
<td>application/set</td>
<td>ebcdic</td>
<td>1.0</td>
</tr>
<tr>
<td>.stl</td>
<td>application/SLA</td>
<td>ebcdic</td>
<td>1.0</td>
</tr>
<tr>
<td>.STL</td>
<td>application/SLA</td>
<td>ebcdic</td>
<td>1.0</td>
</tr>
</tbody>
</table>

The following are vendor-specific CAD-formats commonly used by CERN and in HEP institutes:

<table>
<thead>
<tr>
<th>File Extension</th>
<th>MIME Type</th>
<th>Encoding</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>.dwg</td>
<td>application/acad</td>
<td>binary</td>
<td>1.0</td>
</tr>
<tr>
<td>.DWG</td>
<td>application/acad</td>
<td>binary</td>
<td>1.0</td>
</tr>
<tr>
<td>.SOL</td>
<td>application/solids</td>
<td>binary</td>
<td>1.0</td>
</tr>
<tr>
<td>.DRW</td>
<td>application/drafting</td>
<td>binary</td>
<td>1.0</td>
</tr>
<tr>
<td>.prt</td>
<td>application/pro_eng</td>
<td>binary</td>
<td>1.0</td>
</tr>
<tr>
<td>.PRT</td>
<td>application/pro_eng</td>
<td>binary</td>
<td>1.0</td>
</tr>
<tr>
<td>.unv</td>
<td>application/i-deas</td>
<td>binary</td>
<td>1.0</td>
</tr>
<tr>
<td>.UNV</td>
<td>application/i-deas</td>
<td>binary</td>
<td>1.0</td>
</tr>
<tr>
<td>.CCAD</td>
<td>application/clariscad</td>
<td>binary</td>
<td>1.0</td>
</tr>
<tr>
<td>.snd</td>
<td>audio/basic</td>
<td>binary</td>
<td>1.0</td>
</tr>
<tr>
<td>.au</td>
<td>audio/basic</td>
<td>binary</td>
<td>1.0</td>
</tr>
<tr>
<td>.aiff</td>
<td>audio/x-aiff</td>
<td>binary</td>
<td>1.0</td>
</tr>
<tr>
<td>.aiffc</td>
<td>audio/x-aiff</td>
<td>binary</td>
<td>1.0</td>
</tr>
<tr>
<td>.aiff</td>
<td>audio/x-aiff</td>
<td>binary</td>
<td>1.0</td>
</tr>
<tr>
<td>.wav</td>
<td>audio/x-wav</td>
<td>binary</td>
<td>1.0</td>
</tr>
<tr>
<td>.bmp</td>
<td>image/bmp</td>
<td>binary</td>
<td>1.0</td>
</tr>
<tr>
<td>.gif</td>
<td>image/gif</td>
<td>binary</td>
<td>1.0</td>
</tr>
<tr>
<td>.ief</td>
<td>image/ief</td>
<td>binary</td>
<td>1.0</td>
</tr>
</tbody>
</table>
### Multi-format processing directives

<table>
<thead>
<tr>
<th>AddType</th>
<th>Extension</th>
<th>Type</th>
<th>Encoding</th>
<th>Compatibility</th>
</tr>
</thead>
<tbody>
<tr>
<td>.jpg</td>
<td>image/jpeg</td>
<td>binary 1.0</td>
<td></td>
<td># JPEG</td>
</tr>
<tr>
<td>.JPEG</td>
<td>image/jpeg</td>
<td>binary 1.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>.JPE</td>
<td>image/jpeg</td>
<td>binary 1.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>.JPEP</td>
<td>image/jpeg</td>
<td>binary 1.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>.JPEGP</td>
<td>image/jpeg</td>
<td>binary 1.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>.jpeg</td>
<td>image/jpeg</td>
<td>binary 1.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>.tiff</td>
<td>image/tiff</td>
<td>binary 1.0</td>
<td></td>
<td># TIFF</td>
</tr>
<tr>
<td>.tiff</td>
<td>image/tiff</td>
<td>binary 1.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>.ras</td>
<td>image/x-cmu-raster</td>
<td>binary 1.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>.pnm</td>
<td>image/x-portable-anymap</td>
<td>binary 1.0</td>
<td></td>
<td>PBM Anymap format</td>
</tr>
<tr>
<td>.pbm</td>
<td>image/x-portable-bitmap</td>
<td>binary 1.0</td>
<td></td>
<td>PBM Bitmap format</td>
</tr>
<tr>
<td>.pgm</td>
<td>image/x-portable-grayscale</td>
<td>binary 1.0</td>
<td></td>
<td>PBM Grayscale format</td>
</tr>
<tr>
<td>.ppm</td>
<td>image/x-portable-pixmap</td>
<td>binary 1.0</td>
<td></td>
<td>PBM Pixmap format</td>
</tr>
<tr>
<td>.rgb</td>
<td>image/x-rgb</td>
<td>binary 1.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>.xbm</td>
<td>image/x-xbitmap</td>
<td>ebcdic 1.0</td>
<td></td>
<td>X bitmap</td>
</tr>
<tr>
<td>.xpm</td>
<td>image/x-xpixmap</td>
<td>binary 1.0</td>
<td></td>
<td>X pixmap format</td>
</tr>
<tr>
<td>.xwd</td>
<td>image/x-xwindowdump</td>
<td>binary 1.0</td>
<td></td>
<td>X window dump (xwd)</td>
</tr>
<tr>
<td>.html</td>
<td>text/html</td>
<td>ebcdic 1.0</td>
<td></td>
<td>HTML</td>
</tr>
<tr>
<td>.htm</td>
<td>text/html</td>
<td>ebcdic 1.0</td>
<td></td>
<td>HTML on PCs</td>
</tr>
<tr>
<td>.htmls</td>
<td>text/x-skip-html</td>
<td>ebcdic 1.0</td>
<td></td>
<td>Server-side includes</td>
</tr>
<tr>
<td>.shtml</td>
<td>text/x-skip-html</td>
<td>ebcdic 1.0</td>
<td></td>
<td>Server-side includes</td>
</tr>
<tr>
<td>.c</td>
<td>text/plain</td>
<td>ebcdic 0.5</td>
<td></td>
<td>C source</td>
</tr>
<tr>
<td>.h</td>
<td>text/plain</td>
<td>ebcdic 0.5</td>
<td></td>
<td>C headers</td>
</tr>
<tr>
<td>.c</td>
<td>text/plain</td>
<td>ebcdic 0.5</td>
<td></td>
<td>C++ source</td>
</tr>
<tr>
<td>.cc</td>
<td>text/plain</td>
<td>ebcdic 0.5</td>
<td></td>
<td>C++ headers</td>
</tr>
<tr>
<td>.java</td>
<td>text/plain</td>
<td>ebcdic 0.5</td>
<td></td>
<td>Java source</td>
</tr>
<tr>
<td>.m</td>
<td>text/plain</td>
<td>ebcdic 0.5</td>
<td></td>
<td>Objective-C source</td>
</tr>
<tr>
<td>.f90</td>
<td>text/plain</td>
<td>ebcdic 0.5</td>
<td></td>
<td>Fortran 90 source</td>
</tr>
<tr>
<td>.plain</td>
<td>text/plain</td>
<td>ebcdic 0.5</td>
<td></td>
<td>Plain text</td>
</tr>
<tr>
<td>.css</td>
<td>text/css</td>
<td>ebcdic 0.9</td>
<td></td>
<td>W3C Cascading Style Sheets</td>
</tr>
<tr>
<td>.rtx</td>
<td>text/richtext</td>
<td>ebcdic 1.0</td>
<td></td>
<td>MIME RichText format</td>
</tr>
<tr>
<td>.tsv</td>
<td>text/tab-separated-values</td>
<td>ebcdic 1.0</td>
<td></td>
<td>Tab-separated values</td>
</tr>
<tr>
<td>.etx</td>
<td>text/x-setext</td>
<td>ebcdic 1.0</td>
<td></td>
<td>Structured Enhanced Text</td>
</tr>
<tr>
<td>.MPG</td>
<td>video/mpeg</td>
<td>binary 1.0</td>
<td></td>
<td>MPEG</td>
</tr>
<tr>
<td>.mpg</td>
<td>video/mpeg</td>
<td>binary 1.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>.MPE</td>
<td>video/mpeg</td>
<td>binary 1.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>.mpeg</td>
<td>video/mpeg</td>
<td>binary 1.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>.mpeg</td>
<td>video/mpeg</td>
<td>binary 1.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>.qt</td>
<td>video/quicktime</td>
<td>binary 1.0</td>
<td></td>
<td>QuickTime</td>
</tr>
<tr>
<td>.mov</td>
<td>video/quicktime</td>
<td>binary 1.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>.avi</td>
<td>video/x-msvideo</td>
<td>binary 1.0</td>
<td></td>
<td>MS Video for Windows</td>
</tr>
<tr>
<td>.movie</td>
<td>video/x-sgi-movie</td>
<td>binary 1.0</td>
<td></td>
<td>SGI movioplayer</td>
</tr>
<tr>
<td>.zip</td>
<td>multipart/x-zip</td>
<td>binary 1.0</td>
<td></td>
<td>PKZIP</td>
</tr>
<tr>
<td>.gz</td>
<td>application/x-compress</td>
<td>gzip 1.0</td>
<td></td>
<td>PKZIP</td>
</tr>
<tr>
<td>.tar</td>
<td>multipart/x-tar</td>
<td>binary 1.0</td>
<td></td>
<td>4.3BSD tar</td>
</tr>
<tr>
<td>.ustar</td>
<td>multipart/x-ustar</td>
<td>binary 1.0</td>
<td></td>
<td>POSIX tar</td>
</tr>
<tr>
<td>.*</td>
<td>www/unknown</td>
<td>binary 0.2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>.*</td>
<td>www/unknown</td>
<td>binary 0.2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>.cxx</td>
<td>text/plain</td>
<td>ebcdic 0.5</td>
<td></td>
<td>C++</td>
</tr>
<tr>
<td>.for</td>
<td>text/plain</td>
<td>ebcdic 0.5</td>
<td></td>
<td>Fortran</td>
</tr>
<tr>
<td>.mar</td>
<td>text/plain</td>
<td>ebcdic 0.5</td>
<td></td>
<td>MACRO</td>
</tr>
<tr>
<td>.log</td>
<td>text/plain</td>
<td>ebcdic 0.5</td>
<td></td>
<td>logfiles</td>
</tr>
<tr>
<td>.com</td>
<td>text/plain</td>
<td>ebcdic 0.5</td>
<td></td>
<td>scripts</td>
</tr>
<tr>
<td>.sdml</td>
<td>text/plain</td>
<td>ebcdic 0.5</td>
<td></td>
<td>SDML</td>
</tr>
<tr>
<td>.list</td>
<td>text/plain</td>
<td>ebcdic 0.5</td>
<td></td>
<td>listfiles</td>
</tr>
<tr>
<td>.lst</td>
<td>text/plain</td>
<td>ebcdic 0.5</td>
<td></td>
<td>listfiles</td>
</tr>
<tr>
<td>.def</td>
<td>text/plain</td>
<td>ebcdic 0.5</td>
<td></td>
<td>definition files</td>
</tr>
<tr>
<td>.conf</td>
<td>text/plain</td>
<td>ebcdic 0.5</td>
<td></td>
<td>definition files</td>
</tr>
<tr>
<td>.ini</td>
<td>text/plain</td>
<td>ebcdic 0.5</td>
<td></td>
<td>files with no extension</td>
</tr>
</tbody>
</table>

The following can be used to store ASCII DBCS:

AddType .JP932 text/x-DBCS binary 1.0  # IBM-932 Japanese DBCS
AddType .JPEuc text/x-DBCS binary 1.0  # IBM-932 Japanese DBCS
SuffixCaseSense - Specify whether suffix definitions are case sensitive

Use this directive to specify whether you want your server to distinguish between uppercase and lowercase letters when comparing file suffixes to the suffix patterns on AddClient, AddCharSet, AddType, AddEncoding, and AddLanguage directives. By default, the server does not distinguish between uppercase and lowercase.

Example
SuffixCaseSense On

Initial configuration file setting
SuffixCaseSense Off

AddClient - Specify file extensions for requesting clients

Use this directive to bind files with particular extensions to the type and version of client that is sending the request. This is often referred to as automatic browser detection.

All HTTP requests contain a User-Agent header that identifies the browser sending the request. The HTTP Server enables you to detect which browser was used to send a request and, based on this information, respond with a version of a Web page, a document, or other file that is appropriate for that browser.

For example, your server can send a page written in HTML 3.0 only to browsers that are known to support it and send a version of the same page written in HTML 2.0 to all other browsers.

Automatic browser detection is only effective for multi-format processing. If you use multi-format processing, a requesting URL specifies a file without an extension and no file with that name exists, or a requesting URL specifies a file with the .multi extension. For example, a link from this HTML anchor tag initiates multi-format processing:

```html
<A HREF="http://www.mycompany.com/mydept/tscores.multi">
```

As a result, the server will evaluate the values passed in the request headers, along with the extensions of all the tscores files and the associations specified in the directives. Based on this, it will try to find the file that is the best match to send in its response.

You can have multiple occurrences of this directive in your configuration file. The sequence of AddClient directives is important. The first AddClient directive that matches a client’s User-Agent value is the one that will be used to determine the file extension.

If a client’s User-Agent is not matched in an AddClient directive, the server looks for a generic file extension (.htm or .html) to send. If the server cannot find a generic file extension, it uses an algorithm to calculate the quality of all the extensions for that file and sends the file whose extension yields the highest quality, considering it to be the best match.

The format of the directive is:

```
AddClient .extension user-agent-header
```
Multi-format processing directives

.extension

The file extension you specify for the file you want to send to a particular browser.

This extension can be one of a string of suffixes used to qualify a file. For example, the extension .Netscape can apply to a file named TxtSample.UK.Netscape.html or TxtSample.html.Netscape.eng. You cannot use any wildcard characters in the file extension.

user-agent-header

This field must match the value in the User-Agent header of the incoming request.

This field is case-sensitive. You can use an asterisk (*) as a wildcard character in this field or include the complete header information. Use quotes to include white space in the user-agent name.

Examples:

- Mozilla/2.* applies to all levels of Version 2 of Netscape’s browser, Netscape Navigator.
- "Mozilla/4.0 (compatible; MSIE)" applies to Microsoft’s browser, Internet Explorer Version 4.0.

Examples

AddClient .Netscape Mozilla/2.*
AddClient .MSIE "Mozilla/4.0 (compatible; MSIE)"

Program default setting

If a client’s User-Agent is not matched in an AddClient directive, the server looks for a generic file extension (.htm or .html) to send. If the server cannot find a generic file extension, it uses an algorithm to calculate the quality of all the extensions for that file and sends the file whose extension yields the highest quality, considering it to be the ‘best match’.

Initial configuration file setting

None

Using automatic browser detection for Welcome pages

This example describes how to enable automatic browser detection for Welcome pages. It shows how to serve different versions of the index.html file in the webhome directory. You can also use this example for Welcome.html, welcome.html, or Frntpage.html files.

Follow these steps to enable browser detection and multi-format processing for your welcome pages:

1. Add a Welcome directive to your configuration file that specifies the file name with the .multi extension:

   Welcome index.multi

2. Include AddClient directives in the configuration file that specify which file extensions to send to a particular browser. You can use an asterisk (*) as a wildcard character or include the complete header information. Use quotes to include white space in the user-agent name.

   Examples:

   AddClient .Netscape Mozilla/2.*
   AddClient .MSIE "Mozilla/4.0 (compatible; MSIE)"

3. Create specific versions of the index file for each of these file extensions:

   index.Netscape.html and index.MSIE.html.
4. Create a dummy file in the directory called index.multi.

5. Specify only the directory name in the URL when linking to this page.

http://www.web4hire.com/webhome/

**Resource mapping - Redirect URLs**

Use the directives described in this section to control which requests your server accepts and where the server looks for resources.

Use the mapping directives (Exec, Fail, Map, Pass, and Redirect) to control which requests your server accepts and to map URL requests to your actual files.

You can use the mapping directives to create a virtual hierarchy of Web resources. You can then change the physical location of files or directories without affecting the virtual layout. Even if your server sends documents from different file systems, it can present a virtual layout.

The server applies the mapping directives in the order they appear in the configuration file until a request has been accepted, rejected, or there are no more directives that apply to the request.

**Exec - Run a CGI program for matching requests**

Use this directive to specify a template for requests you want to accept and respond to by running a CGI program. Once a request matches a template on an Exec directive, the request is not compared to request templates on any subsequent directives.

The format of the directive is:

```
Exec request-template program-path [Server-IP-address or hostname]
```

**request-template**

A template for requests that you want your server to accept and respond to by running a CGI program.

You must use an asterisk as a wildcard in both the `request-template` and `program-path`. The part of the request that matches the `request-template` wildcard must begin with the name of the file that contains the CGI program. The wildcard is optional, but if used, it must be in both the `request-template` and `program-path`.

The request can also contain additional data that is passed to the CGI program in the PATH_INFO environment variable. The additional data follows the first slash character that comes after the CGI program file name on the request. The data is passed according to CGI specifications.

**program-path**

The path to the file that contains the CGI program you want the server to execute for the request. `program-path` must also contain a wildcard. The wildcard is replaced with the name of the file that contains the CGI program. The wildcard is optional, but if used, it must be in both the `request-template` and `program-path`.

The Exec directive is recursive and applies to all subdirectories. You do not need a separate Exec directive for each directory under cgi-bin and admin-bin.
Server-IP-address or hostname

If you are using multiple IP addresses or virtual hosts, use this parameter to specify an IP address or a host name. (For more information on using multiple IP addresses or virtual hosts, see "Chapter 16. Running your server with multiple IP addresses or virtual hosts" on page 165.) The server uses the directive only for requests that come to the server on this IP address or for this host. For an IP address, this is the address of the server’s network connection, not the address of the requesting client.

You can specify an IP address (for example, 204.146.167.72) or you can specify a host name (for example, hostA.bcd.com).

This parameter is optional. Without this parameter, the server uses the directive for all requests regardless of the IP address the requests come in on or the host name in the URL.

A wildcard character cannot be specified for a server’s IP address.

Note: If a user requests a URL that matches an Exec directive and the resulting path specifies a directory, the server either fails the request with a 404 error code, or treats this situation as if a Pass directive were matched instead, depending on the setting of the ExecDirPass directive. See the ExecDirPass directive for more information.

If an Exec directive results in a Java CGI being executed, the server invokes the Java Virtual Machine, passing it the name of the class file. The Java Virtual Machine finds the file in the list of directories specified in the CLASSPATH environment variable and executes the first instance of the file in this list. This may not be the same path that is specified in the program path value.

For more information on how to set up class files for use as Java CGI’s, see "Chapter 17. Writing Common Gateway Interface programs" on page 171.

Examples

Exec /idd/depts/* /depts/bin/*

In this example, if your server receives a request of /idd/depts/plan/c92, it attempts to run a CGI program or script matching the following name:

/depts/bin/plan

In the example, c92 is then passed to the first matching CGI program or script as input.

Initial configuration file setting

Exec /cgi-bin/* /usr/lpp/internet/server_root/cgi-bin/*
Exec /admin-bin/* /usr/lpp/internet/server_root/admin-bin/*

The HTTP Server supports CERN preparsed server scripts; however, it is recommended that they be upgraded to CGI scripts. Preparsed server scripts must end with the suffix of .pp when being referenced in URLs. The text following the script name in the URL is always treated by the server as the first parameter to the script. For more information about CERN preparsed server scripts, go to URL:

http://www.w3.org/pub/WWW/Daemon/User/HTBinDoc.html
ExecDirPass - Control access to directories matching Exec directives

Use this directive to specify whether you want your server to allow implicit access to directory names matching Exec directives. The purpose of the Exec directive is to have the server execute a CGI, but if a URL is requested that resolves to only a directory name, this is obviously not a CGI, so it cannot be executed. If you want to let users access directories in this way, use:

    ExecDirPass On

The server treats the Exec directive as if it is a Pass directive, and returns a directory listing or welcome page, if one is available. See the DirAccess and AlwaysWelcome directives for more information on how passed directories are handled.

If you specify ExecDirPass On, you do not need to specify separate Pass statements to allow directory listings of CGI directories. If you do not want to implicitly allow directory access to all directories matching Exec directives, then you must still specify Pass directives for those that you want to allow access to.

Note: Use care when deciding to use this directive. If you enable this function, the names of all files (including CGIs) will become visible when browsing the resulting directory. This may give end users insight into your system conventions, which may expose sensitive information.

If you do not specify this directive, or if you specify Off, the server returns a 404 Failed By Rule error.

Examples

    ExecDirPass On
    ExecDirPass Off

Program default setting

    ExecDirPass Off

Fail - Reject matching requests

Use this directive to specify a template for requests you do not want to process. Once a request matches a template on a Fail directive, the request is not compared to request templates on any subsequent directives.

The format of the directive is:

    Fail request-template [Server-IP-address or hostname]

request-template

A template for requests that you want your server to reject. If a request matches the template, the server sends the requester an error message.

You can use an asterisk as a wildcard in the template. The tilde character just after a slash (/) has to be explicitly matched; a wildcard cannot be used to match it.

Server-IP-address or hostname

If you are using multiple IP addresses or virtual hosts, use this parameter to specify an IP address or a host name. (For more information on using multiple IP addresses or virtual hosts, see [Chapter 16. Running your server with multiple IP addresses or virtual hosts on page 163].) The server uses the
Resource mapping directives

directive only for requests that come to the server on this IP address or for this
host. For an IP address, this is the address of the server’s network connection,
not the address of the requesting client.

You can specify an IP address (for example, 204.146.167.72) or you can specify
a host name (for example, hostA.bcd.com).

This parameter is optional. Without this parameter, the server uses the
directive for all requests regardless of the IP address the requests come in on
or the host name in the URL.

A wildcard character cannot be specified for a server’s IP address.

Examples

Fail /usr/local/private/*

In the above example, the server rejects any requests beginning with
/usr/local/private/.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Fail</td>
<td>/customerB/* 204.146.167.72</td>
</tr>
<tr>
<td>Fail</td>
<td>/customerA/* 9.83.100.45</td>
</tr>
</tbody>
</table>

The above example uses the optional IP address parameter. The server rejects any
requests beginning with /customerB/ if the request comes in on a network
connection with IP address 204.146.167.72. The server rejects any requests
beginning with /customerA/ if the request comes in on a network connection with
an IP address of 9.83.100.45.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Fail</td>
<td>/customerB/* hostA.bcd.com</td>
</tr>
<tr>
<td>Fail</td>
<td>/customerA/* hostB.bcd.com</td>
</tr>
</tbody>
</table>

The above example uses the optional hostname parameter. The server rejects any
requests beginning with /customerB/ if the request comes in for hostA.bcd.com.
The server rejects any requests beginning with /customerA/ if the request comes
in for hostB.bcd.com.

Initial configuration file setting
None.

Map - Change matching requests to a new result string

Use this directive to specify a template for requests you want to change to a new
request string. After your server changes the request, it takes the new request
string and compares it to the request templates on subsequent directives.

The format of the directive is:

Map request-template new-request [Server-IP-address or hostname]

request-template
A template for requests that you want your server to change and then continue
comparing the new request string to other templates.

You can use an asterisk as a wildcard in the template. The tilde character just
after a slash (/) has to be explicitly matched; a wildcard cannot be used to
match it.

new-request
The new request string you want your server to continue to compare to the
request templates on subsequent directives. new-request may contain a wildcard
Resource mapping directives

if the request-template has one. The part of the request that matches the request-template wildcard is inserted in place of the wildcard in new-request.

Server-IP-address or hostname

If you are using multiple IP addresses or virtual hosts, use this parameter to specify an IP address or a host name. (For more information on using multiple IP addresses or virtual hosts, see [Chapter 16. Running your server with multiple IP addresses or virtual hosts](#).) The server uses the directive only for requests that come to the server on this IP address or for this host. For an IP address, this is the address of the server’s network connection, not the address of the requesting client.

You can specify an IP address (for example, 204.146.167.72) or you can specify a host name (for example, hostA.raleigh.ibm.com).

This parameter is optional. Without this parameter, the server uses the directive for all requests regardless of the IP address the requests come in on or the host name in the URL.

A wildcard character cannot be specified for a server’s IP address.

Examples

Map /stuff/* /good/stuff/*

In the above example, your server would take any requests starting with /stuff/ and change the /stuff/ portion of the request to /good/stuff/. Anything that followed /stuff/ on the original request would also be included in the new request string. So /stuff/whatsup/ would change to /good/stuff/whatsup/. Your server would take the new request string and continue to compare it to request templates on subsequent directives.

Map /stuff/* /customerA/good/stuff/* 204.146.167.72
Map /stuff/* /customerB/good/stuff/* 9.83.104.45

The above examples use the optional IP address parameter. If your server receives requests that begin with /stuff/, it changes the request to a different request string based on the IP address of the network connection the request comes in on. For requests coming in on 204.146.167.72, the server changes the /stuff/ portion of the request to /customerA/good/stuff/. For requests coming in on any connection with an address of 9.83.100.45, the server changes the /stuff/ portion of the request to /customerB/good/stuff/.

Map /stuff/* /customerA/good/stuff/* hostA.bcd.com
Map /stuff/* /customerB/good/stuff/* hostB.bcd.com

The above examples use the optional host name parameter. If your server receives requests that begin with /stuff/, it changes the request to a different request string based on the host name in the URL. For requests coming in for hostA, the server changes the /stuff/ portion of the request to /customerA/good/stuff/. For requests coming in for hostB, the server changes the /stuff/ portion of the request to /customerB/good/stuff/.

Initial configuration file setting

None.
Resource mapping directives

Pass - Accept matching requests

Use this directive to specify a template for requests you want to accept and respond to with a document from your server. Once a request matches a template on a Pass directive, the request is not compared to request templates on any subsequent directives.

The format of the directive is:

```
Pass  request-template [file-path [Server-IP-address or hostname]]
```

`request-template`

A template for requests that you want your server to accept and respond to with a document from your server.

You can use an asterisk as a wildcard in the template. The tilde character just after a slash (/) has to be explicitly matched; a wildcard cannot be used to match it.

`file-path`

The path to the file that contains the document you want the server to return. `file-path` may contain a wildcard if the `request-template` has one. The part of the request that matches the `request-template` wildcard is inserted in place of the wildcard in `file-path`.

This parameter is optional. If you do not specify a path, the request itself is used as the path.

`Server-IP-address` or `hostname`

If you are using multiple IP addresses or virtual hosts, use this parameter to specify an IP address or a host name. (For more information on using multiple IP addresses or virtual hosts, see “Chapter 16. Running your server with multiple IP addresses or virtual hosts” on page 165.) The server uses the directive only for requests that come to the server on this IP address or for this host. For an IP address, this is the address of the server’s network connection, not the address of the requesting client.

You can specify an IP address (for example, 204.146.167.72) or you can specify a host name (for example, hostA.raleigh.ibm.com).

This parameter is optional. Without this parameter, the server uses the directive for all requests regardless of the IP address the requests come in on or the host name in the URL.

A wildcard character cannot be specified for a server’s IP address.

Examples

```
Pass /updates/parts/* /usr/lpp/internet/server_root/pub/*
```

In the above, your server would respond to a request starting with

`/updates/parts/` with a document from `/usr/lpp/internet/server_root/pub/`. Anything that followed `/updates/parts/` would also be used to identify the document. So your server would respond to the request

`/updates/parts/printers/ribbon.html` with the document in file

`/usr/lpp/internet/server_root/printers/ribbon.html`.

```
Pass /gooddoc/*
```

In the above example, your server would respond to a request starting with

`/gooddoc/` with a document from `/gooddoc`. So your server would respond to the
Resource mapping directives


Pass /parts/* /customerA/catalog/* 204.146.167.72
Pass /parts/* /customerB/catalog/* 9.83.100.45

The above examples use the optional IP address parameter. If your server receives requests that begin with /parts/, it returns a file from a different directory based on the IP address of the network connection the request comes in on. For requests coming in on 204.146.167.72 the server returns a file from /customerA/catalog/. For requests coming in on any connection with an address of 9.83.100.45, the server returns a file from /customerB/catalog/.

Pass /parts/* /customerA/catalog/* hostA.bcd.com
Pass /parts/* /customerB/catalog/* hostB.bcd.com

The above examples use the optional host name parameter. If your server receives requests that begin with /parts/, it returns a file from a different directory based on the host name in the URL. For requests coming in for hostA, the server returns a file from /customerA/catalog/. For requests coming in for hostB, the server returns a file from /customerB/catalog/.

Initial configuration file setting

Pass /icons/* /usr/lpp/internet/server_root/icons/*
Pass /Admin/* /usr/lpp/internet/server_root/Admin/*.html
Pass /Admin/* /usr/lpp/internet/server_root/Admin/*.gif
Pass /Docs/* /usr/lpp/internet/server_root/Docs/*
Pass /img-bin/* /usr/lpp/internet/server_root/img-bin/*
Pass /reports/* /usr/lpp/internet/server_root/pubs/reports/*
Pass /* /usr/lpp/internet/server_root/pub/*

The path used for Pass /* is your document root directory. Pass /* matches all local server requests that do not match any other Pass directives.

Note: Only protect and pass rules for proxy should follow Pass /*.

Redirect - Send matching requests to another server

Use this directive to specify a template for requests you want to accept and send to another server. Once a request matches a template on a Redirect directive, the request is not compared to templates on any other directives in your configuration file.

The format of the directive is:

Redirect request-template URL [Server-IP-address or hostname]

request-template
A template for requests that you want your server to send to another server.

You can use an asterisk as a wildcard in the template. The tilde character just after a slash (/) has to be explicitly matched; a wildcard cannot be used to match it.

URL
The URL request you want your server to send to another server. The response to this request goes to the original requester without any indication that it did not come from your server.

URL must contain a protocol specification and the name of the server to send the request to. It can also contain a path or file name. If request-template uses a
resource mapping directives

wildcard, the path or file name on URL can also use a wildcard. The part of the original request that matches the wildcard on request-template is inserted in place of the wildcard on URL.

server-ip-address or hostname
If you are using multiple IP addresses or virtual hosts, use this parameter to specify an IP address or a host name. (For more information on using multiple IP addresses or virtual hosts, see “Chapter 16. Running your server with multiple IP addresses or virtual hosts” on page 162.) The server uses the directive only for requests that come to the server on this IP address or for this host. For an IP address, this is the address of the server’s network connection, not the address of the requesting client.

You can specify an IP address (for example, 204.146.167.72) or you can specify a host name (for example, hostA.bcd.com).

This parameter is optional. Without this parameter, the server uses the directive for all requests regardless of the IP address the requests come in on or the host name in the URL.

A wildcard character cannot be specified for a server’s IP address.

example

Redirect /chief/stuff/* http://www.other.org/wahoo/*

In this example, your server sends any requests beginning with /chief/stuff/ to the wahoo directory of the www.other.org server.

    Redirect /stuff/* http://www.chief.org/wahoo/* 204.146.167.72
    Redirect /stuff/* http://www.chief.org/wahoo/* 204.146.167.72
    Redirect /stuff/* http://www.dawg.com/pound/* 9.83.100.45

The above examples use the optional IP address parameter. If your server receives requests that begin with /stuff/, it redirects the request to different servers based on the IP address of the network connection the request comes in on. For requests coming in on 204.146.167.72, the server sends the request to the wahoo directory of the www.chief.org server. For requests coming in on any connection with an address of 9.83.100.45, the server sends the request to the pound directory of the www.dawg.com server.

    Redirect /stuff/* http://www.chief.org/wahoo/* hostA.bcd.com
    Redirect /stuff/* http://www.dawg.com/pound/* hostB.bcd.com

The above examples use the optional IP address parameter. If your server receives requests that begin with /stuff/, it redirects the request to different servers based on the host name in the URL. For requests coming in for hostA, the server sends the request to the wahoo directory of the www.chief.org server. For requests coming in for hostB, the server sends the request to the pound directory of the www.dawg.com server.

initial configuration file setting
None.

inheritenv - specify which environment variables are inherited by CGI programs
Use this directive to specify which environment variables you want your CGI programs to inherit (other than the CGI environment variables that are specific to CGI processing).
Resource mapping directives

If you do not include an InheritEnv directive, all environment variables are inherited by CGI programs. If you include any InheritEnv directive, only those environment variables specified on InheritEnv directives will be inherited along with the CGI-specific environment variables. The directive allows you to optionally initialize the value of the variables that are inherited.

For a list of the CGI-specific environment variables, see "Chapter 17. Writing Common Gateway Interface programs" on page 171.

Example

InheritEnv PATH
InheritEnv LANG=ENUS

In this example, only the PATH and LANG environment variables will be inherited by CGI programs and the LANG environment variable will be initialized with the value of ENUS.

Initial configuration file setting
None. The default is all environment variables are inherited by CGI programs.

DisInheritEnv - Specify which environment variables are disinherited by CGI programs

Use this directive to specify which environment variables you do not want your CGI programs to inherit (other than the CGI environment variables that are specific to CGI processing).

By default, all environment variables are inherited by CGI programs. You can exclude individual environment variables from being inherited with the DisInheritEnv directive.

For a list of the CGI-specific environment variables, see "Chapter 17. Writing Common Gateway Interface programs" on page 171.

Example

DisInheritEnv PATH
DisInheritEnv LANG

In this example, all environment variables except PATH and LANG will be inherited by CGI programs.

Initial configuration file setting
None. The default is all environment variables are inherited by CGI programs.

Security - Set up secure connections for the server

For a secure server, use the directives described in this section to control how your server uses network security functions.

If you change these directives, you must stop your server and then start it again for the changes to take effect. The server will not pick up the changes if you only restart it.

KeyFile - Set name for key database file

Use this directive to set the name for a key database file. A key database is a file where you keep public-private key pairs, certificates, and trusted CA certificates.
Security directives

If you change this directive, you must stop your server and then start it again for the change to take effect. The server will not pick up the change if you only restart it.

You do not have a secure network connection until you have created a key for secure network communications and received a certificate from a certificate authority (CA) who is designated as a trusted CA on your server. Use the IBM Key Management Utility (IKEYMAN) to create key databases, public-private key pairs, and certificate requests. If you are acting as your own CA, you can use IKEYMAN to create self-signed certificates. For an example, see "Setting up secure connections using self-signed certificates" on page 62.

You can have multiple KeyFile directives in your configuration file. If you have multiple KeyFile directives, the default key database for the server is the database shown on the last KeyFile directive.

Example

```
KeyFile key.kdb
KeyFile mycompany.kdb
```

This example specifies that the default key database for the server is mycompany.kdb.

Note: By default, key database files are located in the server root directory. If you move your key database files to another directory, you must specify the complete path on the KeyFile directive, for example:

```
KeyFile /security/kdbfiles/mycompany.kdb
```

Initial configuration file setting

```
KeyFile key.kdb
```

Program default setting

```
None
```

NormalMode - Turn port on or off for HTTP

Use this directive to turn the port defined by the Port directive on or off.

Set NormalMode on for an HTTP connection. If you also want an SSL connection, set SSLMode on.

Note: If both NormalMode and SSLMode are turned off, the server will start in normal mode, and you won't have a secure network connection.

If you change this directive, you must stop your server and then start it again for the change to take effect. The server will not pick up the change if you only restart it.

Initial configuration file setting

```
NormalMode on
```

Program default setting

```
NormalMode on
```
SSLCipherSpec - Specify levels of encryption supported by the server

Use this directive to specify which cipher specifications the server will support. For more information on encryption support, see “Encryption support for the HTTP Server” on page 55.

If you change this directive, you must stop your server and then start it again for the change to take effect. The server will not pick up the change if you only restart it.

The format of this directive is:

```
SSLCipherSpec cipher_specification
```

cipher_specification

A 2-digit code representing one of the cipher specifications that will be supported. More than one of these directives can appear in the server configuration file, httpd.conf. The order of the SSLCipherSpec directives determines the order in which the server will use the specified cipher specifications when attempting to establish secure communications with the requester.

Valid codes for cipher_specification in order from highest to lowest level of encryption:

**SSL V2:**
27    Triple-DES
21    RC4 (128 bit)
23    RC2 (128 bit)
26    DES
22    RC4 (40 bit)
24    RC2 (40 bit)

**SSL V3:**
3A    Triple-DES SHA
35    RC4 SHA (128 bit)
34    RC4 MD5 (128 bit)
39    DES SHA
33    RC4 MD5 (40 bit)
36    RC2 MD5 (40 bit)

**Example**

```
SSLCipherSpec 3A
SSLCipherSpec 35
SSLCipherSpec 27
SSLCipherSpec 26
```

This example specifies that the following encryption methods will be supported by the server in this order:

<table>
<thead>
<tr>
<th>Code</th>
<th>Cipher specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>3A</td>
<td>SSL V3-Triple-DES SHA</td>
</tr>
<tr>
<td>35</td>
<td>SSL V3-RC4 SHA</td>
</tr>
<tr>
<td>27</td>
<td>SSL V2-RC4 SHA</td>
</tr>
<tr>
<td>26</td>
<td>SSL V2-Triple-DES</td>
</tr>
</tbody>
</table>
Security directives

Any other cipher specifications requested by clients will result in an SSL handshake failure, and an error message will be logged in the error file. If the required specification is not included in the directive, an error message is logged in the error file.

Initial configuration file setting

All available cipher specifications are enabled.

Program default setting

The server will use the strongest encryption level supported by both the client and the server.

SSLClientAuth - Select the type of SSL client authentication

Use this directive to select the type of SSL client authentication you want to use.

When you use client authentication, the server requests a certificate from each client that makes an https request. Only use client authentication if you need to authenticate clients; SSL client authentication increases network traffic.

The SSLClientAuth directive can be specified in any of the following formats:

OFF The secure server does not request certificates from clients.

LOCAL The secure server requests certificates from clients. The server validates clients by checking for trusted CA certificates in the local key database. A trusted CA certificate is a certificate signed by a certificate authority who is designated as a trusted CA on your server.

PASSTHRU The secure server requests certificates from clients but does not do any validity checking. This option should be used only be used if alternate validation methods are available, for example, checking performed by a GWAPI or CGI program.

For more information on GWAPI and CGI programs, see:

- "Chapter 17. Writing Common Gateway Interface programs" on page 171
- "Chapter 18. Writing GWAPI programs" on page 197

STRONG

The secure server retrieves trusted CA certificates from the X.500 directory server specified on the SSLX500Host directive.

If the certificate authority software used for the client certificate is provided by the IBM Registry and certificate revocation lists (CRLs) are supported in the certificate, the secure server retrieves them. For information on the IBM Registry, go to URL http://www.ibm.com/software/commerce/registry/.

Note: If you specify STRONG authentication, you must specify the host name or IP address of the X.500 directory server using the SSLX500Host directive. Other directives that control the use of strong authentication are SSLX500Port, SSLX500UserID, SSLX500Password, SSLV2Timeout, and SSLV3Timeout.

If you change this directive, you must stop your server and then start it again for the change to take effect. The server will not pick up the change if you only restart it.
SSLClientAuth off

Program default setting
SSLClientAuth off

SSLMode - Turn port on or off for SSL
Use this directive to turn the port defined by the SSLPort directive on or off.

Set SSLMode on for an SSL connection. If you also want an HTTP connection, set NormalMode on.

Note: If both NormalMode and SSLMode are turned off, the server will start in normal mode, and you won't have a secure network connection.

If you change this directive, you must stop your server and then start it again for the change to take effect. The server will not pick up the change if you only restart it.

Initial configuration file setting
SSLMode on

Program default setting
SSLMode on

SSLPort - Set port for SSL security
Use this directive to set the port for SSL security. The server will use this port only for HTTPS requests. (Requests for HTTP will still come on the port that you set with the Port directive.)

If you want to use a port other than 443, specify a port above 1024.

If you change this directive, you must stop your server and then start it again for the change to take effect. The server will not pick up the change if you only restart it.

Initial configuration file setting
SSLPort 443

Program default setting
SSLPort 443

SSLServerCert - Associate a key database with an IP address
Use this directive to assign a key database label to each IP address the HTTP Server listens on.

If you change this directive, you must stop your server and then start it again for the change to take effect. The server will not pick up the change if you only restart it.

The format of this directive is:
SSLServerCert keyname IP-address

keyname
The name of the entry in the key database file that contains the keys and
Security directives

certificate for the specified host name-IP address combination. The value for the keyname parameter can contain blanks and must include a delimiter. Any delimiter can be used. The server assumes the first and last characters are the delimiters and must be the same character. All entries for the keyname parameter must be in the same key database as specified by the KeyFile directive.

IP-address

The IP address assigned to a host name configured for the server.

Examples

In the following examples, My Server's Certificate is the certificate label in the key database file.

SSLServerCert "My Server's Certificate" 19.123.55.12
SSLServerCert /My Server's Certificate/ 19.123.55.12
SSLServerCert zMy Server's Certificatez 19.123.55.12

The following examples will result in errors during server initialization:

SSLServerCert My Server's Certificate 19.123.55.12
SSLServerCert 'My Server's Certificate' 19.123.55.12
SSLServerCert 19.123.55.12

Initial configuration file setting

None. An example is included in the server configuration file for all of its IP addresses.

Program default setting

The server will use the default certificate from the key database file for all of its IP addresses.

Hints and Tips

- SSLServerCert can be placed anywhere in the configuration file and can appear multiple times.
- If more than one key database label is specified for the same IP address, the last value is the one that is used.
- If SSLServerCert is not specified and SSLMode is on, the server uses the key and certificate marked as the default in the key database file.
- If SSLServerCert is specified for some IP addresses (and not for others), the default key and certificate are used for those not specified on the SSLServerCert keyword.

SSLV2Timeout — Set SSL V2 session timeout value

Use this directive to specify the length of time that a browser can reuse an SSL V2 session ID without renegotiating encryption keys with the server.

If you change this directive, you must stop your server and then start it again for the change to take effect. The server will not pick up the change if you only restart it.

Initial configuration file setting

SSLV2Timeout 100 seconds

Program default setting

SSLV2Timeout 100 seconds
SSLV3Timeout — Set SSL V3 session timeout value

Use this directive to specify the length of time that a browser can reuse an SSL V3 session ID without renegotiating encryption keys with the server.

If you change this directive, you must stop your server and then start it again for the change to take effect. The server will not pick up the change if you only restart it.

Initial configuration file setting
SSLV3Timeout 1000 seconds

Program default setting
SSLV3Timeout 1000 seconds

SSLX500CARoot — Specify location of trusted CA certificates

Use this directive to specify where the secure server will search for trusted CA certificates to validate client certificates.

The SSLX500CARoot directive can be specified in any of the following formats:

LOCAL_ONLY
The secure server uses trusted CA certificates stored in the server’s local key database.

LOCAL_AND_X500
The secure server uses trusted CA certificates stored in the server’s local key database and in an X.500 directory server.

The server first checks the local key database. If a certificate is not found, the server searches the X.500 directory server. Certificates are retrieved from an X.500 directory server using the Lightweight Directory Access Protocol (LDAP). For more information on LDAP support and options, see "Chapter 14. Retrieving LDAP information" on page 153.

If you change this directive, you must stop your server and then start it again for the change to take effect. The server will not pick up the change if you only restart it.

Initial configuration file setting
SSLX500CARoot local_only

Program default setting
SSLX500CARoot local_only

SSLX500Host — Specify host name or IP address of the X.500 directory server

Use this directive to specify either the fully qualified host name or IP address of the X.500 directory server. This directive is required if you specify STRONG on the SSLClientAuth directive.

Examples:
SSLX500Host: www.my_x500_host.com
SSLX500Host: 9.97.123.176

If you change this directive, you must stop your server and then start it again for the change to take effect. The server will not pick up the change if you only restart it.

**Initial configuration file setting**
None

**Program default setting**
None

**SSLX500Port — Specify X.500 directory server port number**
Use this directive to specify the port number used for communicating with the X.500 directory server. Specify a value in the range 1-64000.

Example:

SSLX500Port 22343

If you change this directive, you must stop your server and then start it again for the change to take effect. The server will not pick up the change if you only restart it.

**Initial configuration file setting**
None

**Program default setting**
None

**SSLX500UserID — Specify user ID for the LDAP connection to the X.500 directory server**
Use this directive to specify the distinguished name of the user ID for the Lightweight Directory Access Protocol (LDAP) connection to the X.500 directory server. If you do not specify a distinguished name, anonymous will be used.

If you change this directive, you must stop your server and then start it again for the change to take effect. The server will not pick up the change if you only restart it.

**Initial configuration file setting**
None

**Program default setting**
SSLX500UserID Anonymous
SSLX500Password — Specify user ID password for the LDAP connection to the X.500 directory server

Use this directive to specify the password of the distinguished name of the user ID for the Lightweight Directory Access Protocol (LDAP) connection to the X.500 directory server. This password is valid only if SSLX500UserID has been specified.

You can use the following characters and digits when specifying the password: A-Z, a-z, 0-9.

If you change this directive, you must stop your server and then start it again for the change to take effect. The server will not pick up the change if you only restart it.

Initial configuration file setting
None

Program default setting
None

System Management - Define system management settings

Use the directives described in this section to define and tailor settings used to monitor and manage the performance, throughput, and activity of your servers.

ApplEnv - Specify application environment for workload management

Use this directive to specify the application environments and workload management transaction classes for processing URL requests matching a given template. Each application environment is assigned its own transaction queue. This directive tells the server how to associate requests with a queue. The format of the directive is:

ApplEnv url_template AEName [WLM_transaction_class [ip_addr_template]]

ApplEnv
Specifies a unique name for each queue

url_template
A template for requests that you want your server to accept and respond to with a document from your server. You can use an asterisk as a wildcard in the template. The tilde character just after a slash (/) has to be explicitly matched; a wildcard cannot be used to match it.

AEName
The application environment name can be any alphanumeric name up to thirty-two characters or a single wildcard (*). If a wildcard (*) is specified, the current workload management queue is used. Any application environment names that are referenced by using this directive, MUST also be defined under Application Environments in the WLM application panels. For more information about WLM application panels, see "Workload Management Enablement for the Web server" on page 107 or the OS/390 Planning: Workload Management book.

WLM_transaction_class
Optionally, you can specify the WLM transaction class. The transaction class is an input to WLM classification. Your WLM classification rules for IWEB can use the transaction class qualifier to map the transaction class to a service class.
System management directives

The transaction class can be any alphanumeric name up to eight characters or a single wildcard (*). If a wildcard is specified, the transaction class is not provided to WLM classification. When using HTTP1.1 persistent connections, the first request that matches an ApplEnv directive with an AEName results in classification using the specified transaction class. All subsequent requests on that connection are processed as continuations of one WLM transaction, even if routed to another application environment. The HTTP Server provides the following input for classification:

- **TRANSACTION_CLASS**: WLM_transaction_class (if not *)
- **TRANSACTION_NAME**: HTTP Request Line Method (for example, GET, POST)
- **USERID**: Server's LOGIN user ID
- **SUBSYSTEM_PARM**: Subsystem PARM buffer:
  - 0..7 Subsystem name (-SN xxx)
  - 8 Blank
  - 9..23 Source IPAddr nnn.nnn.nnn.nnn
  - 24 Blank
  - 25..39 Target IPAddr nnn.nnn.nnn.nnn
  - 40 Blank
  - 41..46 Target Port nnnnnn
  - 47 X'00'

Any transaction classes that are referenced using this directive MUST also be defined in the Classification Rules for IWEB using the TN (transaction name) qualifier in the WLM application panels. For more information about WLM application panels, see "Workload Management Enablement for the Web server" on page 107 or the OS/390 MVS Planning: Workload Management book.

**ip_addr_template**

If you are using multiple IP addresses, use this parameter to specify which IP address this directive applies to.

**Example**

```
APPLENV /Admin/*.html WEBHTML ADMIN
```

**Initial configuration file setting**

None.

**ApplEnvConfig - Specify a set of directives to tailor the application environment**

Use this directive to specify a set of directives to tailor application environments to support various workloads and conserve resources. Specify a subset of directives so your application environments are each configured appropriately. Each application environment can be specifically configured to handle the requests that are processed for that Application Environment.

```
ApplEnvConfig AEName|%%QS%%|%%QM%%
```

directive
directive
directive

```
)
```

**ApplEnvConfig**

Specifies a set of directives to tailor a particular application environment

**AENAME**

The application environment name can be any alphanumeric name up to thirty-two characters that is referenced by using this directive and MUST also be defined under application environments in the WLM application panels.
other words, this must be one of the AENames specified on the ApplEnv directive. Refer to "ApplEnv - Specify application environment for workload management" on page 375.

For more information about WLM application panels, see "Workload Management Enablement for the Web server" on page 107 of the OS/390 Planning: Workload Management book. You can replace AEName with either of the following:

%%QS%%
This enables directives specified within the ApplEnvConfig to affect all queue servers. Use this statement to distinguish between the queue manager and the queue servers.

%%QM%%
This enables directives specified within the ApplEnvConfig set to affect the queue manager only. Use %%QM%% to specify directives exclusively for the queue manager, because the queue manager does not have an application environment name.

Directive
Any of these:
- PluginInclude
- PluginExclude
- PluginDefault Include/Exclude
- CacheLocalFile
- ServerInit
- MaxActiveThreads

When the directives listed above are used within the ApplEnvConfig directive, they apply to a particular application environment. When you specify them outside an ApplEnvConfig statement, they apply globally. When the server runs standalone, all statements within the ApplEnvConfig set are ignored. Global directives are not ignored when running standalone.

The three Plugin directives are described later in this section. Read about CacheLocalFile, ServerInit, and MaxActiveThreads directives in previous pages in this section. Using the CacheLocalFile, ServerInit or MaxActiveThreads directives within an ApplEnvConfig statement allows you to tailor what files get cached, which ServerInit Plugins are used (and what parameters are passed), and the number of active threads, respectively.

Read more about how to implement these directives in "Using configuration directives to tailor your application environments" on page 379.

Example
ApplEnvConfig WEBHTML {
  CacheLocalFile /www/powerco/*.html
  MaxActiveThreads 50
  PluginInclude /ics/api/bin/icsexit05.so
  ServerInit /usr/lpp/internet/bin/mvsds.so:mvsdsInit /u/webserv/config/mymvsds.conf
}

Initial configuration file setting
None.
System management directives

**PluginDefault - Specify default Plugin action**

Use this directive to include or exclude Plugins by default. User Plugins specified on GWAPI directives will not be invoked unless loaded during initialization. The system default is to load all referenced user Plugins. The directive format is:

```
PluginDefault Include|Exclude
```

**Include**

Sets the default action to include user Plugins that have not been specifically referenced with a PluginInclude or a PluginExclude directive.

**Exclude**

Sets the default action to exclude user Plugins that have not been specifically referenced with a PluginInclude or PluginExclude directive.

**Example**

```
PluginDefault Exclude
```

**Initial configuration file setting**

None.

**PluginExclude - Specify Plugin not to load during initialization**

Use this directive to indicate that the specified user Plugin should not be loaded during initialization. Plugins specified on GWAPI directives will not be invoked effect unless loaded during initialization. The system default is to load all referenced user Plugins. The directive format is:

```
PluginExclude /path/file
```

```
/path/file
```

The fully qualified file name of your compiled DLL, including the extension.

**Example**

```
PluginExclude /ics/api/bin/icsexit05.so
```

**Initial configuration file setting**

None.

**PluginInclude - Specify Plugin to be loaded during initialization**

Use this directive to indicate that the specified user Plugin should be loaded during initialization. Plugins specified on GWAPI directives will not be invoked unless loaded during initialization. The system default is to load all referenced user Plugins. The format of the directive is:

```
PluginInclude /path/file
```

```
/path/file
```

The fully qualified file name of your compiled DLL, including the extension.

**Example**

```
PluginInclude /ics/api/bin/icsexit05.so
```

**Initial configuration file setting**

None.
Using configuration directives to tailor your application environments

PluginInclude, PluginExclude, and PluginDefault directives, in conjunction with the ApplEnvConfig directive, let you tailor which user DLLs are loaded in each of the Web server environments running WLM mode. Save storage by avoiding loading user DLLs that will not be invoked. Also, disable user Plugins by not loading the DLLs that contain them.

To learn more about these directives, refer to “ApplEnvConfig - Specify a set of directives to tailor the application environment” on page 376. To tailor your application environments for maximum performance, reevaluate your existing GWAPI Plugins and related directives.

Implicit actions of PluginInclude and PluginExclude directives: Be aware that PluginInclude and PluginExclude directives can generate implicit actions beyond what you may intend. Consider the following example:

```
ApplEnvConfig WEBHTML {
  PluginInclude /path/plugin.so
}
```

Besides causing DLL /path/plugin.so to be loaded when application environment WEBHTML is being configured, this specification also causes /path/plugin.so to not be loaded in the other web server application environments (queue servers) or the queue manager, assuming there are no other PluginInclude directives for /path/plugin.so.

If a DLL is needed in only one application environment, only one PluginInclude is needed within an ApplEnvConfig directive. This ensures that the DLL is loaded only for that application environment, and not loaded unnecessarily in other places.

Directives explicitly coded take precedence over implicit actions. If conflicting implicit specifications occur, the DLL is not loaded.

Order of precedence of ApplEnvConfig directives: A definite precedence order is determined by the “match” of the directive. When configuring a queue server, a directive within an ApplEnvConfig directive for the current application environment takes precedence over a match on a %%QS%%, which takes precedence over directives that are outside an ApplEnvConfig. Any explicit directives take precedence over implicit actions. The physical order matters only with multiple directives at the same level, and in some cases may even cause a syntax error.

The example below causes the application environment WEBHTML to have a MaxActiveThreads of value of 40, while all other queue servers would use 50. The queue manager would use 20. All queue servers would cache /other/*.html while WEBHTML would also cache the local files /path/*.html. The queue manager would not cache any. Both WEBHTML and the queue manager would load /mypath/mydll.so. All other queue servers would not load /mypath/mydll.so due to the generated implicit action. WEBHTML would also call the ServerInit function mvsdsInit passing "u/WEBSRV/config/mymvsds.conf".

```
ApplEnvConfig WEBHTML {
  MaxActiveThreads 40
  CacheLocalFile /path/*.html
  PluginInclude /mypath/mydll.so
  ServerInit /usr/lpp/internet/bin/mvsds.so:mvsdsInit u/WEBSRV/config/mymvsds.conf
}
```
The example below causes all user DLLs to not be loaded for the queue manager. This specification is appropriate if no requests are processed by the queue manager (for example, if all requests are routed to the application environments):

```
ApplEnvConfig %QM%% {
    PluginDefault Exclude
}
```

To tailor your application environment:

1. Evaluate your DLLs to see which are used in particular application environments. Identify DLLs to include and exclude for each different environment.

   Decide which directives to use within an ApplEnvConfig directive. If a DLL is needed only in one application environment, use the PluginInclude directive for that DLL within an ApplEnvConfig statement for that application environment. This will also implicitly cause that DLL to not be loaded in other environments.

   **PluginInclude Example**
   ```
   ApplEnvConfig MYAE {
       PluginInclude /mypath/mydll.so
   }
   ```

   This example would cause the MYAE application environment to load /mypath/mydll.so and the other environments to not load it. Use the directives this way if you only want the DLL loaded in one application environment.

   Likewise, use PluginExclude if a DLL should be loaded everywhere except in a particular application environment.

   **PluginExclude Example**
   ```
   ApplEnvConfig NOTMYAE {
       PluginExclude /mypath/mydll.so
   }
   ```

   This example would cause the NOTMYAE application environment to not load /mypath/mydll.so and the other environments to load it. Use the directives in this way if you want the DLL in all but the one application environment.

2. Use the Plugin Default Include/Exclude directive to specify default actions for an application environment.

   **PluginDefault Example**
   ```
   ApplEnvConfig %QM%% {
       PluginDefault Exclude
       PluginInclude /mypath/mydll.so
   }
   ```

   This example would cause the Queue Manager to load /mypath/mydll.so and no other DLL's. Be careful when using PluginDefault Exclude, you could possibly exclude DLL's that you may want.
3. Use the CacheLocalFile directive within the ApplEnvConfig directive to cache additional files for particular application environments.

**CacheLocalFile Example**

```plaintext
CacheLocalFile /all/*/html
ApplEnvConfig WEBAE1 {
  CacheLocalFile /AE1ONLY/*.html
}
```

This example would cause the WEBAE1 to cache the files in /AE1ONLY/*.html in addition to the files in /all/*.html.

4. Use the ServerInit directive within the ApplEnvConfig directive to vary the parameters with which a user Plugin is invoked.

**ServerInit Example**

```plaintext
ApplEnvConfig WEBAE1 {
  ServerInit /usr/lpp/internet/bin/mvsds.so:mvsdsInit /u/config/mvsds1.conf
}
ApplEnvConfig WEBAE2 {
  ServerInit /usr/lpp/internet/bin/mvsds.so:mvsdsInit /u/config/mvsds2.conf
}
```

This example shows how you can have different application environments invoke a ServerInit plugin with different parameters.

5. Use the MaxActiveThreads directive within ApplEnvConfig to set a maximum number of threads to have active for a particular application environment.

**SNMP - Enable and disable SNMP support**

Use this directive to enable or disable SNMP support. The form of the directive is:

```plaintext
SNMP setting
```

The `setting` can have a value of on or off.

**on** SNMP support is turned on.

**off** SNMP support is turned off.

**Example**

```plaintext
SNMP on
```

**Initial configuration file setting**

```plaintext
SNMP off
```

**Program default setting**

```plaintext
SNMP off
```

**SNMPCommunityName - Provide a security password for SNMP**

Use this directive to define the password used between the HTTP Server DPI subagent and the SNMP agent.

The form of the directive is:

```plaintext
SNMPCommunityName com_nam
```

**Example**

```plaintext
SNMPCommunityName public
```
System management directives

Initial configuration file setting
SNMPCommunityName public

Program default setting
SNMPCommunityName public

WebMasterEmail - Create an e-mail address to receive SNMP problem reports
Use this directive to create an e-mail address to receive SNMP problem reports. The default mail address is webmaster.

The form of the directive is:
WebMasterEmail webmastermailaddress

Example
WebMasterEmail webmaster@computer.com

Initial configuration file setting
WebMasterEmail webmaster

Timeouts - Close connections automatically
Use the directives described in this section to control the amount of time the server spends processing requests. If you are using persistent connections, see "PersistTimeout - Specify time to wait for the client to send another request” on page 391.

InputTimeout - Specify time allowed for the client to send a request
Use this directive to set the time allowed for a client to send a request after making a connection to the server. A client first connects to the server and then sends a request. If the client does not send a request within the amount of time on this directive, the server drops the connection.

The format of this directive is:
InputTimeout time

Specify time in any combination of hours, minutes (or mins), and seconds (or secs).

Examples
InputTimeout 1 hour
InputTimeout 3 mins 30 secs
InputTimeout 1 minute 30 seconds

Initial configuration file setting
InputTimeout 30 seconds

Program default setting
InputTimeout 30 seconds
OutputTimeout - Specify maximum time for sending output to the client

Use this directive to set the maximum time allowed for your server to send output to a client. The time limit applies to requests for local files and requests for which the server is acting as a proxy. The time limit does not apply to requests that start a local CGI program.

If the server does not send the complete response within the amount of time on this directive, the server drops the connection.

The format of this directive is:

```
OutputTimeout  time
```

Specify `time` in any combination of hours, minutes (or mins), and seconds (or secs).

**Examples**

```
OutputTimeout 1 hour
OutputTimeout 5 minutes 30 seconds
OutputTimeout 1 min 30 secs
```

**Initial configuration file setting**

```
OutputTimeout 2 minutes
```

**Program default setting**

```
OutputTimeout 2 minutes
```

ScriptTimeout - Specify time allowed for a program to complete

Use this directive to set the time allowed for a program started by the server to finish. When the time runs out, the server sends a message (SIGTERM signal) to the program. Five seconds later, the server sends a KILL signal.

The format of this directive is:

```
ScriptTimeout  time
```

Specify `time` in any combination of hours, minutes (or mins), and seconds (or secs).

**Examples**

```
ScriptTimeout 1 hour
ScriptTimeout 5 mins
ScriptTimeout 2 minutes 30 seconds
```

**Initial configuration file setting**

```
ScriptTimeout 2 minutes
```

**Program default setting**

```
ScriptTimeout 2 minutes
```

Tuning - Define performance and scalability settings

Use the directives described in this section to tune your Web server. For the most current performance hints and tips, see the Tuning section of the Web-based WebSphere Troubleshooter for OS/390.

To link to the Troubleshooter, go to URL:
To enable and customize caching using the Fast Response Cache Accelerator, see "EnableFRCA – Turn dynamic caching on or off" on page 383. For more information on the Cache Accelerator, see "Chapter 9. Customizing cache management with the Fast Response Cache Accelerator" on page 69.

CacheLocalFile - Specify files you want to load in memory at start up

Use this directive to specify the names of files you want to load into the server’s memory each time you start the server. You can have multiple occurrences of this directive in the configuration file. Include a separate directive for each file you want to load into memory.

By keeping your most frequently requested files loaded in the server’s memory, you can improve your server’s response time for those files. For example, if you load your server’s welcome page into memory at startup, the server can handle requests for the page much more quickly than if it had to read the file from a disk. Keep in mind that for each file you load into memory, you are making that amount of memory unavailable for other uses.

Before responding to a request for a file that is stored in memory, the server checks to see if the file has changed since the server was started. If the file has changed, the server responds to the request with the updated file and deletes the old version from its memory. To load the new file into memory, you need to restart the server.

Notes:
1. You can use an asterisk (*) as a wildcard character on the file names.
2. File name matching is not recursive. Only files in the specified directory will be cached. No files in subdirectories are affected.

Example
To cache a specific file
CacheLocalFile /www/html/index.html

To cache all .html files in the powerco directory
CacheLocalFile /www/powerco/*.html

Default
The default configuration includes CacheLocalFile directives for the HTML and graphics files that make up the server’s Front Page.

CacheLocalMaxBytes - Specify maximum amount of memory to use for file caching

Use this directive to specify the maximum amount of memory you want to allow for file caching. You can specify the memory in kilobytes (K) or megabytes (M). You must still use the CacheLocalFile directive to indicate which files you want cached.

Note: CacheLocalMaxBytes can help limit your cache size when you are using the wildcard character to specify the files on the CacheLocalBytes directive.

Example
CacheLocalMaxBytes 5K
Default
CacheLocalMaxBytes 2M

**CacheLocalMaxFiles - Specify the maximum number of files for caching**

Use this directive to specify the maximum number of files you want to be cached at one time. You must still use the CacheLocalFile directive to indicate which files you want cached.

**Note:** CacheLocalMaxFiles can help limit your cache size when you are using the wildcard character to specify the files on the CacheLocalFile directive.

**Example**

```
CacheLocalMaxFiles 150
```

Default
CacheLocalMaxFiles 200

**EnableFRCA — Turn dynamic caching on or off**

Use this directive to enable dynamic caching using the Fast Response Cache Accelerator. To customize dynamic caching, use the following directives:

- FRCACacheSize
- FRCACacheEntries
- FRCACacheOnly or FRCANoCaching
- FRCAMaxFileSize
- FRCAAccessLog
- FRCASystemName
- FRCAVirtualHost
- FRCAWLMParms

For more information on the Cache Accelerator, see "Chapter 9. Customizing cache management with the Fast Response Cache Accelerator" on page 69.

**Example**

```
EnableFRCA On
```

Initial configuration file setting
EnableFRCA Off

Program default setting
EnableFRCA Off

**FRCAAccessLog — Specify a log file path and name for dynamic caching requests**

Use this directive if you want requests served by the Fast Response Cache Accelerator to be logged in a separate log file. If this directive is used, no other log entries will be written for requests served from the Cache Accelerator cache. If this directive is not used, requests that are served by the Cache Accelerator will be logged the same as other requests.

You may want to use this optional log if you are concerned about performance but still want access log information about requests served from the Cache Accelerator cache.

**Note:** The server will ignore this setting if the EnableFRCA directive is set off.
Tuning directives

Example
FRCAAccessLog /usr/lpp/internet/server_root/logs/frca-log

Initial configuration file setting
None

Program default setting
None

FRCACacheEntries — Specify the maximum number of files to be dynamically cached

Use this directive to specify the maximum number of individual files that will be cached by the Fast Response Cache Accelerator. Your OS/390 Communications Server support determines the maximum number of files that can be cached.

Note: The server will ignore this setting if the EnableFRCA directive is set off.

Example
FRCACacheEntries 500

Initial configuration file setting
FRCACacheEntries 1000

Program default setting
FRCACacheEntries 1000

FRCACacheOnly — Specify URLs to be dynamically cached

Use this directive to specify a set of URLs that will be considered for caching by the Fast Response Cache Accelerator. URLs that are not in this list will never be cached.

Notes:
1. The server will ignore this setting if the EnableFRCA directive is set off.
2. Use either this directive or the FRCANoCaching directive.

Example
FRCACacheOnly *.gif
FRCACacheOnly requests/*.html
FRCACacheOnly faqs/*.html

Initial configuration file setting
None

Program default setting
None

FRCACacheSize — Specify size of the dynamic cache

Use this directive to specify the size of the Fast Response Cache Accelerator cache by entering the number of 4K (4096-byte) blocks of memory you want to allocate. The maximum size is limited by the amount of available memory in the OS/390 Communications Server.

Note: The server will ignore this setting if the EnableFRCA directive is set off.
Example
FRCACacheSize 500

In this example, the amount of memory allocated for the Cache Accelerator cache is 2000K (500 x 4K).

Initial configuration file setting
FRCACacheSize 25000

Program default setting
FRCACacheSize 25000

FRCAMaxFileSize — Specify maximum file size for the dynamic cache

Use this directive to specify the maximum size of a single file that will be placed in the Fast Response Cache Accelerator cache. You can specify this value in bytes, kilobytes, or megabytes. There is no minimum size.

Note: When you specify the cache size, use the following guidelines:
- For bytes, enter an integer
- For megabytes, enter an integer followed by M
- For kilobytes, enter an integer followed by K

Your OS/390 Communications Server support determines the maximum size of files that can be cached.

Note: The server will ignore this setting if the EnableFRCA directive is set off.

Example
FRCAMaxFileSize 2M

Initial configuration file setting
FRCAMaxFileSize 1M

Program default setting
FRCAMaxFileSize 32767 bytes or 4K or 1M

FRCANoCaching — Exclude URLs from the dynamic cache

Use this directive to specify a set of URLs which must never be cached by the Fast Response Cache Accelerator. URLs that are not specified in this list will be considered for caching.

Notes:
1. The server will ignore this setting if the EnableFRCA directive is set off.
2. Use either this directive or the FRCACacheOnly directive.

Example
FRCANoCaching /usergroup/*.html
FRCANoCaching /Docs/*.[pdf]

Initial configuration file setting
None

Program default setting
None
**Tuning directives**

**FRCAStackName — Specify the TCP/IP stack that supports the dynamic cache**

This directive is needed only if you are using the OS/390 UNIX System Services common INET function with multiple TCP/IP stacks.

Specify the name of the OS/390 UNIX physical file system that supports the TCP/IP stack used by the Fast Response Cache Accelerator. This name must match the name on the SubFileSysType statement in the OS/390 UNIX BPXPRMxx parmlib member.

**Note:** The server will ignore this setting if the EnableFRCA directive is set off.

**Example**

```
FRCAStackName TCP34
```

**Initial configuration file setting**

```
FRCAStackName TCP34
```

**Program default setting**

None

**FRCAVirtualHost — Indicate to the dynamic cache whether multiple virtual hosts or IP addresses are being used**

Use this directive to indicate to the Fast Response Cache Accelerator cache whether the Web server is configured to use multiple virtual host names or IP addresses. The Cache Accelerator cache uses this setting to expand the cache search key structure to include the use of multiple host names or IP addresses.

Valid settings are:

**AUTO (default)**

The Web server determines automatically whether multiple virtual host names or IP addresses are being used.

**ON**

Indicates that the Web server is configured to use multiple virtual host names or IP addresses.

**OFF**

Indicates that the Web server is configured to use a single host name or IP address.

**Note:** The server will ignore this setting if the EnableFRCA directive is set off.

**Example**

```
FRCAVirtualHost off
```

**Initial configuration file setting**

None

**Program default setting**

```
FRCAVirtualHost auto
```
FRCAWLMParms — Specify parameters for Workload Management

Use this directive to specify the unique subsystem name, application environment name (AEName), and transaction class that will be used to classify the work performed by the Cache Accelerator under Workload Management (WLM).

The format of this directive is:

```
FRCAWLMParms subsystem_name [AEName][WLM_transaction_class ]
```

All parameters must be 8 characters or less.

The unique subsystem name is required; the AEName and transaction class are optional. If the AEName and transaction class are specified, the Cache Accelerator Web serving will be classified under WLM. If not specified, no classification will occur. For more information on WLM parameters, see “ApplEnv - Specify application environment for workload management” on page 373. For tuning hints and tips, see “Monitoring and managing the Cache Accelerator” on page 70.

Note: The server will ignore this setting if the EnableFRCA directive is set off.

Example

```
FRCAWLMParms FRCAHTTP WEBFRCA WEBFRCA
```

In this example, the unique subsystem name is FRCAHTTP, the AEName is WEBFRCA, and the transaction class is WEBFRCA.

Initial configuration file setting

```
FRCAWLMParms FRCAHTTP WEBFRCA WEBFRCA
```

Program default setting

None

ListenBacklog - Specify the number of listen backlog client connections for the server to carry

Use this directive to specify the number of listen backlog client connections you want the server to carry before sending connection refused messages to clients. This number depends upon the number of requests that your server can process in a few seconds and should not be set higher than the number the server can process before the clients timeout and abort the connection from their end. Requests involving secure transactions take longer, for instance, than client requests that do not require users to logon and give a password.

Note: If the ListenBacklog value is greater than the SOMAXCONN value supported by TCP/IP, SOMAXCONN will be used instead.

Example

```
ListenBacklog 100
```

Initial configuration file setting

```
ListenBacklog 128
```
Tuning directives

**LiveLocalCache - Specify whether the cache is updated when a cached file is modified**

Use this directive to specify whether or not the cache is updated when a cached file is modified. Specify ON if you want users requesting a cached file to get the file with the latest updates. OFF is the high performance setting.

*Initial configuration file setting*

LiveLocalCache off

*Program default setting*

LiveLocalCache off

**MaxActiveThreads - Specify the maximum number of threads to have active**

Use this directive to set the maximum number of threads that you want to have active at one time. If the maximum is reached, the server holds new requests until another request finishes and threads become available.

**Hints and tips:**

- The setting of MaxActiveThreads must be lower than the MAXTHREADTASKS setting in your BPXPRMxx member that was used for OS/390 UNIX.
- As a general rule, we recommend setting MaxActiveThreads to a value no greater than 150. However, this is dependent on your system environment. Experience has shown that values greater than 200 can cause storage shortages. The basic recommendation is to lower the value until you do not get any storage problems. However, if problems persist, this needs to be investigated regardless of the value you set.

*Example*

MaxActiveThreads 80

*Program default setting*

MaxActiveThreads 40

*Initial configuration file setting*

MaxActiveThreads 40

**MaxPersistRequest - Specify the maximum number of requests to receive on a persistent connection**

Use this directive to specify the maximum number of requests the server will allow on a persistent connection. When determining this number, be sure to consider the number of images used in your pages. Each image requires a separate request.

The format of this directive is:

MaxPersistRequest number

*number* is the number of requests the server will allow for a persistent connection.

*Initial configuration file setting*

MaxPersistRequest 5

*Program default setting*

MaxPersistRequest 10
PersistTimeout - Specify time to wait for the client to send another request

Use this directive to specify the amount of time the server should wait between client requests before cancelling a persistent connection.

The server uses a different timeout, the input timeout, to determine how long to wait for the client to send the first request after the connection is established. For more information on the input timeout, see “InputTimeout - Specify time allowed for the client to send a request” on page 382.

After the server sends its first response, it uses the persistent timeout to determine how long it should wait for each subsequent request before cancelling the persistent connection.

The format of this directive is:
PersistTimeout time

time can be any valid time increment, but usually will be seconds or minutes.

Examples
PersistTimeout 1 hour
PersistTimeout 2 mins 30 secs
PersistTimeout 15 seconds

Initial configuration file setting:
PersistTimeout 5 seconds

Program default setting:
PersistTimeout 5 seconds

ServerPriority - Specify the priority you want your server to have on your system

Use this directive to override the OS/390 UNIX default priority for scheduling the server process.

Valid values are:
- -20 - maximum OS/390 UNIX priority
- 0 - default OS/390 UNIX priority
- +19 - minimum OS/390 UNIX priority

Example
ServerPriority -20

Initial configuration file setting
ServerPriority -10

UseACLs - Specify whether ACL files will be checked

Use this directive to specify whether the ACL files will be checked for file protection. Set this directive to NEVER or PROTECTONLY for better server performance. The format of the directive is:

UseACLs setting

The setting can have a value of always, protect only, or never.
Tuning directives

always
The server will always look for an ACL file on every file request.

protectonly
The server will only look for an ACL file when the file request is for a file that is covered by a protection statement.

never
The server will never look for an ACL file on a file request.

Example
UseACLs protectonly

Initial configuration file setting
UseACLs protectonly

UseMetaFiles - Specify whether meta files will be used
Use this directive to specify whether the meta files used by the server. Set this directive to off for better server performance. The format of the directive is:
UseMetaFiles setting

The setting can have a value of on or off.
on The server will always use meta files.
off The server will not use meta files.

Example
UseMetaFiles off

Initial configuration file setting
UseMetaFiles off
Appendix D. Data set naming reference

This appendix explains how to name OS/390 data sets when you use the GWAPI MVSDS DLL Service. For more information on this service, see “Appendix F. GWAPI MVSDS DLL Service” on page 405.

Using certain qualifiers in data set names can affect how the HTTP Server determines the data set’s content type. The following table shows two examples:

<table>
<thead>
<tr>
<th>QUALIFIER</th>
<th>CONTENT TYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTML</td>
<td>HTML document</td>
</tr>
<tr>
<td>GIF</td>
<td>GIF image file</td>
</tr>
</tbody>
</table>

An HTML document should be stored in a data set whose fully-qualified name contains the qualifier HTML, for example:

    WEBRSRV.HTML.MYPAGE

Additional qualifiers can be used for language support or other purposes. The following example uses ENU as the qualifier:

    WEBRSRV.PAGES.HTML.ENU(MYPAGE)

In this example, the PAGES qualifier is included, but probably will not be recognized by the HTTP Server as indicating the content type. Unrecognized qualifiers are ignored by the HTTP Server. The ENU qualifier resolves to a U.S. English language document, and the HTML qualifier indicates an HTML document.
Appendix E. Environment variables

Overview

This appendix includes variables that can be used in GWAPI and CGI programs:

- All variables can be used in GWAPI programs.
- The following variables can be used in CGI programs:

<table>
<thead>
<tr>
<th>Variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUTH_TYPE</td>
</tr>
<tr>
<td>CONTENT_ENCODING</td>
</tr>
<tr>
<td>CONTENT_LENGTH</td>
</tr>
<tr>
<td>CONTENT_TYPE</td>
</tr>
<tr>
<td>GATEWAY_INTERFACE</td>
</tr>
<tr>
<td>HTTP_ACCEPT</td>
</tr>
<tr>
<td>HTTP_COOKIE</td>
</tr>
<tr>
<td>HTTP_USER_AGENT</td>
</tr>
<tr>
<td>PATH_INFO</td>
</tr>
<tr>
<td>SERVER_SOFTWARE</td>
</tr>
<tr>
<td>QUERY_STRING</td>
</tr>
<tr>
<td>REFERER_URL</td>
</tr>
<tr>
<td>REMOTE_ADDR</td>
</tr>
<tr>
<td>REMOTE_USER</td>
</tr>
<tr>
<td>REQUEST_METHOD</td>
</tr>
<tr>
<td>SERVER_NAME</td>
</tr>
<tr>
<td>SERVER_PORT</td>
</tr>
<tr>
<td>SERVER_PROTOCOL</td>
</tr>
</tbody>
</table>

You can use the following SSL-related environment variables in CGI programs:

<table>
<thead>
<tr>
<th>Variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTTPS</td>
</tr>
<tr>
<td>HTTPS_CLIENT_CERT_COUNTRY</td>
</tr>
<tr>
<td>HTTPS_CLIENT_CERT_ISSUER_COMMON_NAME</td>
</tr>
<tr>
<td>HTTPS_CLIENT_CERT_ISSUER_COUNTRY</td>
</tr>
<tr>
<td>HTTPS_CLIENT_CERT_ISSUER_LOCALITY</td>
</tr>
<tr>
<td>HTTPS_CLIENT_CERT_ISSUER_STATE_OR_PROVINCE</td>
</tr>
<tr>
<td>HTTPS_CLIENT_CERT_LEN</td>
</tr>
<tr>
<td>HTTPS_CLIENT_CERT_LOCALITY</td>
</tr>
<tr>
<td>HTTPS_CLIENT_CERT_ORGANIZATIO</td>
</tr>
<tr>
<td>HTTPS_CLIENT_CERT_ORG_UNIT</td>
</tr>
<tr>
<td>HTTPS_CLIENT_CERT_STATE_OR_PROVINCE</td>
</tr>
<tr>
<td>HTTPS_SESSION_ID_NEW</td>
</tr>
</tbody>
</table>

Variables with values that are read-only

The following variables contain values that can be extracted from a client request (read-only variables).

**GWAPI Note:** A return of HTTPD_READ_ONLY will result when attempting to change these variables using HTTPD_set.

**ALL VARIABLES**

All the CGI environment variables, for example:

<table>
<thead>
<tr>
<th>Variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACCEPT_RANGES</td>
</tr>
<tr>
<td>AUTH_STRING</td>
</tr>
<tr>
<td>CLIENT_ADDR 9.67.84.3</td>
</tr>
</tbody>
</table>

**AUTH_STRING**

The credentials sent with the request header AUTHORIZATION information
AUTH_TYPE
If the server supports client authentication and the script is protected, this environment variable contains the method used to authenticate the client, for example:
Basic

_BPX_USERID
The server uses the authority of the access control user ID to process a request.

CLIENT_AUTH
State of SSL Client Authentication, for example, On or Off

DOCUMENT_NAME
The name of the document or script on the server

DOCUMENT_ROOT
The path to the document or script as defined by Pass rules

DOCUMENT_URL
The Uniform Request Locator, for example:
ACCEPT_RANGES ON
AUTH_STRING
CLIENT_ADDR 9.67.184.3

FSCP
Returns the value specified on the DefaultFsCp configuration directive

Note: The value for FSCP can only be extracted; you cannot change the value with an HTTPD_set.

GATEWAY_INTERFACE
Contains the version of CGI or GWAPI that the server is using, for example: CGI/1.1 or GWAPI/1.2

HTTP_AUTHORIZATION
Authorization header information

HTTP_PROXY_AUTHORIZATION
Proxy Authorization header information

HTTPS
This is valid only if security is active and a valid client certificate is available.

HTTPS_KEYSIZE
The value of security key size
[128]

HTTPS_CLIENT_CERT
The client certificate body

HTTPS_CLIENT_CERT_COMMON_NAME
From the certificate body

HTTPS_CLIENT_CERT_COUNTRY
From the certificate body

HTTPS_CLIENT_CERT_LOCALITY
From the certificate body

HTTPS_CLIENT_CERT_ORGANIZATION
From the certificate body
HTTPS_CLIENT_CERT_ORG_UNIT
From the certificate body

HTTPS_CLIENT_STATE_OR_PROVINCE
From the certificate body

HTTPS_CLIENT_CERT_LEN
Length of client certificate body

HTTPS_CLIENT_CERT_SERIAL_NUM
The serial number from the certificate body

HTTPS_CLIENT_CERT_ISSUER_COMMON_NAME
From the certificate body

HTTPS_CLIENT_CERT_ISSUER_COUNTRY
From the certificate body

HTTPS_CLIENT_CERT_ISSUER_LOCALITY
From the certificate body

HTTPS_CLIENT_CERT_ISSUER_ORGANIZATION
From the certificate body

HTTPS_CLIENT_CERT_ISSUER_ORG_UNIT
From the certificate body

HTTPS_CLIENT_CERT_ISSUER_STATE_OR_PROVINCE
From the certificate body.

HTTPS_SESSION_ID
The current session ID

HTTPS_SESSION_ID_NEW
This environment variable indicates that the Secure Sockets Layer (SSL) connection renegotiated new session keys and is not using any session information from a previous request. Possible values are 1 or 0.

NETCP
Returns the value specified on the DefaultNetCp configuration directive.

Note: The value for NETCP can only be extracted; you cannot change the value with an HTTPD_set.

PICS_LABEL
PICS label information

PICS_PATHNAME
Path and file name of the document to be rated

PICS_SERVICENAME
Name of the PICS rating service, for example:
Ratings USA

PICS_SITENAME
Example:
http://www.ratings.com/

REQUEST_METHOD
Contains the method (as specified with the METHOD attribute in an HTML form) used to send the request, for example:
GET or POST

Appendix E. Environment variables 397
REQHDR
Defines a list of the headers sent by the client

REFERER_URL
The last URL location of the browser, for example:
http://www.acme.com/homepage

SDH Socket descriptor for the request

SERVER_ADDR
Local IP address of the server

SERVER_CFG_PORT
Contains the port number the server is accepting http requests on, for
example, 80; from the PORT configuration directive or the command line
option -p.

SERVER_CFG_SSLPORT
Contains the port number the server is accepting SSL requests on, for
example, 443; from the PORT configuration directive or the command line
option -sslport.

SERVER_LEVEL
The release level of the currently running server

SERVER_NAME
Contains the server host name or IP address of the server, for example:
www.ibm.com

SERVER_PORT
Contains the port number of the server to which the client request was
sent, for example:
80

SERVER_PROTOCOL
Contains the name and version of the protocol used to make the request,
for example:
HTTP/1.1

SERVER_ROOT
Directory where the server program is installed

SERVER_SOFTWARE
Contains the name and version of the server, for example:
HTTP Server/Version 5.2

SERVER_STATE
Current process step for the request (for example, "PREEXIT")

URI Same as DOCUMENT_URI

URL Same as DOCUMENT_URI

WQ_APPLENV
Workload Management application environment assigned by the -AE
option

WQ_STATE
Workload Management work queue state, for example, HTTPD,
WQ_Daemon, and WQ_Server.

WQ_SUBSYS
Workload Management subsystem name assigned by the -SN option.
Variables with values that can be set or created

The following variables contain values you can set or create when processing a client request.

**ACCEPT RANGES**
Used to accept ranges other than bytes

**CACHE_HIT**
Returns "1" if a document is found in cache, returns "0" otherwise.

**CACHE_UPDATE**
Returns "1" if cache information was updated successfully, returns "0" otherwise.

**CGI_variable**
Returns the value of the user-defined CGI variable, where variable is user-defined from an HTTPD_set() command (for example, CGI_MYVARIABLE). This CGI_variable is appended to the list of environment variables sent to a CGI program.

**CLIENT_ADDR**
IP address of the client, for example: 9.67.193.2

**CLIENT_HOST**
Same as REMOTE_ADDR

**CLIENTMETHOD**
HTTP method used in the request

**CLIENT_NAME**
Host name of the machine making the request, for example: joeblow

**CLIENT_PROTOCOL**
Name and version of the protocol the client is using to make the request, for example: HTTP

**CONNECTIONS**
Number of active connections the server has open, for example: 15

**CONTENT_CHARSET**
Character set of the response for text/*, for example: US ENGLISH

**CONTENT_TYPE_PARAMETERS**
Other MIME attributes but not the character set

**CONVERT_REQUEST_BODY**
Only for OS/390 EBCDIC issues; specifies if the ASCII to EBCDIC conversion is done on the request body. Available settings are YES, NO, and AUTO. The default is NO. If AUTO is set, the server tries to make the decision in the same way CGI requests are converted. For example: AUTO

**CONTENT_ENCODING**
Specifies the encoding used in the document. For example: x-gzip
CONTENT_LENGTH
When information is sent with the method of POST, this variable contains
the number of characters of data. Servers typically do not send an
do-end-of-file flag when they forward the information using stdin. If needed,
you can use the CONTENT_LENGTH value to determine the end of the
input string. For example:
7034

CONTENT_TYPE
When information is sent with the method of POST, this variable contains
the type of data included. You can create your own content type in the
server configuration file and map it to a viewer. For example:
application/x-www-form-urlencoded

ERRORINFO
Specifies the error code to determine the error page, for example:
401

EXPIRES
Defines the expiration for documents stored in a proxy’s cache

HTSERVLETCLASSNAME
Server-side includes servlet class name.

HTSERVLETCODEBASE
The URL location of the server-side includes servlet code.

HTSERVLETINITPARAMS
Initial value pairs for the server-side includes servlet.

HTSERVLETNAME
Server-side includes servlet name.

HTSERVLETREGPARAMS
Server-side includes parameter value pairs.

HTTP_ACCEPT
Contains the list of MIME types the browser accepts. For example:
text/html

HTTP_ACCEPT_RANGES
This is the same as ACCEPT_RANGES.

HTTP_CONTENT_CHARSET
This is the same as CONTENT_CHARSET.

HTTP_CONTENT_ENCODING
This is the same as CONTENT_ENCODING.

HTTP_CONTENT_LENGTH
This is the same as CONTENT_LENGTH.

HTTP_CONTENT_TYPE
This is the same as CONTENT_TYPE.

HTTP_COOKIE
Contains the cookie sent with this request; used to communicate state
information. For example:
CustomerNumber=HJ68944

You can find details about cookies at
HTTP_EXPIRES
   This is the same as EXPIRES.

HTTP_LAST_MODIFIED
   This is the same as LAST_MODIFIED.

HTTP_REASON
   Sets the reason string in the HTTP response header

HTTP_RESPONSE
   Sets the response code in the HTTP response header

HTTP_USER_AGENT
   Contains the name of your Web browser. For example:
   Netscape Navigator / V4.0

INIT_STRING
   The string specified on the ServerInit directive

LAST_MODIFIED
   Specifies the date of the last update, for example:
   12/25/96

LDAP:variable
   This reads the LDAP variable specified.
   LDAP:CURRENT:HOST

LOCAL_VARIABLES
   All the user-defined variables

PASSWORD
   For Basic authentication, contains the decoded password, for example:
   password

PATH
   Fully translated path

PATH_INFO
   Contains the additional path information as sent by the Web browser. For example:
   /ballyhoo

PATH_TRANSLATED
   Contains the decoded or translated version of the path information contained in PATH_INFO. For example:
   d:/wwwhome/ballyhoo
   /wwwhome/ballyhoo

PEAKCONNECTIONS
   Defines peak number of connections the server allows, for example:
   45

PPATH
   Partially translated path

PROXY_ACCESS
   Defines whether the request is a proxy request, for example:
   NO

PROXY_CONTENT_LENGTH
   Content-Length header of the proxy request made through HTTPD_proxy. When information is sent with the method of POST, this variable contains the number of characters of data. Servers typically do not send an
end-of-file flag when they forward the information using stdin. If needed, you can use the CONTENT_LENGTH value to determine the end of the input string. For example:

7034

**PROXY_CONTENT_TYPE**
Content-Type header of the proxy request made through HTTPD_proxy. When information is sent with the method of POST, this variable contains the type of data included. You can create your own content type in the server configuration file and map it to a viewer. For example:
application/x-www-form-urlencoded

**PROXY_METHOD**
Method for the request made through HTTPD_proxy

**QUERY_STRING**
When information is sent using a method of GET, this variable contains the information in a query that follows the ?. This information must be decoded by the CGI program. For example:
NAME=Eugene+T%2E+Fox&ADDR=etfox%7Cibm.net&INTEREST=xyz

**REMOTE_ADDR**
Contains the IP address of the Web browser, if available. For example:
9.23.06.8

**REMOTE_HOST**
Contains the host name of the Web browser, if available. For example:
www.raleigh.ibm.com

**REMOTE_USER**
If the server supports client authentication and the script is protected, this environment variable contains the username passed for authentication. For example:
joeuser

**REQUEST_CONTENT_LENGTH**
When information is sent with the method of POST, this variable contains the number of characters of data. Servers typically do not send an end-of-file flag when they forward the information using stdin. If needed, you can use the CONTENT_LENGTH value to determine the end of the input string. For example:
7034

**REQUEST_CONTENT_TYPE**
The content type of the request body, for example:
integer

**RESPONSE_CONTENT_LENGTH**
When information is sent with the method of POST, this variable contains the number of characters of data. Servers typically do not send an end-of-file flag when they forward the information using stdin. If needed, you can use the CONTENT_LENGTH value to determine the end of the input string. For example:
7034

**RESPONSE_CONTENT_TYPE**
When information is sent with the method of POST, this variable contains the type of data included. You can create your own content type in the server configuration file and map it to a viewer. For example:
SCRIPT_NAME
URL of the request

SSI_variable
Reads the user-defined server-side includes variable. variable is the user-defined name.

SSLMODE
Whether Secure Sockets Layer (SSL) is enabled. This variable is read only. If you attempt to modify it using HTTPD_set, the server will return an HTTPD_READ_ONLY error. For example:
ON

SSL_X500_HOST
If the server is using strong authentication, this environment variable specifies the host name or directory used to retrieve certificates and certificate revocation lists (CRLs) for certificate validation. For example:
myhost or 97.55.4.213

SSL_X500_PORT
If the server is using strong authentication, this environment variable specifies the port number for the TCP/IP connection to the Directory Server. For example:
22165

TOTALKBYTES
Total number of kilobytes the HTTP Server has served, for example:
1000

TOTALTRANSACTIONS
Total number of transactions the HTTP Server has served. For example:
300

USERNAME
Same as REMOTE_USER

Server-Side Include variables
Use NCSA tags to print the value of a server-side include variable on a Web page. For example, to print the name of the server, use the following NCSA tag:
<!--#echo var=SERVER_NAME -->
The Web server supports the following server-side include variables. All variables are read-only except SSI_GLOBAL_VARS.

SERVER_ADDR
Local IP address of the server

SERVER_NAME
The server host name or IP address of the server, for example:
www.ibm.com

SERVER_PORT
The port number of the server to which the client request was sent, for example:
80
SERVER_PROTOCOL
The name and version of the protocol used to make the request, for example:
HTTP/1.1

SERVER_ROOT
Directory where the server program is installed

SERVER_SOFTWARE
Contains the name and version of the server, for example:
HTTP Server/1.0

SSI_GLOBAL_VARS
All the SSI global variables

SSI_DIR
The path of the current file relative to SSI_ROOT. If the current file is in SSI_ROOT, this value is “/”. This variable is read only. If you attempt to modify it using HTTPD_set, the server will return an HTTPD_READ_ONLY error.

SSI_FILE
The file name of the current file. This variable is read only. If you attempt to modify it using HTTPD_set, the server will return an HTTPD_READ_ONLY error.

SSI_INCLUDE
The value used in the include command that retrieved this file. This value is not defined for the topmost file. This variable is read only. If you attempt to modify it using HTTPD_set, the server will return an HTTPD_READ_ONLY error.

SSI_PARENT
The path and file name of the includer, relative to SSI_ROOT. This variable is read only. If you attempt to modify it using HTTPD_set, the server will return an HTTPD_READ_ONLY error.

SSI_ROOT
The path of the topmost file. All include requests must be in this directory or a child of this directory. This variable is read only. If you attempt to modify it using HTTPD_set, the server will return an HTTPD_READ_ONLY error.

Example:
<!--#echo var=SSI_DIR -->

TRACE
Defines whether server side tracing is ON or OFF.
This GWAPI service allows you to access Web contents stored in OS/390 data sets. Data sets to be preloaded are specified in a separate configuration file which is specific to the MVSDS DLL. The default MVSDS DLL configuration file is /etc/mvsds.conf. Preloading of OS/390 data sets is suggested for frequently accessed Web content. Note that MVS data sets can be accessed without preloading them.

Preparing the MVSDS DLL Configuration File

The default MVSDS DLL configuration file is /etc/mvsds.conf.

You can specify an alternate location and file name by using the MVSDS_CFG LE environment variable. Note that this method for specifying an alternate configuration file has precedence over the program default. To set environment variables, see "IMWHTTPD program" on page 253 and the LE Programming Guide.

You can also specify the MVSDS DLL configuration file on the ServerInit directive in the server configuration file.

If no configuration file is found, no initial configuration of the MVSDS DLL occurs and no MVS data sets will be preloaded at server initialization.

Specifying the MVSDS DLL configuration directives

The MVSDS DLL configuration file (by default, /etc/mvsds.conf) contains LOAD directives, one per line, in the following format:

```
LOAD datasetname
```

**datasetname**

Specifies the OS/390 data set name to be preloaded. A *datasetname* can be:

- A JCL DD name (DDN), optionally followed by a PDS member name. If a PDS member name is included, the DDN must resolve to a PDS.
- A fully-qualified name, optionally followed by a PDS member name, and enclosed in single quotes.
- A partially-qualified name, optionally followed by a PDS member name to which the MVS user ID (for example, WEBSRV) gets prepended as a high-level qualifier (no quotes). Note that the MVS user ID used in any fully-qualified name is the user ID the server is running under.

The format for the *datasetname* is:

```
[JCL DD|fully-qualified name|partially-qualified name] [(member)]
```
**MVSDS DLL Service**

*name*

Specifies a sequence of one or more qualifiers, each containing eight or less characters, separated by periods.

*member*

Specifies a member name of no more than eight characters.

**Note:** If you are specifying a fully-qualified name, single quotes are required around the entire *name* and *member* string.

**Examples**

The following shows LOAD directive examples:

```plaintext
LOAD 'WEBsrv.HTML.ENU.PAGES'
LOAD IMAGES.GIF.TREES
LOAD 'WEBsrv.WAV.SOUNDS'
LOAD DD:SALES(JANUARY)
```

**Note:** Whenever a PDS is specified in a LOAD directive, the PDS directory and all of the members are preloaded.

Configuration of the MVSDS DLL preload data sets is also supported through the Administration and Configuration forms.

**Enable MVSDS DLL preloading of OS/390 data sets using GWAPI directives in the server configuration file**

To instruct the server to preload OS/390 data sets defined in the MVSDS DLL configuration file, you must implement three of the GWAPI directives in the server configuration file (by default, `/etc/httpd.conf`). `ServerInit` and `ServerTerm` are optional directives for preloading data sets. However, if a `ServerInit` is specified, a `ServerTerm` must be specified. It is suggested that you use this mechanism to preload data sets which hold frequently accessed Web contents. The three GWAPI directives used are `ServerInit`, `Service`, and `ServerTerm`. The format for the `ServerInit` directive is:

```plaintext
ServerInit /path/file:function_name [INIT_STRING]
```

**Example**

To use the default location of `/etc/mvsds.conf`, enter:

```plaintext
ServerInit /usr/lpp/internet/bin/mvsds.so:mvsdsInit
```

The `INIT_STRING` parameter on the `ServerInit` directive can be used to specify an alternate MVSDS DLL initialization file, for example:

```plaintext
ServerInit /bin/mvsds.so:mvsdsInit /u/WEBsrv/config/mymvsds.conf
```

The value of the `INIT_STRING` can be extracted like any other server CGI variable. If no `INIT_STRING` is defined, an empty string is returned.

The `ServerTerm` directive is used to clean up the DLL during the server restart to avoid memory leaks. See "Appendix C. Configuration directives" on page 267 for information about the `ServerInit`, `Service`, and `ServerTerm` directives.

The format of the `ServerTerm` directive is:

```plaintext
ServerTerm /path/file:function_name
```

An example of this directive is:

```plaintext
ServerTerm /bin/mvsds.so:mvsdsTerm
```
Accessing Web contents from OS/390 data sets

The only directive you need if you are not preloading an MVS data set is the Service directive. You must have the Service directive specified in the server configuration file to activate the GWAPI MVSDS DLL Service. The format for this directive is:

```
Service request-template /path/file:function_name [IP_address_template]
```

Here are examples of the Service directive:

```
Service /MVSDS* /usr/lpp/internet/bin/mvsds.so:mvsdsGet*
Service /MYDATA* /usr/lpp/internet/bin/mvsds.so:mvsdsGet*
Service /SALES* /usr/lpp/internet/bin/mvsds.so:mvsdsGet*
```

Web contents stored in the OS/390 data sets can be accessed by using a URL in the following format:

```
http://hostname/request_template/datasetname
```

- **hostname**
  Specifies the domain name or IP address returned to the client.

- **request_template**
  Specifies to the server that the file specified by the datasetname following the request_template is an MVS data set, not an HFS file or directory. The request_template can be anything meaningful to you that does not conflict with or override other matching directives. This should correspond to the request_template specified on the Service directive.

The following are examples of URL’s containing data set names:

```
http://www.mvs105.tcp.ibm.com/MVSDS/WEBSRV.PAGES.HTML.HOME
http://www.mvs105.tcp.ibm.com/MVSDS/MYPAGES.IMAGES.GIF(NATURE)
```

Where MVSDS is the request_template. The first example specifies a fully-qualified name, the second example specifies a JCL DD reference to a PDS whose member OTHERPAG is to be retrieved, and the third specifies a partially-qualified name to which the MVS user ID will be prepended.

- **datasetname**
  Specifies the name of the MVS data set.

The datasetname specified in the URL can be one of the following:

- **‘THIS.THAT.THEOTHER’**
  Specifies, in quotes, a fully-qualified data set name. The specified data set is opened directly.

- **‘THIS.THAT.THEOTHER(MEMBER)’**
  Specifies, in quotes, a fully-qualified data set name represents a PDS member. The specified member is opened directly.

- **THIS.THAT.THEOTHER**
  Specifies a partially-qualified data set name to which a high level qualifier (hlq) must be prepended. In the OS/390 environment, the MVS user ID is prepended to the specified name to produce the fully-qualified name ‘USERID.THIRD.THEOTHER’, which is opened.

- **THIS.THAT.THEOTHER(MEMBER)**
  Specifies a partially-qualified PDS member. The hlq (MVS user ID) is prepended to produce the fully-qualified name ‘USERID.THIS.THAT.THEOTHER(MEMBER)’, which is opened.
MVSDS DLL Service

**DD:THISNAME**
Specifies that the actual data set or PDS member name is retrieved from the associated JCL DD statement and is then opened.

**DD:THISNAME(MEMBER)**
Specifies that the PDS name is retrieved from the associated JCL DD statement and the MEMBER specified is opened.

If a member name is not specified using the (MEMBER) form, the MVSDS DLL service determines whether the named data set is a PDS or a sequential data set. The contents of the data set get written back to the HTTP Server, depending on the type of data set:

- An HTML directory listing is returned for a PDS
- The data set attributes (content type, content encoding, language encoding, etc.) are evaluated and established in the server environment according to a predefined naming convention. The contents are then returned to the server “as is” for a sequential data set or a PDS member.

**Note:** MVSDS will not be able to serve any data set name that contains a pound sign (#) or a member name that contains a pound sign that is explicitly requested in a URL. This is because of the nature of http, where a pound sign is treated as a terminating character. However, a member list for a partitioned data set that is generated by MVSDS may contain member names that contain pound signs.
Appendix G. Key Management Utility

Before you begin .......................... 409
Review security configuration examples .... 409
Verify that you are authorized to issue the
ikeyman command .......................... 410
Set up your system environment .......... 410
Migrate existing key ring files ............ 410
Using the IKEYMAN command line utility . 411
Starting IKEYMAN .......................... 411
Using IKEYMAN ........................... 411
User interface task reference ............. 411
Creating a new key database ............. 412
Creating a new key pair and certificate
request .................................... 414
Creating a self-signed certificate ......... 414
Exporting keys ............................ 415
Importing keys ............................ 416
Listing and managing keys and certificates
Listing and managing request keys ....... 416
Listing trusted CAs ....................... 417
Opening a key database .................. 417
Receiving a certificate signed by a trusted
CA ........................................... 417
Showing the default key in a key database 418
Storing a CA’s certificate ................ 418
Storing the encrypted database password in a
stash file .................................. 418
Command reference ...................... 418

Before you begin

Review security configuration examples

This appendix provides detailed information on tasks you can perform using the
IBM Key Management Utility (IKEYMAN). It does not explain how to use the
HTTP Server Certificate Authority utility (HTTP Server CA) or how to configure
security options that require updates to the server configuration file.

Before you begin, we recommend that you review the following examples:

• “Setting up secure connections using an external CA” on page 61
  Use this example if you plan to have your operational server certificate signed
  by an external certificate authority (CA), such as VeriSign.

• “Setting up secure connections using self-signed certificates” on page 62
  Use this example if you plan to act as your own CA for a private Web network.
  This example shows you how to use IKEYMAN and HTTP Server CA to set up
  and administer your own certificate authority.

Both examples show you how to change your server security defaults using either
the Configuration and Administration Forms or by editing the configuration file
directives.

Migration note: Prior to Release 5.0, you could create keys and certificate requests
using the Configuration and Administration Forms. This option was removed in
Release 5.0 for security reasons. You now use IKEYMAN to create keys and
certificate requests. The Security Configuration and Administration Forms are now
used to perform the following tasks:

• Enable or disable SSL transactions; by default, SSL transactions are enabled.
• Change the server’s default SSL port of 443
• Specify the type of SSL client authentication to use; by default, client
  authentication is not used.
• Change the server’s default timeout duration for SSL sessions
• Change the server’s default encryption settings; by default, the server will use
  the strongest encryption level supported by both the client and the server.
• Register key databases you have added using IKEYMAN
Key Management Utility

- Specify another default key database for the server
- Associate a certificate with a specific IP address, if using multiple IP addresses on the server

Verify that you are authorized to issue the ikeyman command

By default, the ikeyman command is located in the /usr/lpp/internet/bin directory and has a permission bit setting of 750. By default, WEBADM is the owning user ID and IMWEB is the owning group. To execute this command, you need to log on as WEBADM, be part of the IMWEB group, or execute the command from an ID that has superuser authority.

Set up your system environment

To run IKEYMAN, set your system environment using the following guidelines:

- Add the path where IKEYMAN is installed to your PATH environment variable:
  
  /usr/lpp/internet/bin

- Add the path where the IKEYMAN DLL shared library (*.dll) files are installed to your LIBPATH environment variable:
  
  /usr/lpp/internet/bin

- Add the path where the IKEYMAN message catalog (*.cat) files are installed to your NLSPATH environment variable:
  
  /usr/lpp/internet/%L/%N

Migrate existing key ring files

You must convert your existing key ring files (*.kyr) to the key database format (*.kdb) before IKEYMAN can use your existing files.

To migrate a key ring file using the same filename, issue the following command on a command line:

```
ikeyman -m -r key_ring_filename.kyr
```

When you issue this command, a *.kdb file is created with the same name as your *.kyr file; your original *.kyr file is not overwritten.

To migrate a key ring file using a different filename, issue the following command on a command line:

```
ikeyman -m -k key_database_filename -r key_ring_filename.kyr
```

When you issue this command, a *.kdb file is created with the name you specify; your original *.kyr file is not overwritten.

**Note:** If you copied the old key ring file into a new HFS location, be sure to also copy the stash file.

After your key ring files are converted to the database format, use IKEYMAN to open and manage your databases.

If you need to convert a key database file (*.kdb) to a key ring file (*.kyr), issue the following command on a command line:

```
ikeyman -c -k key_database_filename.kdb
```

When you issue this command, a *.kyr file is created with the same name as your *.kdb file; your original *.kdb file is not changed.
To convert a key database file (*.kdb) to a key ring file (*.kyr) using a different filename, issue the following command on a command line:

```
ikeyman -c -r key_ring_filename -k key_database_filename.kdb
```

When you issue this command, a *.kyr file is created with the name you specify; your original *.kdb file is not changed.

### Using the IKEYMAN command line utility

#### Starting IKEYMAN

To start the IKEYMAN user interface, enter `ikeyman` on a command line.

**Note:** IKEYMAN allows you to enter the fully qualified path and file name when it prompts you for a key database, certificate request, or certificate file name. We recommend that you change to the path where the file should be stored before you start the utility.

When you start IKEYMAN, the following menu is displayed:

```
IBM Key Management Utility

Choose one of the following options to proceed.

1 - Create new key database
2 - Open key database
3 - Change database password
0 - Exit program

Enter your option number:
```

*Figure 5. Main Menu of the IBM Key Management Utility*

#### Using IKEYMAN

You do not have a secure network connection until you have created a key for secure network communications and received a certificate from a certificate authority (CA) who is designated as a trusted CA on your server. Use IKEYMAN to create key databases, public-private key pairs, and certificate requests. If you are acting as your own CA, you can use IKEYMAN to create self-signed certificates.

This section provides a quick reference of IKEYMAN tasks and detailed descriptions of the most common tasks.

**Related information:**
- “[Setting up access control and secure connections” on page 57](#)
- “[Chapter 8. Setting up a secure server” on page 45](#)

**User interface task reference**

The tasks you can perform using the IKEYMAN user interface are summarized in the following table. For a description of `ikeyman` command options, see "Command reference" on page 418.

<table>
<thead>
<tr>
<th>IKEYMAN task</th>
<th>See page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Change a key database password</td>
<td>413</td>
</tr>
</tbody>
</table>

Appendix G. Key Management Utility
Creating a new key database

A key database is a file that the server uses to store one or more key pairs and certificates. You can use one key database for all your key pairs and certificates or create multiple databases.

To create a new key database:

1. Change to the path where the new key database will be located.
2. Enter `ikeyman` to start the utility.
3. Enter `1` to create a new key database.
4. Enter the key database name.

   **Note:** If you do not use the default name (key.kdb) for your operational key database, you must use the server’s Configuration and Administration Forms to register your database with the server and specify it as the default key database. See “Registering a key database with the server” on page 413 for instructions.

   **CA Utility Note:** If you plan to use the HTTP Server CA utility to process certificate requests, your CA key database must be named `cakey.kdb` and copied into the following directory after you exit IKEYMAN:

   `/usr/lpp/internet/server_root/CAServlet/language_code/`

   For `language_code`, enter C (English) or Ja_JP (Japanese).

5. Enter a password using the guidelines in “Setting the database password” on page 413.
6. Enter the password again for verification.
7. Indicate whether the password should expire. If yes, accept the default of 60 days or enter the number of days. A message is displayed confirming that the database has been created.
8. Enter `1` to continue to work with the database.
9. Enter `11` to store the encrypted database password in a stash file. A message is displayed confirming that the file (database_name.sth) has been created.
Note: On OS/390, you must store the encrypted password for your operational key database in a stash file. If this is not done, the server will start but the connection will not be secure.

CA Utility Note: If you plan to use the HTTP Server CA utility to process certificate requests, you can optionally choose to store your encrypted CA key database password in a stash file. If the stash file is in the same directory as the CA key database file (cakey.kdb), then all certificates that are sent to be signed by the CA utility will be approved automatically. If you create a stash file (cakey.sth), copy that file into the following directory after you exit IKEYMAN:

```
/usr/lpp/internet/server_root/CAServlet/language_code/
```

For language_code, enter C (English) or Ja_JP (Japanese).

**Setting the database password:** When you create a new key database, you specify a key database password. This password is important because it protects the private key. The private key is the only key that can sign documents or decrypt messages encrypted with the public key. It’s a good practice to change the key database password frequently.

Use the following guidelines when specifying the password:
- The password must be from the U.S. English character set.
- The password should be at least six characters and contain at least two nonconsecutive numbers. Make sure the password doesn’t consist of publicly obtainable information about you, such as the initials and birth date for you, your spouse, or children.

Note: If you specify an expiration date for the password, keep track of when to change it. If the password expires before you change it, a message will be written to the error log. The server will start, but there will not be a secure network connection if the password has expired.

**Changing the database password:** To change an existing key database password:
1. Change to the path where the key database file is located.
2. Enter `ikeyman` to start the utility.
3. Enter 3 to change the password.
4. Enter the key database name.
5. Enter the old database password
6. Enter the new database password
7. Enter the new database password again for verification.

**Registering a key database with the server:** The initial configuration setting for the default key database name is key.kdb. If you use key.kdb as your default key database name, you do not need to register the database with the server. The server will use the initial setting on the KeyFile directive in the httpd.conf configuration file. If you do not use key.kdb as your default key database name or if you create additional key databases, you must register those databases with the server using the Configuration and Administration Forms.

To register a key database with the server and specify it as the default key database:
1. From the Front Page of your server, click **Configuration and Administration Forms.**
2. Click **Security Configuration,** then click **Key Database Configuration.**
Key Management Utility

3. Click the option to add the key database as the current (or default) key database for the server, and enter the path and name of the database, for example, /usr/lpp/internet/bin/mydatabase.kdb.

4. Click Submit to save the configuration.

5. To use the new configuration, stop and restart the server.

Creating a new key pair and certificate request
Key pairs and certificate requests are stored in a key database. To create a public-private key pair and certificate request:

1. Change to the path of the key database where the key pair and certificate request will be stored. If you have not created the key database, see “Creating a new key database” on page 412 for instructions.

2. Enter ikeyman to start the utility.

3. Enter 2 to open the key database.

4. Enter the database name.

5. Enter the database password.

6. Enter 3 to create a new key pair and certificate request.

7. Enter a certificate request file name, or press Enter to use the default file name, certreq.arm. The certificate request file is a PKCS10 file in Base64 encoded format.

8. Enter a name (label) that is used to identify the key and certificate in the database, for example, my key. Avoid using the word request when specifying labels.

9. Enter the number of the key size you want to use.

10. Enter the fully qualified host name of the server as the common name, for example, www.myserver.com.

11. Enter an organization name.

12. Optionally enter an organization unit.

13. Optionally enter a city or locality.

14. Optionally enter a state or province.

15. Enter a country code. You must specify at least 2 characters, for example, US.

A message is displayed confirming that the key pair (*.rdb) and certificate request (*.arm) files have been created.

Note: Do not attempt to edit or move the *.rdb file. If this file is not present or is corrupted when you attempt to receive the CA-signed certificate into the key database, you will have to resubmit your certificate request to the CA.

16. Enter 0 to exit the utility.

17. Start a Web browser and enter the URL of the certificate authority (CA) from whom you want to obtain the certificate. To send your certificate request, follow the instructions provided by the CA.

While you are waiting for the CA to process your certificate request, you can use a self-signed certificate that you create. For more information, see “Creating a self-signed certificate”.

Related information:
• “Certificate authorities supported by the HTTP Server” on page 54
• “Encryption support for the HTTP Server” on page 55

Creating a self-signed certificate
It usually takes two to three weeks to get a certificate from an external CA. While waiting for a certificate to be issued, you can use IKEYMAN to create a self-signed server certificate to enable SSL sessions between clients and the server. You also use this procedure if you are acting as your own CA for a private Web network.
To create a self-signed certificate:

1. Change to the path of the key database where the self-signed certificate will be stored. If you have not created the key database, see “Creating a new key database” on page 412 for instructions.
2. Enter ikeyman to start the utility.
3. Enter 2 to open the key database.
4. Enter the database name.
5. Enter the database password.
6. Enter 5 to create a self-signed certificate.
7. Enter the version number. The higher the version number, the more fields are included in the certificate. For example, a version 3 certificate request includes all possible fields.
8. Enter a name (label) that is used to identify the key and certificate in the database, for example, my self-signed certificate.
9. Enter the number of the key size you want to use.
10. Enter the fully qualified host name of the server as the common name, for example, www.myserver.com.
11. Enter an organization name.
12. Optionally enter an organization unit.
13. Optionally enter a city or locality.
14. Optionally enter a state or province.
15. Enter a country code. You must specify at least 2 characters, for example, US.
16. Indicate the number of days that the certificate will be valid or accept the default of 365 days.
17. Indicate whether the certificate will be the default key in the database.
18. Indicate whether you want to save the certificate to a file. If yes, indicate whether the output certificate file format is binary or Base64.
19. Enter a certificate file name, or press Enter to use the default file name, cert.arm.

Exporting keys

• To export keys to another key database:
  1. Change to the path of the key database you are exporting keys from.
  2. Enter ikeyman to start the utility.
  3. Enter 2 to open the key database.
  4. Enter the database name.
  5. Enter the database password.
  6. Enter 9 to export keys.
  7. Enter 1 to export keys to another database.
  8. Enter the number of the key to export. To export more than one key, use a comma as your delimiter, for example, 2, 5, 7.
  9. Enter the name of the key database you are exporting keys to. If the database is located in a different directory, include the fully qualified path.
  10. Enter the database password.

• To export keys to a PKCS12 file:
  1. Change to the path of the key database you are exporting keys from.
  2. Enter ikeyman to start the utility.
  3. Enter 2 to open the key database.
  4. Enter the database name.
  5. Enter the database password.
  6. Enter 9 to export keys.
  7. Enter 2 to export keys to a PKCS12 file.
  8. Accept the default output file name or enter a name.
  9. Enter a password using the guidelines in “Setting the database password” on page 413.
  10. Enter the password again for verification.
Key Management Utility

Importing keys

- To import keys from another key database:
  1. Change to the path of the key database you are importing keys to.
  2. Enter `ikeyman` to start the utility.
  3. Enter 2 to open the key database.
  4. Enter the database name.
  5. Enter the database password.
  6. Enter 8 to import keys.
  7. Enter 1 to import keys from another database.
  8. Enter the name of the key database you are importing keys from. If the database is located in a different directory, include the fully qualified path.
  9. Enter the password.
  10. Enter the number of the key to import. To import more than one key, use a comma as your delimiter, for example, 2, 5, 7.

- To import keys from a PKCS12 file:
  1. Change to the path of the key database you are importing keys to.
  2. Enter `ikeyman` to start the utility.
  3. Enter 2 to open the key database.
  4. Enter the database name.
  5. Enter the database password.
  6. Enter 8 to import keys.
  7. Enter 2 to import keys from a PKCS12 file.
  8. Enter the PKCS12 file name.
  9. Enter a password using the guidelines in “Setting the database password” on page 413.
  10. Enter the password again for verification.

Listing and managing keys and certificates

To list and manage keys and certificates:

1. Change to the path of the key database where the keys and certificates are stored.
2. Enter `ikeyman` to start the utility.
3. Enter 2 to open the key database.
4. Enter the database name.
5. Enter the database password.
6. Enter 1 to display the list of trusted CA keys in the database.
7. Select a key to work with, then select one of the following options:
   - Copy the certificate of this key to a file
   - Copy the certificate request of this key to a file
   - Delete the key
   - Export the key to another database
   - Export the key to a file
   - Remove trust root status
   - Set the selected key as default
   - Show key information
   - View the certificate of the key

Listing and managing request keys

To list and manage request keys:

1. Change to the path of the key database where the keys and certificates are stored.
2. Enter `ikeyman` to start the utility.
3. Enter 2 to open the key database.
4. Enter the database name.
5. Enter the database password.
6. Enter 2 to display the list of request keys in the database.
7. Select a request key to work with, then select one of the following options:
   - Copy the certificate request of this key to a file
   - Delete the key
   - Show key information

Listing trusted CAs
To display a list of trusted certificate authorities (CAs) in a key database:
1. Change to the path where the key database is located.
2. Enter `ikeyman` to start the utility.
3. Enter 2 to open the key database.
4. Enter the database name.
5. Enter the database password.
6. Enter 10 to list all trusted CAs.

Opening a key database
To open an existing key database:
1. Change to the path where the key database is located.
2. Enter `ikeyman` to start the utility.
3. Enter 2 to open the key database.
4. Enter the database name.
5. Enter the database password.

Receiving a certificate signed by a trusted CA
Use this procedure to receive a signed certificate from a certificate authority (CA) who is designated as a trusted CA on your server. By default, the following CA certificates are stored in the key database and marked as trusted CA certificates:
   - VeriSign Class 1 Public Primary CA
   - VeriSign Class 2 Public Primary CA
   - VeriSign Class 3 Public Primary CA
   - VeriSign Class 4 Public Primary CA
   - VeriSign Test CA
   - RSA Secure Server CA (from VeriSign)
   - Thawte Personal Basic CA
   - Thawte Personal Freemail CA
   - Thawte Personal Premium CA
   - Thawte Premium Server CA
   - Thawte Server CA

Note: If the CA who issues your CA-signed certificate is not a trusted CA in the key database, you must first store the CA’s certificate and designate the CA as a trusted CA. Then you can receive your signed certificate into the database. You cannot receive a signed certificate from a CA who is not a trusted CA. For instructions, see “Storing a CA’s certificate” on page 418.

To receive the CA-signed certificate into a key database:
1. Use the TSO RECEIVE command to put the certificate into a partitioned data set (PDS) file.
2. Use the OS/390 UNIX System Services command to put the certificate into the file system.
3. Change to the path where the key database is located.
4. Enter `ikeyman` to start the utility.
5. Enter 2 to open the key database.
6. Enter the database name.
7. Enter the database password.
8. Enter 4 to receive the CA-signed certificate.
9. Enter the certificate file name.
10. Indicate whether you want the certificate to be the default key in the key database.

**Showing the default key in a key database**

To display the default key entry:
1. Change to the path where the key database is located.
2. Enter `ikeyman` to start the utility.
3. Enter 2 to open the key database.
4. Enter the database name.
5. Enter the database password.
6. Enter 7 to display the default key.

**Storing a CA’s certificate**

To store a certificate from a CA who is not a trusted CA:
1. Change to the path where the key database is located.
2. Enter `ikeyman` to start the utility.
3. Enter 2 to open the key database.
4. Enter the database name.
5. Enter the database password.
6. Enter 6 to store the certificate.
7. Enter the name of the certificate.
8. Enter a label for the certificate.

To check that the CA is now trusted on your server, select option 10 to list all trusted CAs.

**Storing the encrypted database password in a stash file**

For a secure network connection, you must store the encrypted database password in a stash file after creating a new operational key database.

**CA Utility Note:** You can optionally choose to store your encrypted CA key database password in a stash file. If the stash file is in the same directory as the CA key database file (cakey.kdb), then all certificates that are sent to be signed by the CA utility will be approved automatically. If you create a stash file (cakey.sth), copy that file into the same directory as the cakey.kdb file.

To store the password:
1. Change to the path where the key database is located.
2. Enter `ikeyman` to start the utility.
3. Enter 2 to open the key database.
4. Enter the database name.
5. Enter the database password.
6. Enter 11 to store the encrypted database password in a file. A message is displayed confirming that the file (database_name.sth) has been created.

**Command reference**

Syntax: `ikeyman [-Flag[-Flag[-Flag]]]`

Flags:
- `-h` Display the IKEYMAN help file.
- `-m` Migrate a key ring file to a key database file.
  For an example, see “Migrate existing key ring files” on page 410.
- `-c` Convert a key database file to a key ring file.
  For an example, see “Migrate existing key ring files” on page 410.
Issue a certificate for the given certificate request to act as your own CA:

followed by a version number for the certificate (1, 2, or 3) (default=3)

The higher the version number, the more fields are included in the certificate. For example, a version 3 certificate includes all possible fields.

followed by the number of valid days for the certificate to be created (default=364 days); this number should be smaller than the number of valid days for the issuer’s certificate

followed by a certificate request file name

followed by a certificate file name

The default key of the given key database will be used as the issuer.

Export keys to a file in PKCS12 format:

from a key database file

from a key ring file

Only one key or all keys can be exported each time. The following flags can be used to specify the keys to be exported:

all keys

the default key

the first key

the key with the given label

If no option is given, the default key is exported.

Import keys from a file in PKCS12 format.

Store encrypted database password in a file.

followed by key_database_file_name.kdb

followed by key_ring_file_name.kyr

followed by a PKCS12 file to be imported or to store exported keys
Key Management Utility
Appendix H. Messages

Message updates on the Web ........................................... 421
Overview of IMW Messages ............................................. 421
Message severity ......................................................... 421
Message ID ranges and types .......................................... 421
IMW0001E-0412E: IMWHTTPD Messages .................. 422
Explanation of errno and errno2 codes in messages ........ 422
Message descriptions ................................................. 422
IMW2001E-2026E: Proxy Server Messages .............. 457
IMW3501I-3542E: CONSOLE Messages .................... 459
Explanation of process descriptor in messages .............. 459
Message descriptions ................................................. 459
IMW3701E-3726E: HTCounter Program Messages .... 463
IMW4000E-4018E: HTIMAGE Messages .................... 466
IMW5001E-5010E: HTADM Messages ...................... 467
IMW6102I-6805E: SSL Security Messages ................ 468

Message updates on the Web
For the most current information on messages, see the Messages section of the
Web-based WebSphere Troubleshooter for OS/390.

to link to the Troubleshooter, go to URL:

Overview of IMW Messages

Message severity
The alphabetic character at the end of each message ID indicates the message’s severity:
I Informational message.
E A recoverable error occurred. Follow the instructions given in the user
response to fix the problem.
S Fatal error. Contact the IBM Software Support Center.

Message ID ranges and types
Each server component has a range of message IDs:

<table>
<thead>
<tr>
<th>Message ID Range</th>
<th>Component</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMW0001-IMW2000</td>
<td>IMWHTTPD messages</td>
</tr>
<tr>
<td>IMW2001-IMW2500</td>
<td>Proxy Server messages</td>
</tr>
<tr>
<td>IMW3501-IMW3700</td>
<td>CONSOLE Messages</td>
</tr>
<tr>
<td>IMW3701-IMW3999</td>
<td>HTCounter Program Messages</td>
</tr>
<tr>
<td>IMW4000-IMW5000</td>
<td>HTIMAGE messages</td>
</tr>
<tr>
<td>IMW5001-IMW6000</td>
<td>HTADM messages</td>
</tr>
<tr>
<td>IMW6100-IMW6900</td>
<td>SSL Security Messages</td>
</tr>
</tbody>
</table>
IMWHTTPD Messages

IMW0001E-0412E: IMWHTTPD Messages

Explanation of errno and errno2 codes in messages

Many IMWHTTPD messages include OS/390 UNIX System Services return (errno) and reason (errno2) codes that can provide additional information about the cause of a failure or error.

For an explanation of these codes, see the following sections in the UNIX System Services Messages and Codes book:
- For errno codes, Return Codes Listed by Value
- For errno2 codes, Reason Codes Listed by Value

You can access the UNIX System Services Messages and Codes book on the Web at URL:


Message descriptions

**IMW0001E** Cannot get hostname. Icon URLs will not have host name.

**Explanation:** The server cannot determine its own hostname using gethostname(). The HTML generated by the server in response to directory list requests contains relative URLs for the file type icons (.gif) preceding each file in the list. The server normally generates absolute URLs that contain its own network name.

**User Response:** Correct your system or name server configuration so gethostname() returns the name of the system that your server is running on.

**IMW0002E** Cannot open the configuration file: name.

**Explanation:** Indicates configuration file name cannot be read.

**User Response:** Verify the name of the configuration file specified in the -r parameter of the server job stream (default: /etc/httpd.conf). If the named file exists, verify that your server has appropriate permissions to read the file.

**IMW0003E** Configuration not loaded due to errors.

**Explanation:** The configuration file contains errors and was not loaded. See the server error log or trace log for details.

**User Response:** Correct the configuration file and restart the server.

**IMW0004I** gc-mem-usage must be >20. Using 100.

**Explanation:** Small values prevent gc from working efficiently. You suggested: specifier.

**User Response:** Correct the configuration file.

**IMW0005E** Insufficient parameters for directive: directive.

**Explanation:** This directive requires more parameters than provided.

**User Response:** Correct the configuration file.

**IMW0006E** Must be MVS UserID, not UID: number.

**Explanation:** You entered a numeric UID, not a surrogate MVS user ID. The same UID number could be assigned to several MVS user IDs. To avoid ambiguity when accessing MVS resources, the server requires a surrogate MVS user ID. The UID and GID used for accessing OS/390 UNIX System Services resources is assigned from the specified surrogate MVS user ID.

**User Response:** Specify a surrogate MVS user ID or %%CLIENT%%.

**IMW0007E** Syntax error in configuration file.

**Explanation:** The correct syntax is:

```
Protection name {
  protection directive
}
```
IMW0008E  The cache lock timeout specifier is not valid: time specifier.
Explanation: You entered a time specifier that is not a valid amount of time.
User Response: Correct the configuration file.

IMW0009E  The cache time margin specifier is not valid: time specifier.
Explanation: You entered a time specifier that is not a valid amount of time.
User Response: Correct the configuration file.

IMW0010E  The cache_clean_def time specifier is not valid: time specifier.
Explanation: You entered a time specifier that is not a valid amount of time.
User Response: Correct the configuration file.

IMW0011E  The cache_unused_def time specifier is not valid: time specifier.
Explanation: You entered a time specifier that is not a valid amount of time.
User Response: Correct the configuration file.

IMW0012E  The CacheLastModifiedFactor is not valid: string.
Explanation: You entered a string that the server could not interpret as a real number (for example, 0.25) or a negative string.
User Response: Correct the configuration file.

IMW0013E  The configuration directive is not valid: directive.
Explanation: You entered a directive that is not valid or not spelled correctly.
User Response: Correct the configuration file.

IMW0014E  The daily gc time specifier is not valid: time specifier.
Explanation: You entered a time specifier that is not valid or a negative string.
User Response: Correct the configuration file.

IMW0015E  The default expiry time specifier is not valid: time specifier.
Explanation: You entered a time specifier that is not valid.
User Response: Correct the configuration file.

IMW0016E  The Dir directive is not recognized: directive.
Explanation: You entered a directive starting with Dir that is not valid or not spelled correctly.
User Response: Correct the configuration file.

IMW0017E  The DirAccess mode is not recognized: mode.
Explanation: You entered a mode that is not valid or not spelled correctly. The mode can be selective, a positive string, or a negative string.
User Response: Correct the configuration file.

IMW0018E  The DirReadme mode is not recognized: mode.
Explanation: You entered a mode that is not valid or not spelled correctly. The mode can be top, bottom, or a negative string.
User Response: Correct the configuration file.

IMW0019E  The DirShow directive is not valid: directive.
Explanation: You entered a directive that is not valid or not spelled correctly.
User Response: Correct the configuration file.

IMW0020E  The DirShowMaxDescriptionLength parameter is not valid: parameter.
Explanation: You entered a parameter which the server could not interpret as an integer.
User Response: Correct the configuration file.

IMW0021E  The DirShowMaxLength parameter is not valid: parameter.
Explanation: You entered a parameter which the server could not interpret as an integer.
User Response: Correct the configuration file.

IMW0022E  The DirShowMinLength parameter is not valid: parameter.
Explanation: You entered a parameter that the server could not interpret as an integer.
IMWHTTPD Messages

User Response:  Correct the configuration file.

IMW0023E  The FTPDirInfo parameter is not valid:  

parameter.

Explanation:  You entered a parameter that is not valid or not spelled correctly. The parameter can be top, bottom, or a negative string.

User Response:  Correct the configuration file.

IMW0024E  The Icon directive is not recognized:  

directive.

Explanation:  You entered a directive that is not valid or not spelled correctly.

User Response:  Correct the configuration file.

IMW0025E  The logfile format is not recognized (use 

Old or Common):  format.

Explanation:  You entered a format that is not valid or not spelled correctly.

User Response:  Correct the configuration file.

IMW0026E  The logtime is not recognized (use GMT 

or LocalTime):  parameter.

Explanation:  You entered a parameter that is not valid or not spelled correctly.

User Response:  Correct the configuration file.

IMW0027E  The method you tried to disable is not valid:  method name.

Explanation:  You entered a method that is not valid or not spelled correctly.

User Response:  Correct the configuration file.

IMW0028E  The method you tried to enable is not valid:  method name.

Explanation:  You entered a method that is not valid or not spelled correctly.

User Response:  Correct the configuration file.

IMW0029E  The number of parameters to directive is not valid:  directive.

Explanation:  You entered either too few or too many parameters for this directive.

User Response:  Correct the configuration file.

IMW0030E  The parameter to Port directive is not valid:  parameter.

Explanation:  You entered a parameter that the server could not interpret as an integer.

User Response:  Correct the configuration file.

IMW0031E  The protection setup name is not defined:  name.

Explanation:  You entered a name which is either not defined or not correctly spelled.

User Response:  Correct the configuration file.

IMW0032E  The proxy directive is not recognized:  

directive.

Explanation:  You entered a directive which is either not defined or not correctly spelled.

User Response:  Correct the configuration file.

IMW0033E  The SecurityLevel parameter is not valid:  

parameter.

Explanation:  You entered a parameter that is not valid. You can enter either normal or high.

User Response:  Correct the configuration file.

IMW0034E  The ServerType is not recognized:  string.

Explanation:  You entered a string that is not valid or not spelled correctly. The Web server supports only ServerType StandA lone.

User Response:  Correct the configuration file.

IMW0035E  The timeout specifier is not valid:  time specifier.

Explanation:  You entered a time specifier that is not valid.

User Response:  Correct the configuration file.

IMW0036E  There are too many arguments on one line in the configuration file. Max:  number.

Explanation:  You entered more arguments than are supported for any configuration directive.

User Response:  Correct the configuration file.

IMW0037E  Too many parameters for 

CacheLastModifiedFactor:  number.

Explanation:  You entered number strings on this line. CacheLastModifiedFactor accepts one or two parameters: an optional URL_template and a required real number.
User Response: Correct the configuration file.

**IMW0038E** Cannot open the cache access log: name.

Explanation: The server was unable to open the log file. Verify the log file name, the permissions, the user mask, and the available space in the file system.

User Response: Correct the log file name specified in the configuration file or correct the file system problem.

**IMW0039E** Cannot open the error log: name.

Explanation: The server was unable to open the log file. Verify the log file name, the permissions, the user mask, and the available space in the file system.

User Response: Correct the log file name specified in the configuration file or correct the file system problem.

**IMW0040E** Cannot open the log file: name.

Explanation: The server was unable to open the log file. Verify the log file name, the permissions, the user mask, and the available space in the file system.

User Response: Correct the log file name specified in the configuration file or correct the file system problem.

**IMW0041E** Cannot open the proxy access log: name.

Explanation: The server was unable to open the log file. Verify the log file name, the permissions, the user mask, and the available space in the file system.

User Response: Correct the log file name specified in the configuration file or correct the file system problem.

**IMW0042E** Cannot open the trace log: name.

Explanation: The server was unable to open the log file. Verify the log file name, the permissions, the user mask, and the available space in the file system.

User Response: Correct the log file name specified in the configuration file or correct the file system problem.

**IMW0043E** Relative log name when ServerRoot not specified: directive.

Explanation: Log file names that do not begin with a ‘/’ are relative to the ServerRoot directory.

User Response: Specify a ServerRoot directory or use absolute log names.

**IMW0044E** There were more errors, but queued only: number.

Explanation: Errors that occur prior to opening the error log are queued in memory. More errors occurred than could be queued.

User Response: Correct the indicated errors and restart the server.

**IMW0045S** Could not create adult_mtx.

Explanation: The server was not able to create a required mutex (lock) at initialization.

User Response: Verify that the Web server has sufficient virtual memory and is running with the required level of OS/390 UNIX System Services.

If necessary, contact the IBM Software Support Center for assistance.

**IMW0046S** Could not create auth_mtx.

Explanation: The server was not able to create a required mutex (lock) at initialization.

User Response: Verify that the Web server has sufficient virtual memory and is running with the required level of OS/390 UNIX System Services.

If necessary, contact the IBM Software Support Center for assistance.

**IMW0047S** Could not create free_presentation_mtx.

Explanation: The server was not able to create a required mutex (lock) at initialization.

User Response: Verify that the Web server has sufficient virtual memory and is running with the required level of OS/390 UNIX System Services.

If necessary, contact the IBM Software Support Center for assistance.

**IMW0048S** Could not initialize the translate tables (iconv).

Explanation: The server was not able to open the iconv translation service for translation between ASCII ISO8859-1 and EBCDIC IBM-1047.

User Response: Verify that the Web server is running with the required level of OS/390 UNIX System Services.

If necessary, contact the IBM Software Support Center for assistance.

**IMW0049E** Cannot open protection setup file: name.

Explanation: A Protect directive referenced a protection setup file that the server was unable to open for reading.

User Response: Verify that the file name is spelled correctly in the configuration file. Verify that the file exists and has the correct file permissions. Correct the configuration file or protection setup file.

---

**IMWHTTPD Messages**

Appendix H. Messages 425
IMW0050E Group not found: name.

Explanation: The group name is not defined or is not spelled correctly.

User Response: Verify the name in your protection mask directives and group file. Correct the configuration, protection setup, group or .acl file.

IMW0051E User not found: name.

Explanation: The user name is not defined or is not spelled correctly.

User Response: Verify the name in your protection mask directives and group file. Correct the configuration, protection setup, group or .acl file.

IMW0052E You did not specify a protection file in the Protect directory. Default protection was not set.

Explanation: The Protect directive that was matched for this request did not have an in-line protection setup, a named protection setup or a protection setup file name. This is only allowed when there is a preceding DefProt directive that also matches the current request. The client’s request is refused.

User Response: Correct the configuration file.

IMW0053E You did not specify the required protection file in the DefProt directive.

Explanation: The DefProt directive must always have an in-line protection setup, a named protection setup or a protection setup file name.

User Response: Correct the configuration file.

IMW0054E Cannot fstat() the new cache file: name.

Explanation: This is either an internal logic error or an HFS system problem.

User Response:
- Verify the cache root name defined in the configuration file.
- Verify that the server has appropriate permissions to the cache root and that the HFS is not damaged.

If necessary, contact the IBM Software Support Center for assistance.

IMW0055E Cannot get the cache information for: name.

Explanation: This is either an internal logic error or HFS system problem.

User Response:
- Verify the cache root name defined in the configuration file.
- Verify that the server has appropriate permissions to the cache root and that the HFS is not damaged.

If necessary, contact the IBM Software Support Center for assistance.

IMW0056E Cannot write the cache information entry for: string.

Explanation: This is either an internal logic error or HFS system problem.

User Response:
- Verify the cache root name defined in the configuration file.
- Verify that the server has appropriate permissions to the cache root and that the HFS is not damaged.

If necessary, contact the IBM Software Support Center for assistance.

IMW0057E Cannot write the cache information entry for: name.

Explanation: This is either an internal logic error or HFS system problem.

User Response:
- Verify the cache root name defined in the configuration file.
- Verify that the server has appropriate permissions to the cache root and that the HFS is not damaged.

If necessary, contact the IBM Software Support Center for assistance.

IMW0058E Content-Length mismatch when retrieved:

Explanation: This is either an internal logic error or HFS system problem.

User Response:
- Verify the cache root name defined in the configuration file.
- Verify that the server has appropriate permissions to the cache root and that the HFS is not damaged.

If necessary, contact the IBM Software Support Center for assistance.

IMW0059E HTLoadCacheToStream read error. Read returns: number.

Explanation: This is either an internal logic error or HFS system problem.

User Response:
- Verify the cache root name defined in the configuration file.
- Verify that the server has appropriate permissions to the cache root and that the HFS is not damaged.

If necessary, contact the IBM Software Support Center for assistance.
If necessary, contact the IBM Software Support Center for assistance.

<table>
<thead>
<tr>
<th>IMW0060E</th>
<th>The ExecDirPass mode is not valid: mode.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Explanation:</strong></td>
<td>You entered a <em>mode</em> that is not valid. The mode can be a positive or negative string.</td>
</tr>
<tr>
<td><strong>User Response:</strong></td>
<td>Enter a positive or negative value for ExecDirPass in the configuration file.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IMW0062I</th>
<th>CacheTrace.. On</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Explanation:</strong></td>
<td>You have requested cache access tracing in your configuration.</td>
</tr>
<tr>
<td><strong>User Response:</strong></td>
<td>None.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IMW0063E</th>
<th>Cannot bind and listen on port number. Ensure TCP/IP is configured and running and that httpd is not already running.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Explanation:</strong></td>
<td>The server was not able to get ownership of the specified TCP port.</td>
</tr>
<tr>
<td><strong>User Response:</strong></td>
<td>Make the port available for use. Verify that:</td>
</tr>
<tr>
<td></td>
<td>• The server is a superuser (if port number is below 1024).</td>
</tr>
<tr>
<td></td>
<td>• There must be an AF_INET transport provider connection to OS/390 UNIX System Services.</td>
</tr>
<tr>
<td></td>
<td>• The port number is reserved for use by OS/390 UNIX in all connected AF_INET transport providers.</td>
</tr>
<tr>
<td></td>
<td>• If a HostName is specified, there must not be any other application already using the specified port on the same IP address or on all OS/390 UNIX IP addresses.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IMW0064E</th>
<th>Cannot create socket.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Explanation:</strong></td>
<td>Indicates a problem with TCP/IP.</td>
</tr>
<tr>
<td><strong>User Response:</strong></td>
<td>Make sure TCP/IP is up and running.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IMW0065E</th>
<th>Cannot create socket.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Explanation:</strong></td>
<td>This is an internal error.</td>
</tr>
<tr>
<td><strong>User Response:</strong></td>
<td>If necessary, contact the IBM Software Support Center for assistance.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IMW0066E</th>
<th>Cannot get passwd entry for uid: number.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Explanation:</strong></td>
<td>This is an internal error.</td>
</tr>
<tr>
<td><strong>User Response:</strong></td>
<td>If necessary, contact the IBM Software Support Center for assistance.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IMW0067E</th>
<th>Cannot get passwd entry for user string</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Explanation:</strong></td>
<td>This is an internal error.</td>
</tr>
<tr>
<td><strong>User Response:</strong></td>
<td>If necessary, contact the IBM Software Support Center for assistance.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IMW0068E</th>
<th>Cannot initialize groups for user: id.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Explanation:</strong></td>
<td>This is an internal error.</td>
</tr>
<tr>
<td><strong>User Response:</strong></td>
<td>If necessary, contact the IBM Software Support Center for assistance.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IMW0069E</th>
<th>Cannot initialize groups for user name.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Explanation:</strong></td>
<td>This is an internal error.</td>
</tr>
<tr>
<td><strong>User Response:</strong></td>
<td>If necessary, contact the IBM Software Support Center for assistance.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IMW0070E</th>
<th>Cannot open cache file for reading: name.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Explanation:</strong></td>
<td>This is either an internal logic error or HFS system problem.</td>
</tr>
<tr>
<td><strong>User Response:</strong></td>
<td>Verify:</td>
</tr>
<tr>
<td></td>
<td>• The cache root name defined in the configuration file.</td>
</tr>
<tr>
<td></td>
<td>• The server has appropriate permissions to the cache root and that the HFS is not damaged.</td>
</tr>
<tr>
<td></td>
<td>If necessary, contact the IBM Software Support Center for assistance.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IMW0071E</th>
<th>Cannot open gc report file for reading: name.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Explanation:</strong></td>
<td>The garbage collection (cache cleanup) thread failed to complete normally.</td>
</tr>
<tr>
<td><strong>User Response:</strong></td>
<td>Check the error and trace logs for correctable problems.</td>
</tr>
<tr>
<td></td>
<td>If necessary, contact the IBM Software Support Center for assistance.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IMW0072E</th>
<th>Cannot open pid file for writing: name.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Explanation:</strong></td>
<td>The pid file cannot be opened.</td>
</tr>
<tr>
<td><strong>User Response:</strong></td>
<td>Verify the following:</td>
</tr>
<tr>
<td></td>
<td>• The name specified in the configuration file is valid.</td>
</tr>
<tr>
<td></td>
<td>• The file system is read/write.</td>
</tr>
<tr>
<td></td>
<td>• The file permissions are correct in the path.</td>
</tr>
<tr>
<td></td>
<td>• The server has the appropriate permissions.</td>
</tr>
<tr>
<td></td>
<td>Correct any problems found with the pid file or configuration.</td>
</tr>
</tbody>
</table>
IMWHTTPD Messages

**IMW0073E Cannot open pid file: name.**

**Explanation:** The pid file cannot be opened.

**User Response:** Verify the following:
- The name specified in the configuration is valid.
- The file has not been erased or damaged by another user.
- The file permissions are correct in the path.
- The server has the appropriate permissions.

Correct any problems found with the pid file or configuration.

**IMW0074E Cannot read pid from name.**

**Explanation:** The pid file cannot be read.

**User Response:** Verify the following:
- The name specified in the configuration is valid.
- Verify that the file has not been erased or damaged by another user.
- Verify that the file permissions are correct in the path.
- Verify that the server has the appropriate permissions.

Correct any problems found with the pid file or configuration.

**IMW0075E Cannot set group id to: number.**

**Explanation:** This is an internal error.

**User Response:** Contact the IBM Software Support Center for assistance.

**IMW0076E Cannot set parent group id to: number.**

**Explanation:** This is an internal error.

**User Response:** Contact the IBM Software Support Center for assistance.

**IMW0077E Cannot set parent user id to: number.**

**Explanation:** This is an internal error.

**User Response:** Contact the IBM Software Support Center for assistance.

**IMW0078E Cannot set socket option.**

**Explanation:** This is an internal error.

**User Response:** Contact the IBM Software Support Center for assistance.

**IMW0079E Cannot set socket option.**

**Explanation:** This is an internal error.

**User Response:** Contact the IBM Software Support Center for assistance.

**IMW0080E Cannot set user id to: number.**

**Explanation:** This is an internal error.

**User Response:** Contact the IBM Software Support Center for assistance.

**IMW0081E Configuration file is required.**

**Explanation:** Indicates configuration file could not be found.

**User Response:** Provide a valid configuration file.

**IMW0082E Connection interrupted (SIGPIPE)**

**Explanation:** The client broke the connection prior to sending all of the request.

**User Response:** None.

**IMW0083E Connection interrupted (SIGPIPE), req: request**

**Explanation:** The client broke the connection prior to receiving all of the data.

**User Response:** None.

**IMW0084E CRASHED (bus error, core dumped).**

**Explanation:** This is an internal error.

**User Response:** Contact the IBM Software Support Center for assistance.

**IMW0085E ABEND signal signal_number has been caught for thread type thread_number**

**Explanation:** The Web server’s ABEND recovery signal handler has detected an ABEND condition. A CEE dump may be taken for this error. The Web server may recover from this error.

**signal_number can be:**

| 4 | SIGILL (Illegal instruction): An attempt was made to execute an illegal instruction. |
| 8 | SIGFPE (Fixed-point exception): An attempt was made to perform an arithmetic operation that would have resulted in an overflow condition, such as a divide by zero. |
| 11 | SIGSEGV (Segmentation violation): An attempt was made to access inaccessible storage. |

**thread_number can be:**
If a CEE dump was taken, determine if the ABEND occurred in Web server code or in a plug-in or other code. If the ABEND occurred in Web server code, report the problem to the IBM Software Support Center. If the Web server recovered from the ABEND, stop and restart the Web server as soon as you can to ensure that resources are properly cleaned up.

### IMW0086E

**Crashing request was:** `string`.

**Explanation:** This is an internal error.

**User Response:** Contact the IBM Software Support Center for assistance.

### IMW0087E

**Extra command line parameter:** `parameter`.

**Explanation:** Additional unexpected parameters were specified on the `/EXEC JCL` statement.

**User Response:** Correct the JCL and resubmit.

### IMW0088E

**Extra parameter:** `parameter`.

**Explanation:** Additional unexpected parameters were specified on the `/EXEC JCL` statement.

**User Response:** Correct the JCL and resubmit.

### IMW0089E

**Forbidden by rule.**

**Explanation:** This string is returned by the server in the HTTP status line on requests that are not allowed by your configuration file.

**User Response:** None.

### IMW0090E

**gc report file is empty:** `string`.

**Explanation:** The garbage collection (cache cleanup) thread failed to complete normally.

**User Response:** Check the error and trace logs for correctable problems.

If necessary, contact the IBM Software Support Center for assistance.

### IMW0091E

**gc report has wrong process id. Real gc might have crashed.**

**Explanation:** The garbage collection (cache cleanup) thread failed to complete normally.

**User Response:** Check the error and trace logs for correctable problems.

If necessary, contact the IBM Software Support Center for assistance.

### IMW0092E

**gc was returned, but caching is not occurring.**

**Explanation:** This is an internal error.

**User Response:** Contact the IBM Software Support Center for assistance.

### IMW0093E

**Method `method name` is not enabled on this server.**

**Explanation:** This string is returned by the server in the HTTP status line on requests that specify a method name that is not enabled in your configuration.

**User Response:** For this request to succeed, explicitly enable this method in your configuration.

### IMW0094E

**Method `method name` is not valid or not implemented.**

**Explanation:** This string is returned by the server in the HTTP status line on requests that specify an unrecognized method name.

**User Response:** None.

### IMW0095E

**Permission denied.**

**Explanation:** This string is returned by the server in the HTTP status line of the request.

**User Response:** Verify that the user ID that the server is running under has the appropriate permissions.

### IMW0096E

**Request parsing failed.**

**Explanation:** This is an internal error.

**User Response:** Contact the IBM Software Support Center for assistance.

### IMW0097E

**Restart failed.**

**Explanation:** This is an internal error.

**User Response:** Contact the IBM Software Support Center for assistance.
IMW0098I Restart succeeded.
Explanation: The restart signal was successfully sent to the server.
User Response: None.

IMW0099I Restarting.. httpd
Explanation: The server has quiesced all requests and is reloading the configuration file.
User Response: None.

IMW0100E Script timed out.
Explanation: A CGI program has failed to complete processing within the time specified in your configuration and will be terminated.
User Response: None.

IMW0101I Sending..... HUP signal to process: number.
Explanation: A warning signal has been sent to a CGI program that has timed out.
User Response: None.

IMW0102E SETUID ERROR: there is no such user: string.
Explanation: This is an internal error.
User Response: Contact the IBM Software Support Center for assistance.

IMW0103I SIGHUP caught: reloading configuration files.
Explanation: The server has received the restart signal.
User Response: None.

IMW0104E The -d option is not valid: string
Explanation: The -dxx parameter (diraccess and dirreadme override) is not specified correctly.
User Response: Correct the JCL and resubmit.

IMW0105E The cache root was not specified when started in gc-only mode
Explanation: A cache root is required to run garbage collection.
User Response: Correct the configuration.

IMW0106E The cache root was not specified when started in gc-only mode string
Explanation: A cache root is required to run garbage collection.
User Response: Correct the configuration.

IMW0107E The command line option is not recognized: string.
Explanation: You entered an option that is either not defined or not correctly spelled.
User Response: Correct the JCL and resubmit.

IMW0108E The command line option is not recognized: string.
Explanation: You entered an option that is either not defined or not correctly spelled.
User Response: Correct the JCL and resubmit.

IMW0109E The parameter to -p option is not valid: parameter.
Explanation: You entered a parameter that is not a valid port number.
User Response: Correct the JCL and resubmit.

IMW0110E The request is not valid: string.
Explanation: You entered an HTTPD request string that is not properly formatted. This may be from an improperly written client or a user trying to break in by telneting to the server port.
User Response: None.

IMW0111E The request is not valid: string.
Explanation: The server received an HTTPD request string that is not properly formatted. This may be from an improperly written client or a user trying to break in by telneting to the server port.
User Response: None.

IMW0112E The request is not valid or not recognized: string.
Explanation: The server received an HTTPD request string that contained an unrecognized method.
User Response: None.

IMW0113E There is no parameter for -errlog option.
Explanation: This parameter must be followed by the errorlog name.
User Response: Correct the JCL and resubmit.
IMW0114E  There is no parameter for -errlog option.
Explanation:  This parameter must be followed by the errorlog name.
User Response:  Correct the JCL and resubmit.

IMW0115E  There is no such process.
Explanation:  The server was started with the -restart parameter. The pid file specified in the configuration contains an incorrect PID number. Either the server is no longer running or the wrong pid file was used.
User Response:  Correct the JCL and resubmit.

IMW0116E  Timeout occurred when reading request.
Explanation:  A complete request was not received from the client within the amount of time specified in the configuration. The connection from the client has been closed.
User Response:  None.

IMW0117E  Timeout occurred when sending response.
Explanation:  The server was not able to complete sending the response within the amount of time specified in the configuration. The connection from the client has been closed.
User Response:  None.

IMW0118E  Translated NULL when should do search.
Explanation:  A search request URL was received from the client that did not contain a QUERY_STRING.
User Response:  None.

IMW0119E  Unknown error occurred.
Explanation:  This is an internal error.
User Response:  Contact the IBM Software Support Center for assistance.

IMW0120E  You did not specify a parameter for -l option.
Explanation:  This parameter must be followed by the name of the log.
User Response:  Correct the JCL and resubmit.

IMW0121E  You did not specify a parameter for -l/-newlog/-oldlog option
Explanation:  This parameter must be followed by the name of the log.

IMW0122E  Empty item not allowed.
Explanation:  Your group file is missing field specific information.
User Response:  Correct the group file.

IMW0123E  Expected address part (single address or list).
Explanation:  Item in message is missing from the group file.
User Response:  Correct the group file.

IMW0124E  Expecting ')' closing address list.
Explanation:  Item in message is missing from the group file.
User Response:  Correct the group file.

IMW0125E  Expecting ')' closing user/group list.
Explanation:  Item in message is missing from the group file.
User Response:  Correct the group file.

IMW0126E  Expecting a single address or '(' beginning list.
Explanation:  Item in message is missing from the group file.
User Response:  Correct the group file.

IMW0127E  Expecting a single name or '(' beginning list.
Explanation:  Item in message is missing from the group file.
User Response:  Correct the group file.

IMW0128E  Expecting an address template.
Explanation:  Item in message is missing from the group file.
User Response:  Correct the group file.

IMW0129E  Expecting field separator.
Explanation:  Item in message is missing from the group file.
User Response:  Correct the group file.
IMW0130E   Expecting group name.
Explanation: Item in message is missing from the 
group file.
User Response: Correct the group file.

IMW0131E   Expecting user or group name.
Explanation: Item in message is missing from the 
group file.
User Response: Correct the group file.

IMW0132I   Group is NULL
Explanation: The group has no name.
User Response: None.

IMW0133I   Group: group_name.
Explanation: Indicates group name.
User Response: None.

IMW0134I   NULL-ITEM
Explanation: Indicates an empty item in group file.
User Response: None.

IMW0135I   NULL RECORD
Explanation: Indicates an empty line in the group file.
User Response: None.

IMW0136E   There is garbage after group definition.
Explanation: There is unrecognized information in the 
group file.
User Response: None.

IMW0137I   Accepted
Explanation: Returned to client program in the HTTP 
status line.
User Response: None.

IMW0138E   An error occurred that has no 
explanation. Call IBM software support.
Explanation: This is an internal error.
User Response: Contact the IBM Software Support 
Center for assistance.

IMW0139I   Created
Explanation: Returned to client program in the HTTP 
status line.
User Response: None.

IMW0140E   Forbidden
Explanation: Returned to client program in the HTTP 
status line.
User Response: None.

IMW0141I   Found
Explanation: Returned to client program in the HTTP 
status line.
User Response: None.

IMW0142E   Internal error. This is a software 
problem. Call IBM software support.
Explanation: This is an internal error.
User Response: Contact the IBM Software Support 
Center for assistance.

IMW0143E   Method
Explanation: Returned to client program in the HTTP 
status line.
User Response: None.

IMW0144E   Moved
Explanation: Returned to client program in the HTTP 
status line.
User Response: None.

IMW0145E   No response
Explanation: Returned to client program in the HTTP 
status line.
User Response: None.

IMW0146E   Not authorized.
Explanation: Returned to client program in the HTTP 
status line.
User Response: None.

IMW0147E   Not found.
Explanation: Returned to client program in the HTTP 
status line.
User Response: None.
IMW0148E  Not implemented
Explanation: Returned to client program in the HTTP status line.
User Response: None.

IMW0149E  Not modified
Explanation: Returned to client program in the HTTP status line.
User Response: None.

IMW0150I  OK
Explanation: Returned to client program in the HTTP status line.
User Response: None.

IMW0151E  Partial information
Explanation: Returned to client program in the HTTP status line.
User Response: None.

IMW0152E  Payment required.
Explanation: Returned to client program in the HTTP status line.
User Response: None.

IMW0153E  The request is not valid.
Explanation: Returned to client program in the HTTP status line.
User Response: None.

IMW0154E  Cannot open cache report file for writing: string.
Explanation: Verify the file name and permissions. Verify that the file is system is mounted, read/write and not full.
User Response: Correct any problems found.

IMW0155E  Cannot rewrite cache info file in directory: string.
Explanation: Verify the file name and permissions. Verify that the file is system is mounted, read/write and not full.
User Response: Correct any problems found.

IMW0156E  Internal error: spawn() failed.
errno=return_code, errno2=reason_code, errmsg=string
Explanation: The server received an error when it tried to spawn a new address space to run a CGI program. The OS/390 UNIX System Services error information and message are displayed.
The OS/390 UNIX System Services return_code and reason_code may provide additional information about the cause of the problem. See Explanation of errno and errno2 codes in messages on page 422.
User Response: Verify:
• That the file permissions allow execution under the user ID used to process this request.
• You have not exceeded the number of OS/390 UNIX address spaces allowed.
Correct the OS/390 UNIX environment problem.

If necessary, contact the IBM Software Support Center for assistance.

IMW0157E  Internal error: Cannot create pipe.
Explanation: This is an internal error.
User Response: Contact the IBM Software Support Center for assistance.

IMW0158E  Internal error: Cannot read script output pipe.
Explanation: This is an internal error.
User Response: Contact the IBM Software Support Center for assistance.

IMW0159E  The preparse script request is not valid.
string is not executable.
Explanation: The preparse script program string is not marked as executable. Verify that the HTML page, the script name and permissions are correct.
User Response: Correct any problems.

IMW0160E  The script execution request is not valid.
Explanation: The script program is not found. Verify that the HTML page, the script name and permissions are correct.
User Response: Correct any problems.

IMW0161E  The script request is not valid. No variation of string is executable.
Explanation: An executable script program is not found. The server also looked for variations of the named program by appending .sh and .pp to the requested name.
If the following message is accompanied by Error 500, this indicates that the helpout file cannot be executed because of a permission bit problem.

**IMW0161E** The script request is not valid. No variation of /usr/lpp/internet/server_root/admin-bin/helpout is executable.

**User Response:** Verify that the HTML page, the script name and permissions are correct. Correct any problems.

If you suspect a permission bit problem, check permission bits along the directory structure using the `ls -dl` command to make sure that the execute bit is turned on. The following example shows a valid set of permission bits:

```bash
$ ls -dl /
 drwxr-xr-x 37 IBMUSER OMVSGRP 0 Sep 29 09:01 /
 $ ls -dl /usr
 drwxr-xr-x 14 IBMUSER OMVSGRP 0 Jul 15 13:52 /usr
 $ ls -dl /usr/lpp
 drwxr-xr-x 18 IBMUSER OMVSGRP 0 Aug 20 07:27 /usr/lpp
 $ ls -dl /usr/lpp/internet
 drwxr-xr-x 9 USER88 WEBTEAM 0 Aug 20 14:37 /usr/lpp/internet
 $ ls -dl /usr/lpp/internet/server_root
 drwxr-xr-x 14 USER88 WEBTEAM 0 Sep 28 13:13 /usr/lpp/internet/server_root
 $ ls -dl /usr/lpp/internet/server_root/admin-bin
 drwxr-xr-x 4 USER88 WEBTEAM 0 Aug 20 14:20 /usr/lpp/internet/server_root/admin-bin
 $ ls -dl /usr/lpp/internet/server_root/admin-bin/helpout
 drwxr-xr-x 1 USER88 WEBTEAM 0 Aug 12 17:10 /usr/lpp/internet/server_root/admin-bin/helpout
```

**IMW0162E** Internal error running function `function_name` from DLL module `file_name`

**Explanation:** This message is sent to a client when an ABEND occurs in a plug-in running on behalf of a request sent by the client. The plug-in by the name of `function_name` from `file_name` was executed by the Web server. The Web server was able to recover from the ABEND, and a CEE dump may have been taken. Message IMW0085E should also appear in the error log.

**User Response:** Retry the request. If the failure occurs again, correct the error before retrying. See message IMW0085E for additional information.

**IMW0163E** RFC931: bind() failed: BIND_ERROR

**Explanation:** This is an internal error.

**User Response:** Contact the IBM Software Support Center for assistance.

**IMW0164E** RFC931: fdopen() failed: FDOPEN_ERROR

**Explanation:** This is an internal error.

**User Response:** Contact the IBM Software Support Center for assistance.

**IMW0165E** RFC931: socket() failed: SOCKET_ERROR

**Explanation:** This is an internal error.

**User Response:** Contact the IBM Software Support Center for assistance.

**IMW0166E** An error occurred while allocating storage for SSI processing

**Explanation:** During processing of a server-side include request, a memory allocation request failed.

**User Response:** Ensure there is enough memory for the server.

**IMW0167E** directive failed: `var = variable_text, value value_text`

**Explanation:** During processing of a server-side include, the server attempted to resolve `directive` but failed. The `variable_text` can be specified using: VAR, FILE, CMNTMSG, ERRORMSG, TIMEFMT, or SIZEFMT.

**User Response:** Correct the error.

**IMW0168E** Expected `name` but found `string`

**Explanation:** During processing of a server-side include, the server encountered `string` when it was expecting `name`.

**User Response:** Correct the error.

**IMW0169E** Internal server error: Imbedded BINARY code page is not supported.

**Explanation:** During processing of a server-side include request, the server attempted to process binary data when it is already processing non-binary data.

**User Response:** Contact the IBM Software Support Center for assistance.

**IMW0170E** Stat on file failed: `name`

**Explanation:** During processing of a server-side include request, the server attempted to gather statistics on file `name`, but failed.

**User Response:** Verify `name` exists and the permissions are correct.

---

IMW0162E Internal error running function `function_name` from DLL module `file_name`

**Explanation:** This message is sent to a client when an ABEND occurs in a plug-in running on behalf of a request sent by the client. The plug-in by the name of `function_name` from `file_name` was executed by the Web server. The Web server was able to recover from the ABEND, and a CEE dump may have been taken. Message IMW0085E should also appear in the error log.

**User Response:** Retry the request. If the failure occurs again, correct the error before retrying. See message IMW0085E for additional information.

**IMW0163E** RFC931: bind() failed: BIND_ERROR

**Explanation:** This is an internal error.

**User Response:** Contact the IBM Software Support Center for assistance.

**IMW0164E** RFC931: fdopen() failed: FDOPEN_ERROR

**Explanation:** This is an internal error.

**User Response:** Contact the IBM Software Support Center for assistance.

**IMW0165E** RFC931: socket() failed: SOCKET_ERROR

**Explanation:** This is an internal error.

**User Response:** Contact the IBM Software Support Center for assistance.

**IMW0166E** An error occurred while allocating storage for SSI processing

**Explanation:** During processing of a server-side include request, a memory allocation request failed.

**User Response:** Ensure there is enough memory for the server.

**IMW0167E** directive failed: `var = variable_text, value value_text`

**Explanation:** During processing of a server-side include, the server attempted to resolve `directive` but failed. The `variable_text` can be specified using: VAR, FILE, CMNTMSG, ERRORMSG, TIMEFMT, or SIZEFMT.

**User Response:** Correct the error.

**IMW0168E** Expected `name` but found `string`

**Explanation:** During processing of a server-side include, the server encountered `string` when it was expecting `name`.

**User Response:** Correct the error.

**IMW0169E** Internal server error: Imbedded BINARY code page is not supported.

**Explanation:** During processing of a server-side include request, the server attempted to process binary data when it is already processing non-binary data.

**User Response:** Contact the IBM Software Support Center for assistance.

**IMW0170E** Stat on file failed: `name`

**Explanation:** During processing of a server-side include request, the server attempted to gather statistics on file `name`, but failed.

**User Response:** Verify `name` exists and the permissions are correct.
| IMW0171E strftime() call failed: TIMEFMT timefmt, file name, errno=return_code, errno2=reason_code.

**Explanation:** During processing of a server-side include request, the server attempted a call to the C function strftime() with the time format string timefmt for document name.

**User Response:** Verify the time format string is correct and the server has enough memory to process the request.

The OS/390 UNIX System Services return_code and reason_code may provide additional information about the cause of the problem. See “Explanation of errno and errno2 codes in messages” on page 422.

See “Using server-side includes to insert information into CGI programs and HTML documents” on page 93 for more information on using server-side includes and directives.

| IMW0172E The directive was not complete on HTSSI_abort

**Explanation:** During processing of a server-side include request, abnormal termination of the output stream occurred while processing a server-side include directive.

**User Response:** Correct the error.

See “Using server-side includes to insert information into CGI programs and HTML documents” on page 93 for more information on using server-side includes and directives.

| IMW0173E The directive was not complete on HTSSI_free

**Explanation:** During processing of a server-side include request, termination of the output stream occurred while processing a server-side include directive.

**User Response:** Correct the error.

See “Using server-side includes to insert information into CGI programs and HTML documents” on page 93 for more information on using server-side includes and directives.

| IMW0174E The directive is not supported: directive

**Explanation:** During processing of a server-side include request, the document contained a directive which is not valid or not spelled correctly.

**User Response:** Correct the error.

See “Using server-side includes to insert information into CGI programs and HTML documents” on page 93 for more information on using server-side includes and directives.

| IMW0175E The directive was not valid: string

**Explanation:** During processing of a server-side include request, the document contained a directive that was not valid or not correctly spelled.

**User Response:** Correct the error.

See “Using server-side includes to insert information into CGI programs and HTML documents” on page 93 for more information on using server-side includes and directives.

| IMW0176E The text was not valid: string

**Explanation:** During processing of a server-side include request, the directive text string was found to contain an illegal character.

**User Response:** Correct the error.

See “Using server-side includes to insert information into CGI programs and HTML documents” on page 93 for more information on using server-side includes and directives.

| IMW0177E SIZEFMT is not valid: value

**Explanation:** During processing of a server-side include request, the document contained a file size format value which is either not valid or not correctly spelled. Valid values are bytes and abbrev.

**User Response:** Correct the error.

See “Using server-side includes to insert information into CGI programs and HTML documents” on page 93 for more information on using server-side includes and directives.

| IMW0178E There was an illegal attempt to use .. in file name name.

**Explanation:** During processing of a server-side include request, the document attempted to access the document name which is outside SSI_ROOT and its descendents.

**User Response:** Correct the error.

See “Using server-side includes to insert information into CGI programs and HTML documents” on page 93 for more information on using server-side includes and directives.

| IMW0179E VARREF was not valid: varref

**Explanation:** During processing of a server-side include request, the variable reference varref was found to contain an illegal character.

**User Response:** Correct the error.

See “Using server-side includes to insert information into CGI programs and HTML documents” on page 93 for more information on using server-side includes and directives.
IMWHTTPD Messages

for more information on using server-side includes and directives.

IMW0180E Not authorized. Proxy-Authentication failed (or your browser does not support it).
Explanation: The proxy request requires a user ID and password.
User Response: Provide a user ID and password if the browser supports the PROXY-AUTHENTICATE function.

IMW0181E The Idle thread timeout specifier is not valid: timeout specifier
Explanation: You entered a timeout specifier that is not valid.
User Response: Correct the configuration file.

IMW0182E The HomeDirs directive is not valid. You did not specify a directory.
Explanation: You entered a HomeDirs directive that is not valid.
User Response: Correct the configuration file.

IMW0183E Unknown error type error type
Explanation: The ErrorPage directive in the configuration file is not correct.
User Response: Correct the configuration file.

IMW0184E No error file specified for error type error type
Explanation: The ErrorPage directive in the configuration file is not correct.
User Response: Correct the configuration file.

IMW0185E The AccessLogExcludeURL directive is not valid. You did not specify a Method.
Explanation: You entered an AccessLogExcludeURL directive that is not valid.
User Response: Correct the configuration file.

IMW0186E The AccessLogExcludeMethod directive is not valid. You did not specify a Method.
Explanation: You entered an AccessLogExcludeMethod directive that is not valid.
User Response: Correct the configuration file.

Explanation: You entered an AccessLogExcludeReturnCode directive that is not valid or did not enter a Return Code.
User Response: Correct the configuration file.

IMW0188E The AccessLogExcludeMimeType directive is not valid. You did not specify a MimeType.
Explanation: You entered an AccessLogExcludeMimeType directive that is not valid.
User Response: Correct the configuration file.

IMW0189E Invalid API directive.
Explanation: You entered an API directive that is not valid.
User Response: Correct the configuration file.

IMW0190E Invalid parameter for UseAcls directive: parameter
Explanation: You entered a parameter for the UseAcls directive that is not valid.
User Response: Correct the configuration file.

IMW0191E iconv() does not support the combination of codepages specified.
Explanation: Specifications for DefaultFsCp and DefaultNetCp in the configuration file are not compatible.
User Response: Verify and correct the code pages that the server runs under.

IMW0192E Cannot open the CGI error log: name
Explanation: The server was unable to open the CGI log file. Verify the log file name, the permissions, the user mask, and the available space in the file system.
User Response: Correct the log file name specified in the configuration file or correct the file system problem.

IMW0193I OK
Explanation: The page was served successfully.
User Response: None.

436 HTTP Server Planning, Installing, and Using
<table>
<thead>
<tr>
<th>Message Code</th>
<th>Message Code</th>
<th>Explanation</th>
<th>User Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMW0194I</td>
<td>OK-GATEWAY</td>
<td>The page was served successfully.</td>
<td>None.</td>
</tr>
<tr>
<td>IMW0195I</td>
<td>OK-REDIRECT</td>
<td>The page was served successfully.</td>
<td>None.</td>
</tr>
<tr>
<td>IMW0196I</td>
<td>NOT AUTHENTICATED</td>
<td>The user was not authenticated, so the page was not served.</td>
<td>Enter a valid user ID and password.</td>
</tr>
<tr>
<td>IMW0197E</td>
<td>NOT AUTHORIZED</td>
<td>The user was not in the mask.</td>
<td>Enter a user that is in the ACL for the page.</td>
</tr>
<tr>
<td>IMW0198E</td>
<td>FORBIDDEN BY IP</td>
<td>The IP address of the browser is not in the group.</td>
<td>Add the IP address to the group.</td>
</tr>
<tr>
<td>IMW0199E</td>
<td>FORBIDDEN BY IP FOR PROXY</td>
<td>Running as a proxy server.</td>
<td>Add the IP address to the group.</td>
</tr>
<tr>
<td>IMW0200E</td>
<td>FORBIDDEN BY RULE</td>
<td>Could not get a page because it did not pass a rule.</td>
<td>Correct the rule.</td>
</tr>
<tr>
<td>IMW0201E</td>
<td>NO ACL FILE</td>
<td>Could not read any of the ACL files.</td>
<td>Create ACL files, or set ACL settings to NEVER.</td>
</tr>
<tr>
<td>IMW0202E</td>
<td>NO ACL ENTRY</td>
<td>There are no allowed groups in the Access Control List (ACL) file.</td>
<td>Include groups in the ACL file or turn checking off.</td>
</tr>
<tr>
<td>IMW0203E</td>
<td>* SETUP ERROR *</td>
<td>Protection file not found or syntax error in file.</td>
<td>Correct the file.</td>
</tr>
<tr>
<td>IMW0204E</td>
<td>..IN URL</td>
<td>Cannot have .. in URL.</td>
<td>Remove .. from URL.</td>
</tr>
<tr>
<td>IMW0205E</td>
<td>HTBIN OFF</td>
<td>Htbin is not enabled on your server.</td>
<td>Enable htbin on your server.</td>
</tr>
<tr>
<td>IMW0206E</td>
<td>INVALID REDIRECT</td>
<td>The configuration file contains a redirect that is not valid.</td>
<td>Correct the redirection.</td>
</tr>
<tr>
<td>IMW0207E</td>
<td>NO SUCH USER</td>
<td>The user directory is not correct.</td>
<td>Correct the configuration file.</td>
</tr>
<tr>
<td>IMW0208E</td>
<td>PUT NOT ALLOWED</td>
<td>PUT/DELETE must be explicitly allowed.</td>
<td>Correct the configuration file.</td>
</tr>
<tr>
<td>IMW0209E</td>
<td>NOT FOUND</td>
<td>The file does not exit or is read protected.</td>
<td>Correct the configuration file.</td>
</tr>
<tr>
<td>IMW0210E</td>
<td>MULTI FAILED</td>
<td>The file does not exit or is read protected.</td>
<td>Correct the configuration file.</td>
</tr>
<tr>
<td>IMW0211E</td>
<td>?</td>
<td>Indicates unknown error.</td>
<td>None.</td>
</tr>
<tr>
<td>IMWHTTPD Messages</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
| **IMW0212E** Cannot find help file *name*. Help is disabled.  
*Explanation:* The help file you requested cannot be found.  
*User Response:* Try again. |
| **IMW0213E** Cannot initialize application Help. Help is disabled.  
*Explanation:* The help application cannot be initialized.  
*User Response:* Correct the configuration. |
| **IMW0214E** Help is not available.  
*Explanation:* Help is not available.  
*User Response:* None. |
| **IMW0215E** Help is not available for this item.  
*Explanation:* Help is not available for the information.  
*User Response:* None. |
| **IMW0216E** Not authorized. Authentication failed.  
*Explanation:* You are not allowed to do the function you are trying to do.  
*User Response:* Verify the user ID, password, and protect rule. |
| **IMW0217E** Not authorized to access the document.  
*Explanation:* You are not allowed to access the document.  
*User Response:* Contact your system programmer to increase your privilege. |
| **IMW0218E** Forbidden by rule.  
*Explanation:* You specified an action that is forbidden by the rules specified in the configuration file.  
*User Response:* Correct the configuration file. |
| **IMW0219E** Server will not serve to your IP address.  
*Explanation:* The mask specifications indicate that the client is not coming from a supported IP address.  
*User Response:* Correct the configuration file. |
| **IMW0220E** Proxy server will not serve to your IP address (at least with this HTTP method).  
*Explanation:* The mask specifications indicate that the client is not coming from a supported IP address. |
| **IMW0221E** Access to this file is not allowed (no ACL file).  
*Explanation:* The file is protected, but no mask or ACL is specified. This is mandatory for proxy protection.  
*User Response:* Correct the configuration file. |
| **IMW0222E** Access to this file is not allowed (no ACL entry).  
*Explanation:* No ACL entry is specified.  
*User Response:* Correct the configuration file. |
| **IMW0223E** Server protection setup error occurred.  
Probably the protection setup file was not found or it contained a syntax error.  
*Explanation:* The protection setup file was not found or the file contains a syntax error.  
*User Response:* Verify the setup file exists or correct the syntax error. |
| **IMW0224E** Forbidden - URL containing ... forbidden (do not try to break in)  
*Explanation:* The URL specified is not valid. You are not allowed to specify a period (.) in the URL.  
*User Response:* Try again. |
| **IMW0226E** The redirection in the configuration file is not valid.  
*Explanation:* There is no destination for the specified redirect.  
*User Response:* Verify the configuration. |
| **IMW0227E** The user directory is not valid.  
*Explanation:* The user directory is not valid.  
*User Response:* Verify the UserDir directive in the configuration file. |
| **IMW0228E** The PUT and DELETE methods must be specified in the server's protection setup.  
*Explanation:* Methods must be explicitly specified or allowed.  
*User Response:* Correct your configuration file. |
IMW0229E The file was not found, even after searching on any extensions to the file name filename. The file does not exist or is read-protected.

Explanation: The file filename was not found.
User Response: Verify existence of the file or change its protection.

IMW0230I Document follows.
Explanation: The server is indicating the requested document follows.
User Response: None.

IMW0231I Gatewaying.
Explanation: The server is acting as a gateway.
User Response: None.

IMW0232I Found
Explanation: The redirection specified is valid.
User Response: None.

IMW0233E Access denied. Cannot specify reason. Call IBM software support.
Explanation: Server access is not allowed.
User Response: Contact the IBM Software Support Center for assistance.

IMW0234I Starting httpd
Explanation: Server initialization is in progress.
User Response: None.

IMW0235I Server is ready.
Explanation: Server initialization is complete. The server is ready to serve client requests.
User Response: None.

IMW0236E Access denied - password expired. Enter old_password/new_password/new_password to change your password.
Explanation: Your password has expired.
User Response: Enter your old password and your new password twice.

IMW0237E Password changed. Enter newpw to continue.
Explanation: Your password has been changed.
User Response: None.

IMW0238E New passwords are not equal, try again. Enter old_password/new_password/new_password to change your password.
Explanation: The two new passwords you entered are not the same.
User Response: The two new passwords must be the same. Try again.

IMW0239E New password has invalid format, try again. Enter old_password/new_password/new_password to change your password.
Explanation: The new password you entered is incorrect.
User Response: Try again.

IMW0240E Access denied. Unauthorized program loaded.
Explanation: You are running in an authorized environment, for example, BPX.DAEMON. This environment has been corrupted.
User Response: Determine how the environment has been corrupted and correct the problem.

IMW0241E Access denied - surrogate user setup error.
Explanation: An access control user ID is required.
User Response: Specify a valid access control user ID or surrogate user ID.

IMW0242E Access denied - system error using SAF.
Explanation: The user ID and password did not pass authentication.
User Response: Verify that the user ID and password are valid for this system.

IMW0243E Configuration file name not found or in error. The server cannot start without a valid configuration file.
Explanation: The server was either unable to find the specified configuration file, or the configuration file contained errors.
User Response: You might have entered the name
IMWHTTPD Messages

incorrectly, or the configuration file may contain errors. Try again using a different name or check the file for errors.

IMW0244E  Too many redirection-on-the-fly hops, max 10 (probably looping)
Explanation: More than 10 redirections have been issued; to avoid an infinite loop, the request processing has been ended.
User Response: Contact the server administrator.

IMW0245E  Unable to PUT file: filename filename
Explanation: The attempted PUT operation has failed.
User Response: Contact the server administrator.

IMW0246E  Unable to DELETE file: filename filename
Explanation: The attempted DELETE operation has failed.
User Response: Contact the server administrator.

IMW0247E  Transfer-Encoding transfer_encoding is not implemented.
Explanation: An unknown transfer-encoding was applied to the body of the request.
User Response: Contact the client or proxy vendor.

IMW0248E  Length Required.
Explanation: An HTTP/1.1 protocol violation has occurred. A PUT request was received and there was no transfer-encoding was applied. No content-length was specified. The server cannot determine the object size.
User Response: Contact the client or proxy vendor.

IMW0249E  Input timer expired while serving client for request
Explanation: The request was not received from the client in the time specified on the InputTimeout directive.
User Response: Try again.

IMW0250E  Output timer expired while serving client for request
Explanation: The request was not completed (sent to the client) in the time specified on the OutputTimeout directive.
User Response: Try again.

IMW0251E  Script timer expired while serving client for request
Explanation: The CGI script failed to complete in the time specified on the ScriptTimeout directive.
User Response: Contact the script author or server administrator.

IMW0252E  Timer of unspecified origin expired while serving client for request
Explanation: An internal timer expired.
User Response: None.

IMW0253E  Redirection. This document can be found elsewhere. You see this message because your browser does not support automatic redirection handling.
Explanation: This page is issued when a redirection occurs for the benefit of those using browsers that do not understand redirection.
User Response: Follow the link.

IMW0254E  Error
Explanation: This is the template for server generated error messages.
User Response: After you read the error, act accordingly.

IMW0255E  Server mapping error. The server is misconfigured.
Explanation: The server cannot process the request because a value that it needs to proceed with the processing is not set up. This can be caused by a configuration error, such as a Pass directive matching a POST request. If you do not find a configuration error, this message may indicate a defect in the server.
User Response: Correct the configuration error.
If you cannot find the error, contact the IBM Software Support Center for assistance.

IMW0256E  Can’t open temporary PICS Configuration file: filename (original was filename)
Explanation: The Web server address space was unable to open a temporary file previously created by the Web server daemon address space. Both the temporary file name and the original file names are provided in this message. Server processing has stopped. This is an internal error.
User Response: Contact the IBM Software Support Center for assistance.
**IMW0257E** Can’t open temporary Config

**Configuration file: filename (original was filename), errno=return_code, errno2=return_code.**

**Explanation:** The Web server address space was unable to open a temporary file previously created by the Web server daemon address space. Both the temporary file name and the original file names are provided in this message. Server processing has stopped. This is an internal error.

The OS/390 UNIX System Services return_code and reason_code provide information about the cause of the problem. See “Explanation of errno and errno2 codes in messages” on page 423.

**User Response:** Contact the IBM Software Support Center for assistance.

---

**IMW0258E** Cannot open temporary protection setup file filename

**Explanation:** The Web server address space was unable to open a temporary file previously created by the Web server daemon address space. The temporary protection setup file name is provided in this message. Server processing has stopped. This is an internal error.

**User Response:** Contact the IBM Software Support Center for assistance.

---

**IMW0259E** Original file name filename

**Explanation:** The Web server address space was unable to open a temporary file previously created by the Web server daemon address space. This message gives the original protection file name (as it existed when the original configuration file was read). A previous message gave the temporary file name that could not be opened. The server address space has stopped. This is an internal error.

**User Response:** Contact the IBM Software Support Center for assistance.

---

**IMW0260E** Cannot access configuration data, errno=return_code

**Explanation:** The Web server address space was unable to access configuration data that is created and maintained in shared memory by a daemon address space. The server address space has stopped. This is an internal error.

The OS/390 UNIX System Services return_code provides information about the cause of the problem. Message IMW0261E or IMW0264E may also be issued with an OS/390 UNIX System Services reason_code that can provide additional information. See “Explanation of errno and errno2 codes in messages” on page 423.

**User Response:** Contact the IBM Software Support Center for assistance.

---

**IMW0261I** errno2=return_code

**Explanation:** This message provides additional, OS/390 UNIX-specific error information associated with a previous message.

The OS/390 UNIX System Services reason_code provides information about the cause of the problem. For more information, see “Explanation of errno and errno2 codes in messages” on page 423.

**User Response:** If necessary, contact the IBM Software Support Center for assistance.

---

**IMW0262E** WorkQueue... Unable to create configuration set, terminating.

**Explanation:** The daemon address space was unable to create a configuration set snapshot. Previous messages indicate the cause of the error.

**User Response:** If necessary, contact the IBM Software Support Center for assistance.

---

**IMW0263E** Cannot create shared memory, errno=return_code

**Explanation:** The daemon or server address space was unable to create a shared memory segment. The address space has stopped.

The OS/390 UNIX System Services return_code provides information about the cause of the problem. Message IMW0261E or IMW0264E may also be issued with an OS/390 UNIX System Services reason_code that can provide additional information. See “Explanation of errno and errno2 codes in messages” on page 423.

**User Response:** If necessary, contact the IBM Software Support Center for assistance.

---

**IMW0264E** errno2=return_code

**Explanation:** This message provides additional, OS/390 UNIX-specific error information associated with a previous message.

The OS/390 UNIX System Services reason_code provides information about the cause of the problem. For more information, see “Explanation of errno and errno2 codes in messages” on page 423.

**User Response:** If necessary, contact the IBM Software Support Center for assistance.

---

**IMW0265E** Cannot open temporary file: filename

**Explanation:** When running with workload management, the daemon address space creates temporary files that reflect a snapshot of the configuration data as it existed when the daemon started. This file was successfully created, but could not be opened for writing. The daemon address space has stopped.
**IMWHTTPD Messages**

**User Response**: Determine the reason that the temporary file could not be opened for writing.

**IMW0266E** Cannot read configuration file: *filename*

**Explanation**: When running with workload management, the daemon address space creates temporary files that reflect a snapshot of the configuration data as it existed when the daemon started. Either the original configuration file could not be read, or the temporary file could not be written.

**User Response**: Determine the reason that the configuration file could not be read or the temporary file could not be written.

**IMW0267E** Request not valid -- HOST header was not sent

**Explanation**: This message indicates an HTTP/1.1 protocol violation. The Web server expected a host header that was not sent. This is a browser problem.

**User Response**: Contact the vendor of the browser.

**IMW0268E** Precondition failed: request has been modified.

**Explanation**: The object requested has been modified since the time specified in the If-Unmodified-Since header. You should not receive this message.

**User Response**: Contact the vendor of the browser.

**IMW0269E** Host not found or not responding

**Explanation**: An attempt was made to connect to the destination server that the client requested and failed.

**User Response**: Try again later.

**IMW0270E** Precondition failed: Could not match entity tags

**Explanation**: The version of the object does not match the version requested. You should not receive this message.

**User Response**: Contact the vendor of the browser.

**IMW0271I** command -d level -h hostname -c community -shm

**Explanation**: This is a usage message for DPI run in test mode as a standalone process.

**User Response**: Try again after you correct the command syntax.

**IMW0273I** Where: -d level -debug level, default level=number

**Explanation**: This is a usage message for DPI run in test mode as a standalone process.

**User Response**: Try again after you correct the command syntax.

**IMW0274I** -h host -send request to specified host

**Explanation**: This is a usage message for DPI run in test mode as a standalone process.

**User Response**: Try again after you correct the command syntax.

**IMW0275I** -c community -use specified community name

**Explanation**: This is a usage message for DPI run in test mode as a standalone process.

**User Response**: Try again after you correct the command syntax.

**IMW0276I** -shm -connect over shared memory, not

**Explanation**: This is a usage message for DPI run in test mode as a standalone process.

**User Response**: Try again after you correct the command syntax.

**IMW0277I** Defaults: command -d 0 -h host -c community

**Explanation**: This is a usage message for DPI run in test mode as a standalone process.

**User Response**: Try again after you correct the command syntax.

**IMW0278I** command -command

**Explanation**: This is a usage message for DPI run in test mode as a standalone process.

**User Response**: Try again after you correct the command syntax.
IMW0279E DPI open failed with return code
\texttt{return\_code}

\texttt{errno=OS/390\_UNIX\_return\_code,}
\texttt{errno2=OS/390\_UNIX\_reason\_code.}

\textbf{Explanation:} The do_connect_open function failed. The Web server SNMP subagent cannot establish a connection with SNMP. SNMP support will not be successful.

The OS/390 UNIX System Services \texttt{return\_code} and \texttt{reason\_code} may provide additional information about the cause of the problem. See “Explanation of errno and errno2 codes in messages” on page 423.

\textbf{User Response:} Make sure:
\begin{itemize}
  \item The SNMPD or DPID2 processes are running.
  \item The SNMP agent you are running is DPI capable and running with the DPI support on.
  \item The community name in the Web server configuration file permits SNMP to access this system.
\end{itemize}

If necessary, contact the IBM Software Support Center for assistance.

IMW0280E DPI do_register() failed with return code
\texttt{return\_code}

\texttt{errno=OS/390\_UNIX\_return\_code,}
\texttt{errno2=OS/390\_UNIX\_reason\_code.}

\textbf{Explanation:} The registration of the server SNMP MIB with the SNMP agent failed. SNMP support will not be successful.

The OS/390 UNIX System Services \texttt{return\_code} and \texttt{reason\_code} may provide additional information about the cause of the problem. See “Explanation of errno and errno2 codes in messages” on page 423.

\textbf{User Response:} Make sure the SNMP agent process is running.

If necessary, contact the IBM Software Support Center for assistance.

IMW0281E DPI dpi_process_request() failed with return code
\texttt{return\_code}

\texttt{errno=OS/390\_UNIX\_return\_code,}
\texttt{errno2=OS/390\_UNIX\_reason\_code.}

\textbf{Explanation:} The server SNMP subagent could not process the DPI request. SNMP support will not be successful.

The OS/390 UNIX System Services \texttt{return\_code} and \texttt{reason\_code} may provide additional information about the cause of the problem. See “Explanation of errno and errno2 codes in messages” on page 423.

\textbf{User Response:} Make sure the SNMP agent is running.

IMW0282E Problem getting a new DPI packet.
\texttt{return\_code len=number}

\texttt{errno=OS/390\_UNIX\_return\_code,}
\texttt{errno2=OS/390\_UNIX\_reason\_code}

\textbf{Explanation:} Retreiving a DPI packet using select, read, or DPIawait_packet_from_agent has failed with a bad return code, or has returned a DPI packet with a bad length. This DPI request processing has failed, but SNMP support will continue. The connection to the SNMP agent may be terminated.

The OS/390 UNIX System Services \texttt{return\_code} and \texttt{reason\_code} may provide additional information about the cause of the problem. See “Explanation of errno and errno2 codes in messages” on page 423.

\textbf{User Response:} Make sure the SNMP agent is running. The SNMP subagent attempts to reestablish the connection automatically.

If necessary, contact the IBM Software Support Center for assistance.

IMW0283E DPI cannot parse packet from agent

\textbf{Explanation:} The server SNMP subagent received a bad DPI packet. The parser DPIpacket returned a null header. SNMP support will continue if this failure happens after connect, open, and registration. If this fails during connect, open, or MIB registration to the SNMP agent, then SNMP subagent support will not continue.

\textbf{User Response:} Make sure the SNMP agent is running. If necessary, contact the IBM Software Support Center for assistance.

IMW0284E Unexpected DPI packet type \texttt{type}

\textbf{Explanation:} Received a DPI packet that could be parsed, but the request type was not valid. The fulfillment of this request failed, but SNMP subagent support will continue.

\textbf{User Response:} If necessary, contact the IBM Software Support Center for assistance.

IMW0285E Processing the DPI request had an unexpected result, \texttt{rc=return\_code.}

\textbf{Explanation:} The DPI request type is valid, but fulfilling that request produced an error. This request fulfillment failed, but SNMP subagent processing will continue.

\textbf{User Response:} If necessary, contact the IBM Software Support Center for assistance.
**IMW0286E**  
DPI connect failed, return_code, reason_code.

**Explanation:** The connection from the Web server SNMP subagent to the SNMP agent could not be established. SNMP support will not operate.

The OS/390 UNIX System Services return_code and reason_code may provide additional information about the cause of the problem. See [Explanation of errno and errno2 codes in messages](#) on page 422.

**User Response:** Make sure the SNMP agent is running, the SNMP agent is DPI capable, and the community name in the server configuration file is valid.

If necessary, contact the IBM Software Support Center for assistance.

---

**IMW0287E**  
DPI open failed

**Explanation:** The connection from the server SNMP subagent to the SNMP agent could not be established. SNMP support will not operate.

**User Response:** Make sure the SNMP agent is running, the SNMP agent is DPI capable, and the community name in the server configuration file is valid.

If necessary, contact the IBM Software Support Center for assistance.

---

**IMW0288E**  
DPI send failed, return_code, reason_code.

**Explanation:** The open packet could not be sent to the SNMP agent. SNMP support will not operate.

The OS/390 UNIX System Services return_code and reason_code may provide additional information about the cause of the problem. See [Explanation of errno and errno2 codes in messages](#) on page 422.

**User Response:** Make sure the SNMP agent is running, the SNMP agent is DPI capable, and the community name in the server configuration file is valid.

If necessary, contact the IBM Software Support Center for assistance.

---

**IMW0289E**  
DPI had a problem opening the subagent.

**Explanation:** A response to the open packet was not received from the SNMP agent. SNMP support will not operate.

**User Response:** Make sure the SNMP agent is running, the SNMP agent is DPI capable, and the community name in the server configuration file is valid.

---

**IMW0290E**  
DPI received responses with error.

**Explanation:** The SNMP agent sent the server SNMP subagent a DPI response with an error code in it during the connect and open or register process. SNMP support will not operate.

**User Response:** Make sure the SNMP agent is running, the SNMP agent is DPI capable, and the community name in the server configuration file is valid.

If necessary, contact the IBM Software Support Center for assistance.

---

**IMW0291E**  
DPI cannot create an HTTP MIB registration packet.

**Explanation:** The connection from the Web server SNMP subagent to the SNMP agent could not be established. SNMP support will not operate.

**User Response:** Make sure the SNMP agent is running and the SNMP agent is DPI capable.

If necessary, contact the IBM Software Support Center for assistance.

---

**IMW0292E**  
DPI cannot send an HTTP MIB registration packet.

**Explanation:** SNMP subagent could not send the registration packet for the server MIB to SNMP. SNMP support will not operate.

**User Response:** Make sure the SNMP agent is running and the SNMP agent is DPI capable.

If necessary, contact the IBM Software Support Center for assistance.

---

**IMW0293E**  
DPI received no reply to an HTTP MIB registration packet.

**Explanation:** SNMP subagent did not receive a response to the registration packet for the Web server MIB from the SNMP agent. SNMP support will not operate.

**User Response:** Make sure the SNMP agent is running and the SNMP agent is DPI capable.

If necessary, contact the IBM Software Support Center for assistance.

---

**IMW0294E**  
DPI response not of RESPONSE type.

**Explanation:** The SNMP subagent expected a response type packet during connect and open or register and did not receive a packet of RESPONSE type. SNMP support will not operate.

**User Response:** If necessary, contact the IBM Software Support Center for assistance.
DPI cannot create the Application MIB register packet.

Explanation: The SNMP subagent could not create a registration packet for the Web server MIB. SNMP support will not operate.

User Response: Make sure the SNMP agent is running and the SNMP agent is DPI capable.

If necessary, contact the IBM Software Support Center for assistance.

DPI cannot send the Application MIB register packet.

Explanation: The SNMP subagent could not send the registration packet for the Web server MIB to SNMP. SNMP support will not operate.

User Response: Make sure the SNMP agent is running and the SNMP agent is DPI capable.

If necessary, contact the IBM Software Support Center for assistance.

DPI received no reply to the Application MIB register packet.

Explanation: The SNMP subagent did not receive a response to the registration packet for the Web server MIB from the SNMP agent. SNMP support will not operate.

User Response: Make sure the SNMP agent is running and the SNMP agent is DPI capable.

If necessary, contact the IBM Software Support Center for assistance.

DPI Error: For instance ???

Explanation: During GET or GET NEXT processing, the variable list from the mkDPIset API was returned as null. This GET request failed, but SNMP subagent support will continue.

User Response: If necessary, contact the IBM Software Support Center for assistance.

DPI Error: varBind_p is null

Explanation: During GET or GET NEXT processing, a variable bind structure in a list was null. This GET request failed, but SNMP subagent support will continue.

User Response: If necessary, contact the IBM Software Support Center for assistance.

DPI cannot create GET Response packet

Explanation: During GET processing, mkDPIresponse returned a null response packet. This GET request failed, but SNMP subagent support will continue.

User Response: If necessary, contact the IBM Software Support Center for assistance.

SNMP open/registration failed, going into retries

Explanation: The server’s SNMP subagent is trying to contact the SNMP master agent on the host. It has timed or an error has been received. The SNMP subagent will begin trying session establishment on an increasing interval (1 minute, 2 minutes, 4 minutes, 8 minutes...). SNMP MIB values for the server will not be available.

User Response: Make sure the SNMP master agent is running and responding to SNMP requests.

SNMP: No local host name could be found, domain name will be used.

Explanation: The gethostbyname function could not look up the hostname, so the hostname returned by gethostname is used, which does not have a domain name attached to it.

User Response: Set the local host name on your system.

An SNMP thread is already running.

Explanation: SNMP subagent thread was running, so another one could not be started. This happens when the server is issued a restart and the SNMP request is already operational.

User Response: None.

The server is not permitted to the BPX.SMF facility class and cannot record SMP records.

Explanation: RACF permission has not been set up to allow the server to record SMF records.

User Response: Authorize the user ID that your Web server is running under to have read authority to the BPX.SMF facility class in RACF.

SMF record length was too big, Server performance record not written.

Explanation: The server performance record is too long and could not be written.

User Response: If necessary, contact the IBM Software Support Center for assistance.
IMW0306E Received unexpected error from SMF trying to write Server Performance record, record not written.

Explanation: An error was received from SMF. The error is not documented.

User Response: If necessary, contact the IBM Software Support Center for assistance.

IMW0307E SMF is not accepting Server Performance records at this time (type 103, subtype 02).

Explanation: SMF has not been enabled to record record type 103, subtype 02 and has rejected the record.

User Response: Your system programmer should:
- Change the SMF parameters so that SMF is enabled to write SMF records type 103, subtype 02
- Start SMF

IMW0308E SMF is not active, Server Performance record not written

Explanation: The SMF application is not running. The performance record does not get written.

User Response: Have your system programmer or operations center start SMF.

IMW0309E Received error from SMF trying to write Server Performance record, record not written.

Explanation: An error was received by the server from SMF. SMF is running and is accepting performance records, and the record is accurate.

User Response: If necessary, contact the IBM Software Support Center for assistance.

IMW0310E SMF record length was too big, Server Configuration record not written.

Explanation: The server configuration record is too long and could not be written.

User Response: If necessary, contact the IBM Software Support Center for assistance.

IMW0311E SMF is not accepting Server Configuration records at this time (type 103, subtype 01).

Explanation: SMF has not been enabled to record record type 103, subtype 01 and has rejected the record.

User Response: Have your system programmer change the SMF parameters so that SMF is enabled to write SMF records type 103, subtype 01.

IMW0312E SMF is not active, Server Configuration record not written.

Explanation: The SMF application is not running. The configuration record was not be written.

User Response: Have your system programmer or operations center start SMF.

IMW0313E Received error from SMF trying to write Server Configuration record, record not written.

Explanation: An error was received by the server from SMF. SMF is running and is accepting performance records, and the record is accurate.

User Response: If necessary, contact the IBM Software Support Center for assistance.

IMW0314E Received unexpected error from SMF trying to write Server Configuration record, record not written.

Explanation: An error was received from SMF that is not documented.

User Response: If necessary, contact the IBM Software Support Center for assistance.

IMW0315E WorkQueue... initialization failed, terminating.

Explanation: A _server_init() request to workload management was made and failed.

User Response: Determine the reason for this error:
- Use _errno2() to obtain the workload management service reason for the failure.
- The calling thread’s address space is not permitted to the BPX.WLMSERVER facility class.

IMW0316I WorkQueue... connecting to WLM as queue_manager or queue_server, SN: subsystem_name, AE: application_environment, EU: execution_units

Explanation: This message indicates which address space is being connected to workload management and relevant information about queue manager and queue server.

User Response: None.

IMW0317E Invalid request -- completely unable to parse it

Explanation: A request line was sent to the parser and it is not understood. A 400 message is returned to the server.
IMW0318E  Error: Parameter *parameter* specified more than once

**Explanation:** A parameter has been specified more than once on the passed parameters. -SN and -AE must be specified only once per workload management address space.

**User Response:** Correct your parameters and resubmit the job.

---

IMW0319E  Missing value for *parameters* parameters

**Explanation:** The value for the specified parameter is missing (-SN or -AE).

**User Response:** Be sure you are using the appropriate syntax, update your job, and resubmit it.

---

IMW0320E  Parameter conflict. Only -SN is allowed with -AE

**Explanation:** Additional parameters were found when trying to verify the queue server address space. The only ones accepted are -SN and -AE.

**User Response:** Be sure you are using the appropriate syntax, update your job, and resubmit it.

---

IMW0321E  Error: Parameter -SN required with -AE

**Explanation:** A subsystem name is required with the ApplEnv parameter.

**User Response:** Be sure you are using the appropriate syntax, update your job, and resubmit it.

---

IMW0322E  Connection established

**Explanation:** The client has been notified that a connection was established.

**User Response:** None.

---

IMW0323E  Application Environment currently not available

**Explanation:** The ApplEnv directive for a URL does not specify a correctly configured WLM Application Environment.

**User Response:** Ensure that the ApplEnv directive for this URL specifies a correctly configured WLM Application Environment, and check the status of the WLM Application Environment associated with this request.

For additional information, check the error log for the OS/390 UNIX System Services errno (return code) and errno2 (reason code) related to _server_pwu() or WorkQueue operations. If you do not see these codes in the error log, start the Web server with the -vv trace option and examine the trace output to find the return and reason codes associated with the failure. See

---

IMW0324E  Expectation failed

**Explanation:** A client browser sent an expect header that could not be satisfied by the HTTP server. The server returns a 417 Failed Response message if the expect header has data in it that wasn’t recognized by the server if the server doesn’t support expect headers.

**User Response:** None

---

IMW0325E  Accept headers not matched

**Explanation:** This message, which is written to the log file, is equivalent to IMW0326E.

**User Response:** None

---

IMW0326E  Not Acceptable - No file exists which can satisfy the accept headers sent with the request.

**Explanation:** Client accept headers can identify various file characteristics (for example: mime-type, language, encoding type, compression, character set, or user agent). The browser requested a file with specific characteristics and no file with those characteristics was found on the server.

**User Response:** Ensure the client and server are properly configured. For example, requesting a file based on language or character sets, make sure the client and browser are configured appropriately. If accompanied by a 406 response, the file was found on the server, but it did not match one or more of the criteria specified in the accept header.

---

IMW0329E  Completion code from REXX exec *exec_name* is completion_code.

**Explanation:** When a GWAPI REXX executable program exits, it must set a numeric completion code with the EXIT instruction. This message is issued when the exec completion code is outside the acceptable HTTP return code range of -1 to 599.

**User Response:** Correct the error.

---

IMW0330E  Error error_code attempting to execute *exec_name*.

**Explanation:** A problem occurred attempting to invoke the GWAPI REXX executable program *exec_name*.

error_code is one of the following values:

-99  A storage request failed.

-98  Necessary information, such as a path name or exec name, could not be extracted from the server variables.
IMWHTTPD Messages

-97 Problems occurred processing the executable program file. Possible problems include non-existent file, improper permission, or no free file descriptor.

-96 The executable program contained more than 16,384 lines.

-95 BPXWRBLD service which creates OS/390 REXX environment failed.

User Response: Correct the error.

IMW0332E Can't obtain shared memory for connection pool, errno=return_code.

Explanation: The server cannot obtain the requested shared memory.

The OS/390 UNIX System Services return_code provides information about the cause of the problem. See "Explanation of errno and errno2 codes in messages" on page 422.

User Response: Modify the connection pool parameters, or ask the system programmer to increase the amount of shared memory that can be allocated on the system.

IMW0333E Missing values on -ss parameter.

Explanation: The -ss parameter was specified without the necessary number of additional arguments.

User Response: Restart the server specifying the -ss parameter with the correct number of arguments. The format of the -ss parameter is: -ss CONNECTION_POOL start_address size.

IMW0334E More than one set of values specified for CONNECTION_POOL. Last set used

Explanation: The -ss parameter was specified with multiple CONNECTION_POOLS. This is not currently supported. The last set CONNECTION_POOL argument was used.

User Response: The server was started using the last CONNECTION_POOL argument specified. If incorrect, stop the server, then restart the server using the correct parameters.

IMW0335I Missing ":"). Incomplete ApplEnvConfig statement.

Explanation: The end of the configuration file was reached without finding the ending ""):" of the ApplEnvConfig directive.

User Response: Correct the configuration file.

The correct syntax is:

ApplEnvConfig AEName {
    directive
    directive
}

IMW0336I Multiple conflicting PluginDefault statements

Explanation: There were conflicting PluginDefault statements in the configuration file. PluginDefault Include and PluginDefault Exclude cannot both be specified at the same level.

User Response: Correct the configuration file.

IMW0337I Multiple conflicting Plugin directives for this DLL: string

Explanation: There were conflicting PluginInclude and PluginExclude statements in the configuration file for the same DLL at the same level. PluginInclude and PluginExclude cannot both be specified at the same level for the same DLL.

User Response: Correct the configuration file.

IMW0338I Invalid parameter to PluginDefault directive: string

Explanation: An invalid value was specified on the PluginDefault Directive.

User Response: Correct the configuration file. Valid values are Include and Exclude.

IMW0339I Syntax error in configuration file, expecting '{' to start ApplEnvConfig statement for string

Explanation: The ApplEnvConfig directive in the configuration file is not valid.

User Response: Correct the configuration file. The correct syntax is:

ApplEnvConfig AEName {
    directive
directive
}

IMW0340I Active threads reset from number to system limit of number

Explanation: The number of threads specified in the configuration file exceeded the maximum number of threads per process allowed by the system.

User Response: To eliminate this warning message, reduce the maximum number of threads in the configuration file to less than the limit imposed by the operating system.
Value for EnableFRCA is not valid: value.

Explanation: The value specified for the EnableFRCA directive is not valid. This directive is used to turn the Fast Response Cache Accelerator dynamic caching function on or off.

For more information, see "EnableFRCA — Turn dynamic caching on or off" on page 383.

User Response: Correct the configuration file.

FRCAStackName is not valid: name.

Explanation: The TCP/IP stack name specified on the FRCAStackName directive is not valid.

For more information, see "FRCAStackName — Specify the TCP/IP stack that supports the dynamic cache" on page 388.

User Response: Correct the configuration file.

FRCAWLMParms parameter is not valid: parameter.

Explanation: The parameter specified on the FRCAWLMParms directive is not valid. This directive is used to specify the unique subsystem name, application environment name, and transaction class that will be used to classify the work performed by the Fast Response Cache Accelerator under Workload Management (WLM).

For more information, see "FRCAWLMParms — Specify parameters for Workload Management" on page 389.

User Response: Correct the configuration file.

Both FRCACacheOnly and FRCANoCaching cannot be used.

Explanation: You must use either the FRCACacheOnly and FRCANoCaching directive.

For more information, see "FRCACacheOnly — Specify URLs to be dynamically cached" on page 388 and "FRCANoCaching — Exclude URLs from the dynamic cache" on page 387.

User Response: Correct the configuration file.

Fast Response Cache Accelerator did not initialize.

Explanation: This message can occur as a result of system errors or user setup errors.

User Response: Check the error log for the OS/390 UNIX System Services errno (return code) and ermm2 (reason code) related to _server_pwu() or WorkQueue operations. If you do not see these codes in the error log, start the Web server with the -vv trace option and examine the trace output to find the return and reason codes associated with the failure. See "Explanation of errno and ermm2 codes in messages" on page 422.

Debug Tool start had failure code: failure_code.

Explanation: While attempting to start the OS/390 Debug Tool, a failure occurred that prevented the tool from starting. Message IMW0347E is also issued and shows the GWAPI plugin module name.

The failure code is usually one of the following values:

- CEE000: Non-zero return code from CEE3CBTS or Dflow_allocation failure
- CEE2F2: The Debug Tool is not available.
- CEE2F7: Profiler loaded, but Debug Tool is not available.

User Response: Verify that the Remote Debugger workstation has been started. If the Remote Debugger workstation is up and running, verify that the DebugToolAddr directive in your httpd.conf file specifies the correct address and port number for the Remote Debugger workstation.

Related information:
- "DebugToolAddr - Identify the workstation running the Remote Debugger" on page 312
- "Debugging C/C++ GWAPI programs" on page 226

start_dbg() could not start OS/390 Debug Tool for GWAPI_plugin_module_name.

Explanation: While attempting to start the OS/390 Debug Tool, a failure occurred that prevented the tool from starting. Message IMW0346E is also issued and provides information on the cause of the problem.

User Response: See the description of message IMW0346E.

DebugToolAddr directive had gethostbyname error.

Explanation: During configuration processing, the host name for the Remote Debugger workstation could not be resolved.

User Response: Verify that the host name specified on the DebugToolAddr directive is correct and that the host name is recognized by your Domain Name Server. You can specify the IP address of the Remote Debugger instead of the host name on this directive.

For more information, see "DebugToolAddr - Identify the workstation running the Remote Debugger" on page 313.
### IMWHTTPD Messages

<table>
<thead>
<tr>
<th>Message Code</th>
<th>Description</th>
<th>Explanation</th>
<th>User Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMW0349E</td>
<td>DebugToolAddr directive had PORT parameter error.</td>
<td>The port number value specified for the OS/390 Debug Tool Remote Debugger workstation is not valid. The port number must be an integer containing numbers 0-9.</td>
<td>Correct the port number on the DebugToolAddr directive. For more information, see [DebugToolAddr - Identify the workstation running the Remote Debugger](page 312) on page 312.</td>
</tr>
<tr>
<td>IMW0350I</td>
<td>Storage allocation error for InternalTraceLog. Using default size.</td>
<td>The amount of storage requested for the Web server Internal Trace on the InternalTraceLog directive cannot be allocated. The Web server will attempt to use the default Internal Trace log size.</td>
<td>For guidance and additional information on Web server traces, contact the IBM Software Support Center.</td>
</tr>
<tr>
<td>IMW0351E</td>
<td>Unable to allocate number KB. No InternalTraceLog available.</td>
<td>This error occurs when storage cannot be allocated for the Web server Internal Trace log. This message usually indicates that storage for the default Internal Trace size could not be allocated. Message IMW0350I is also issued.</td>
<td>You may need to increase the region size. See Message IMW0350I for additional information.</td>
</tr>
<tr>
<td>IMW0352E</td>
<td>InternalTraceSize error - invalid characters in size.</td>
<td>This error occurs when the InternalTraceSize directive contains characters that are not valid. The trace size value must be an integer containing numbers 0-9.</td>
<td>For guidance and additional information on Web server traces, contact the IBM Software Support Center.</td>
</tr>
<tr>
<td>IMW0353E</td>
<td>Failure waiting for I/O completion, errno = return_code.</td>
<td>A signal timed wait for a previously issued asynchronous I/O request failed. As a result of this condition, I/O from the client may be lost, and the connection will eventually be closed when the proper timeout (input, output, or persistent) is reached.</td>
<td>Correct content-range settings and resubmit request.</td>
</tr>
<tr>
<td>IMW0354E</td>
<td>Immediate failure on asynch read operation for socket socket_number, errno = return_code.</td>
<td>This failure indicates a system resource constraint or an error in the Web server. It occurs when an asynchronous I/O request to read data on socket_number fails. This failure occurs immediately on the issuance of the asynchronous I/O.</td>
<td>If necessary, contact the IBM Software Support Center for assistance.</td>
</tr>
<tr>
<td>IMW0355E</td>
<td>Immediate failure on asynch write operation for socket socket_number, errno = return_code.</td>
<td>This failure indicates a system resource constraint or an error in the Web server. It occurs when an asynchronous I/O request to write data on socket_number fails. This failure occurs immediately, that is, on the issuance of the asynchronous I/O.</td>
<td>If necessary, contact the IBM Software Support Center for assistance.</td>
</tr>
<tr>
<td>IMW0356E</td>
<td>Data in authorization header not valid: string.</td>
<td>Browser credentials need to be updated.</td>
<td>Start a new browser and enter a user ID and password with proper credentials.</td>
</tr>
<tr>
<td>IMW0357E</td>
<td>Data in content-range header not valid for content-range string.</td>
<td>The current content-range settings need to be changed.</td>
<td>Correct content-range settings and resubmit request.</td>
</tr>
</tbody>
</table>
IMW0358E Unknown expect header: string
Explanation: The only supported expect header is 100-CONTINUE.
User Response: Correct the expect header or contact the browser vendor.

IMW0359E Unknown pragma header: string
Explanation: The only supported pragma header is NO-CACHE.
User Response: Correct the pragma header or contact the browser vendor.

IMW0360E Data in proxy authorization header not valid: string.
Explanation: Browser credentials need to be updated.
User Response: Start a new browser and enter a user ID and password with proper credentials.

IMW0361E Unknown or unexpected header: string.
Explanation: The header is not recognized by the Web server.
User Response: Correct the header or contact the browser vendor.

IMW0362E FastCGI error: failed to open configuration_file, errno=return_code, errno2=reason_code.
Explanation: The FastCGI configuration file could not be opened during FastCGI initialization.
The OS/390 UNIX System Services return_code and reason_code may provide additional information about the cause of the problem. See “Explanation of errno and errno2 codes in messages” on page 422.
User Response: If necessary, contact the IBM Software Support Center for assistance.

IMW0363E FastCGI initialization failed: no FastCGI applications defined.
Explanation: No Local or External FastCGI applications are defined in the FastCGI configuration file.
User Response: Correct the FastCGI configuration file.

IMW0364E FastCGI initialization failed, errno=return_code, errno2=reason_code.
Explanation: FastCGI processes could not be created during FastCGI initialization.
The OS/390 UNIX System Services return_code and reason_code may provide additional information about the cause of the problem. See “Explanation of errno
type and errno2 codes in messages” on page 422.

IMW0365E FastCGI initialization failed to establish signal handler for signal_number, errno=return_code, errno2=reason_code.
Explanation: A signal handler could not be created during FastCGI initialization.
The OS/390 UNIX System Services return_code and reason_code may provide additional information about the cause of the problem. See “Explanation of errno and errno2 codes in messages” on page 422.
User Response: If necessary, contact the IBM Software Support Center for assistance.

IMW0366E FastCGI error: could not create socket, errno=return_code, errno2=reason_code.
Explanation: The request cannot be handled because FastCGI could not create a TCP/IP socket to connect to a FastCGI process.
The OS/390 UNIX System Services return_code and reason_code may provide additional information about the cause of the problem. See “Explanation of errno and errno2 codes in messages” on page 422.
User Response: If necessary, contact the IBM Software Support Center for assistance.

IMW0367E FastCGI error: could not set socket option option, errno=return_code, errno2=reason_code.
Explanation: FastCGI could not set the indicated TCP/IP socket option.
The OS/390 UNIX System Services return_code and reason_code may provide additional information about the cause of the problem. See “Explanation of errno and errno2 codes in messages” on page 422.
User Response: If necessary, contact the IBM Software Support Center for assistance.

IMW0368E FastCGI error: failed to connect to FastCGI process, errno=return_code, errno2=reason_code.
Explanation: This error usually occurs when you attempt to run a FastCGI program and have logged into the browser with the same user ID used to start the Web server. In addition to this message, you may get a browser Error 503.
The OS/390 UNIX System Services return_code and reason_code may provide additional information about the cause of the problem. See “Explanation of errno
and errno2 codes in messages” on page 422.
User Response: If necessary, contact the IBM Software Support Center for assistance.
IMWHTTPD Messages

**User Response:** Log into the browser with a user ID that is different from the user ID you used to start the Web server.

If necessary, contact the IBM Software Support Center for assistance.

**IMW0371E** FastCGI error: write request failed,
\[errno=return_code, \texttt{errno2=reason\_code}\].

**Explanation:** FastCGI cannot write the request to the FastCGI process.

The OS/390 UNIX System Services return\_code and reason\_code may provide additional information about the cause of the problem. See "Explanation oferrno and errno2 codes in messages" on page 422.

**User Response:** If necessary, contact the IBM Software Support Center for assistance.

**IMW0372E** FastCGI error: could not retrieve Web server environment variables.

**Explanation:** FastCGI cannot retrieve environment variables from the Web server.

**User Response:** If necessary, contact the IBM Software Support Center for assistance.

**IMW0373E** FastCGI error: attempt to read request failed, content length: string.

**Explanation:** FastCGI cannot read the request body. Possible causes are:
- The Stop option was selected on the browser before a request completed.
- The request body is not valid.

**User Response:** If necessary, contact the IBM Software Support Center for assistance.

**IMW0374E** FastCGI initialization error: failed to create socket\_number socket returning 0,
\[errno=return_code, \texttt{errno2=reason\_code}\].

**Explanation:** During FastCGI initialization, communication between the FastCGI process and the Web server could not be established.

The OS/390 UNIX System Services return\_code and reason\_code may provide additional information about the cause of the problem. See "Explanation oferrno and errno2 codes in messages" on page 422.

**User Response:** If necessary, contact the IBM Software Support Center for assistance.

**IMW0375E** FastCGI initialization error: failed to bind to socket\_number socket,
\[errno=return_code, \texttt{errno2=reason\_code}\].

**Explanation:** Bind to TCP/IP socket failed during FastCGI initialization.

The OS/390 UNIX System Services return\_code and reason\_code may provide additional information about the cause of the problem. See "Explanation oferrno and errno2 codes in messages" on page 422.

**User Response:** If necessary, contact the IBM Software Support Center for assistance.

**IMW0376E** FastCGI initialization error: unlink failed on directory\_entry,
\[errno=return_code, \texttt{errno2=reason\_code}\].

**Explanation:** Unlinking the AF_UNIX directory entry failed during FastCGI initialization.

The OS/390 UNIX System Services return\_code and reason\_code may provide additional information about the cause of the problem. See "Explanation oferrno and errno2 codes in messages" on page 422.

**User Response:** If necessary, contact the IBM Software Support Center for assistance.

**IMW0377E** FastCGI initialization error: failed to set correct permission bits for directory\_entry,
\[errno=return_code, \texttt{errno2=reason\_code}\].

**Explanation:** During FastCGI initialization, correct permission bits were not set for the AF_UNIX directory entry.

The OS/390 UNIX System Services return\_code and reason\_code may provide additional information about the cause of the problem. See "Explanation oferrno and errno2 codes in messages" on page 422.

**User Response:** If necessary, contact the IBM Software Support Center for assistance.

**IMW0378E** FastCGI initialization error: failed to get host name host\_name,
\[errno=return_code, \texttt{errno2=reason\_code}\].

**Explanation:** During FastCGI initialization, host\_name could not be found.

The OS/390 UNIX System Services return\_code and reason\_code may provide additional information about the cause of the problem. See "Explanation oferrno and errno2 codes in messages" on page 422.

**User Response:** If necessary, contact the IBM Software Support Center for assistance.
IMW0379E  FastCGI error: failed to read data, errno=return_code, errno2=return_code.

Explanation: FastCGI attempted to read the response from the FastCGI application, but no data was received.

The OS/390 UNIX System Services return_code and reason_code may provide additional information about the cause of the problem. See “Explanation of errno and errno2 codes in messages” on page 422.

User Response: If necessary, contact the IBM Software Support Center for assistance.

IMW0380E  FastCGI error: failed to find matching entry for the request, errno=return_code, errno2=return_code.

Explanation: FastCGI could not find a matching entry for the request in the FastCGI configuration file.

The OS/390 UNIX System Services return_code and reason_code may provide additional information about the cause of the problem. See “Explanation of errno and errno2 codes in messages” on page 422.

User Response: Ensure that the URL directive in the FastCGI configuration file matches the FastCGI request.

If necessary, contact the IBM Software Support Center for assistance.

IMW0381E  CGI error: no script name in the request.

Explanation: The CGI script cannot be run because there is no script name in the request.

User Response: Ensure that the request contains a script name and resubmit.

IMW0382E  CGI read error: failed to read CGI script, return_code =code, errno=return_code, errno2=return_code.

Explanation: A failure occurred while reading the CGI script results. No data was read.

The OS/390 UNIX System Services return_code and reason_code may provide additional information about the cause of the problem. See “Explanation of errno and errno2 codes in messages” on page 422.

User Response: If necessary, contact the IBM Software Support Center for assistance.

IMW0383E  Internal read error: failed to read CGI script, errno=return_code, errno2=return_code.

Explanation: The fdopen() failed to open a pipe for reading CGI results. No data was read.

The OS/390 UNIX System Services return_code and reason_code may provide additional information about the cause of the problem. See “Explanation of errno and errno2 codes in messages” on page 422.

User Response: If necessary, contact the IBM Software Support Center for assistance.

IMW0384E  CGI read error: failed to read CGI script, errno=return_code, errno2=return_code.

Explanation: CGI script results cannot be read.

The OS/390 UNIX System Services return_code and reason_code may provide additional information about the cause of the problem. See “Explanation of errno and errno2 codes in messages” on page 422.

User Response: If necessary, contact the IBM Software Support Center for assistance.


Explanation: The size of a DPI packet exceeded the size of the buffer. DPI request processing has failed, but SNMP support will continue.

The OS/390 UNIX System Services return_code and reason_code may provide additional information about the cause of the problem. See “Explanation of errno and errno2 codes in messages” on page 422.

User Response: Ensure that the SNMP agent is running. The SNMP subagent automatically attempts to reestablish the connection.

If necessary, contact the IBM Software Support Center for assistance.

IMW0386E  Initialization error: Protection setup file filename not read.

Explanation: A Protect directive referenced a protection setup file that could not be opened by the Web server.

User Response: Verify:
- The protection setup file exists.
- The filename is spelled correctly in the Web server configuration file.
- The file has the correct permissions.

Correct the configuration file or protection setup file.

IMW0387E  Caching error: failed to open file filename, URL =URL, thread = number, errno = return_code.

Explanation: The RequestToCache operation failed.

The OS/390 UNIX System Services return_code provides information about the cause of the problem. See “Explanation of errno and errno2 codes in messages” on page 422.
**IMWHTTPD Messages**

**IMW0388E** Internal I/O error: unable to load file contents to the browser, \( \text{errno} = \text{return\_code}, \text{errno2} = \text{reason\_code} \).

**Explanation:** An I/O error occurred. The OS/390 UNIX System Services return_code and reason_code may provide additional information about the cause of the problem. See “Explanation of errno and errno2 codes in messages” on page 422.

**User Response:** If necessary, contact the IBM Software Support Center for assistance.

**IMW0389E** Internal HTLoadCacheToStream error: unable to load file contents from cache to the browser, \( \text{errno} = \text{return\_code}, \text{errno2} = \text{reason\_code} \).

**Explanation:** An I/O error occurred. The OS/390 UNIX System Services return_code and reason_code may provide additional information about the cause of the problem. See “Explanation of errno and errno2 codes in messages” on page 422.

**User Response:** If necessary, contact the IBM Software Support Center for assistance.

**IMW0390E** Internal HTLoadToStream error: unable to load proxy result to the browser, \( \text{errno} = \text{return\_code}, \text{errno2} = \text{reason\_code} \).

**Explanation:** An I/O error occurred. The OS/390 UNIX System Services return_code and reason_code may provide additional information about the cause of the problem. See “Explanation of errno and errno2 codes in messages” on page 422.

**User Response:** If necessary, contact the IBM Software Support Center for assistance.

**IMW0391E** Internal HTLoadToStream error: unable to load proxy result to the browser, \( \text{errno} = \text{return\_code}, \text{errno2} = \text{reason\_code} \).

**Explanation:** An I/O error occurred. The OS/390 UNIX System Services return_code and reason_code may provide additional information about the cause of the problem. See “Explanation of errno and errno2 codes in messages” on page 422.

**User Response:** If necessary, contact the IBM Software Support Center for assistance.

**IMW0392E** Internal HTLoadToStream error: unable to load proxy file to the source server, \( \text{errno} = \text{return\_code}, \text{errno2} = \text{reason\_code} \).

**Explanation:** The proxy request cannot be written to the source server. The OS/390 UNIX System Services return_code and reason_code may provide additional information about the cause of the problem. See “Explanation of errno and errno2 codes in messages” on page 422.

**User Response:** If necessary, contact the IBM Software Support Center for assistance.

**IMW0393E** CGI write error: failed to write CGI output to the browser, \( \text{errno} = \text{return\_code}, \text{errno2} = \text{ reason\_code} \).

**Explanation:** An I/O error occurred. The OS/390 UNIX System Services return_code and reason_code may provide additional information about the cause of the problem. See “Explanation of errno and errno2 codes in messages” on page 422.

**User Response:** If necessary, contact the IBM Software Support Center for assistance.

**IMW0394E** Internal HTLoadStreamFromRamCache error: unable to load file contents from cache to the browser, \( \text{errno} = \text{return\_code}, \text{errno2} = \text{reason\_code} \).

**Explanation:** An I/O error occurred. The OS/390 UNIX System Services return_code and reason_code may provide additional information about the cause of the problem. See “Explanation of errno and errno2 codes in messages” on page 422.

**User Response:** If necessary, contact the IBM Software Support Center for assistance.

**IMW0395E** Internal I/O error: unable to copy file contents to the browser, \( \text{errno} = \text{return\_code}, \text{errno2} = \text{reason\_code} \).

**Explanation:** An I/O error occurred. The OS/390 UNIX System Services return_code and reason_code may provide additional information about the cause of the problem. See “Explanation of errno and errno2 codes in messages” on page 422.

**User Response:** If necessary, contact the IBM Software Support Center for assistance.
<table>
<thead>
<tr>
<th>Message ID</th>
<th>Description</th>
<th>Explanation</th>
<th>User Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMW0396E</td>
<td>Internal HTLoadScriptResultSSI error: unable to write file contents to the browser, errno=return_code, errno2=reason_code.</td>
<td>An I/O error occurred while attempting to write the result of an SSI script to the browser. The OS/390 UNIX System Services return_code and reason_code may provide additional information about the cause of the problem. See &quot;Explanation of errno and errno2 codes in messages&quot; on page 422.</td>
<td>If necessary, contact the IBM Software Support Center for assistance.</td>
</tr>
<tr>
<td>IMW0397E</td>
<td>Caching error: failed to get statistics for filename, URL = URL, thread = number, errno = return_code.</td>
<td>The RequestToCache operation failed. The OS/390 UNIX System Services return_code provides information about the cause of the problem. See &quot;Explanation of errno and errno2 codes in messages&quot; on page 422.</td>
<td>If necessary, contact the IBM Software Support Center for assistance.</td>
</tr>
<tr>
<td>IMW0398E</td>
<td>Caching storage request failed: unable to allocate number bytes, thread = number, URL = URL, errno = return_code.</td>
<td>A request for storage failed. The OS/390 UNIX System Services return_code provides information about the cause of the problem. See &quot;Explanation oferrno and errno2 codes in messages&quot; on page 422.</td>
<td>If necessary, contact the IBM Software Support Center for assistance.</td>
</tr>
<tr>
<td>IMW0399E</td>
<td>Caching error: failed to read file filename, thread = number, errno = return_code.</td>
<td>File cannot be read. The OS/390 UNIX System Services return_code provides information about the cause of the problem. See &quot;Explanation of errno and errno2 codes in messages&quot; on page 422.</td>
<td>If necessary, contact the IBM Software Support Center for assistance.</td>
</tr>
<tr>
<td>IMW0400E</td>
<td>GWAPI HTLoadScriptResultAPI error: unable to write file contents to the browser, errno=return_code, errno2=reason_code.</td>
<td>An I/O error occurred while attempting to write the result of a script called from a GWAPI to the browser. The OS/390 UNIX System Services return_code and reason_code may provide additional information about the cause of the problem. See &quot;Explanation of errno and errno2 codes in messages&quot; on page 422.</td>
<td>If necessary, contact the IBM Software Support Center for assistance.</td>
</tr>
<tr>
<td>IMW0401E</td>
<td>GWAPI HTLoadStreamFromRamCache error: unable to write file contents from cache to the browser, errno=return_code, errno2=reason_code.</td>
<td>An I/O error occurred in the GWAPI. The OS/390 UNIX System Services return_code and reason_code may provide additional information about the cause of the problem. See &quot;Explanation of errno and errno2 codes in messages&quot; on page 422.</td>
<td>If necessary, contact the IBM Software Support Center for assistance.</td>
</tr>
<tr>
<td>IMW0402E</td>
<td>GWAPI error: unable to copy file contents to the browser, errno=return_code, errno2=reason_code.</td>
<td>An I/O error occurred in the GWAPI. The OS/390 UNIX System Services return_code and reason_code may provide additional information about the cause of the problem. See &quot;Explanation of errno and errno2 codes in messages&quot; on page 422.</td>
<td>If necessary, contact the IBM Software Support Center for assistance.</td>
</tr>
<tr>
<td>IMW0403E</td>
<td>GWAPI write error: unable to write results to the browser, errno=return_code, errno2=reason_code.</td>
<td>An I/O error occurred in the GWAPI. The OS/390 UNIX System Services return_code and reason_code may provide additional information about the cause of the problem. See &quot;Explanation of errno and errno2 codes in messages&quot; on page 422.</td>
<td>If necessary, contact the IBM Software Support Center for assistance.</td>
</tr>
</tbody>
</table>
IMWHTTPD Messages

IMW0404E GWAPI writeCP error: unable to write results to the browser, errno=return_code, errno2=reason_code.

Explanation: An I/O error occurred.

The OS/390 UNIX System Services return_code and reason_code may provide additional information about the cause of the problem. See "Explanation of errno and errno2 codes in messages" on page 422.

User Response: If necessary, contact the IBM Software Support Center for assistance.

IMW0405E Internal I/O error: unable to write file contents to the browser, errno=return_code, errno2=reason_code.

Explanation: An I/O error occurred.

The OS/390 UNIX System Services return_code and reason_code may provide additional information about the cause of the problem. See "Explanation of errno and errno2 codes in messages" on page 422.

User Response: If necessary, contact the IBM Software Support Center for assistance.

IMW0406E Failed to load file filename in Fast Response Cache Accelerator cache, thread = number, errno = return_code.

Explanation: File cannot be loaded in the cache.

The OS/390 UNIX System Services return_code provides information about the cause of the problem. See "Explanation of errno and errno2 codes in messages" on page 422.

User Response: If necessary, contact the IBM Software Support Center for assistance.

IMW0407E Failed to unload file filename from Fast Response Cache Accelerator cache, thread = number, errno = return_code.

Explanation: File cannot be unloaded from the cache.

The OS/390 UNIX System Services return_code provides information about the cause of the problem. See "Explanation of errno and errno2 codes in messages" on page 422.

User Response: If necessary, contact the IBM Software Support Center for assistance.

IMW0408E Failed to register filename in Fast Response Cache Accelerator cache, errno=return_code.

Explanation: Internal error. File cannot be registered in the cache.

The OS/390 UNIX System Services return_code provides information about the cause of the problem. See "Explanation of errno and errno2 codes in messages" on page 422.

User Response: If necessary, contact the IBM Software Support Center for assistance.

IMW0409E Fast Response Cache Accelerator did not initialize, errno=return_code, errno2=reason_code.

Explanation: Initialization failed.

The OS/390 UNIX System Services return_code and reason_code may provide additional information about the cause of the problem. See "Explanation of errno and errno2 codes in messages" on page 422.

User Response: If necessary, contact the IBM Software Support Center for assistance.

IMW0410E Internal error: global lock request failed, errno=return_code.

Explanation: An internal error occurred when the Web server attempted to remove a shared memory semaphore. A global lock is required for exclusive write access to the memory segment.

The OS/390 UNIX System Services return_code provides information about the cause of the problem. See "Explanation of errno and errno2 codes in messages" on page 422.

User Response: If necessary, contact the IBM Software Support Center for assistance.

IMW0411E Internal error: global unlock request failed, errno=return_code.

Explanation: An internal error occurred when the Web server attempted to remove a shared memory semaphore.

The OS/390 UNIX System Services return_code provides information about the cause of the problem. See "Explanation of errno and errno2 codes in messages" on page 422.

User Response: If necessary, contact the IBM Software Support Center for assistance.

IMW0412E Caching error: unable to locate file in the cache, errno=return_code, errno2=reason_code.

Explanation: File cannot be opened in the cache.

The OS/390 UNIX System Services return_code and reason_code may provide additional information about the cause of the problem. See "Explanation of errno and errno2 codes in messages" on page 422.

User Response: If necessary, contact the IBM Software Support Center for assistance.

456 HTTP Server Planning, Installing, and Using
**IMW2001E-2026E: Proxy Server Messages**

**IMW2001E** Proxy Error: Host name not recognized or host not found.

**Explanation:** This error is returned when an incorrect or unrecognizable host name is specified. The proxy server has tried to parse the host name and found the name to be not valid (for example, xyz.ibm.com).

**User Response:** Verify that the name of the destination server is correct.

**IMW2002E** Proxy Error: Unable to connect to remote host or host not responding

**Explanation:** This error is returned when the proxy server is unable to establish a socket connection with the remote host. The remote host is down or for some reason unavailable, as a network connection cannot be established.

**User Response:** Try again to establish the connection. The destination server may be temporarily unavailable.

**IMW2003E** Proxy Error: SSL Handshake with upstream server failed.

**Explanation:** This error occurs when the proxy server is unable to establish an SSL connection. To generate this error, cause a handshake failure between the secure server and secure proxy server. One way to do this is to configure disjointed certificates, that is, each side has a certificate that the other does not trust.

**User Response:** Verify that the certificates between the two servers are able to authenticate.

**IMW2004E** Proxy Error: Host connected but unable to send request. Please retry.

**Explanation:** This error occurs when the proxy server established a socket connection with the destination server, but then the destination server immediately closed the connection. This might be caused by a degenerate Web server. To generate this error, write a simple application, or modify HOWL, to accept the socket connection and then immediately drop the connection.

**User Response:** Send the request again. The destination server may be temporarily unavailable.

**IMW2005E** Proxy Error: Host connected but unable to forward request. Please retry.

**Explanation:** This error occurs when the proxy server has sent the request headers, but the connection with the destination server is lost before the proxy server can send the content body. This scenario is easy to create for test purposes. Try sending the request header followed by a huge content body (for example 64KB) on a POST request, and then closing the connection after the request headers have been read, but before the content body has been received by the destination server. Use a debugger to get the timing right.

**User Response:** Resend the request. The remote server may be temporarily unavailable.

**IMW2006E** Proxy Error: Remote host did not send any data.

**Explanation:** The destination server did not respond to the proxy server's request. In most cases, the destination server or network connection went down before the server could respond.

**User Response:** Retry the request.

**IMW2007E** Invalid forced expiry time spec: string.

**Explanation:** The syntax for the CacheMinHold directive in the javelin configuration file is incorrect or was not specified.

**User Response:** Correct the error. See the IBM Web Traffic Express for Multiplatforms User's Guide V1.0 for the correct syntax for the CacheMinHold directive.

**IMW2008E** Javelin..... sox.conf could not be opened.

**Explanation:** Unable to open the socks configuration file (socks.conf). This file is in the /etc directory.

**User Response:** Ensure that the file exists and check the file permissions.

**IMW2009E** PICS Rule starting at line number could not be parsed.

**Explanation:** There is an error in the PICS Rule.

**User Response:** Check the syntax of the PICS Rule in javelin configuration file (for example, javelin.conf).

**IMW2010E** PICS Rule starting at line number could not be parsed. The error was detected near the text string.

**Explanation:** There is an error in the PICS Rule.

**User Response:** Check the syntax of the PICS Rule in the javelin.conf configuration file.

**IMW2012E** Cannot open configuration file: filename.

**Explanation:** The server was unable to open the javelin configuration file.

**User Response:** Check for the existence and permissions of the javelin configuration file.
Proxy Server messages

**IMW2013E** Unknown Javelin directive *directive*.

**Explanation:** The specified directive from the javelin configuration file is unrecognized.

**User Response:** Remove this directive from the javelin configuration file.

**IMW2014E** Too many arguments on one line in configuration file, max: *number*

**Explanation:** There were too many arguments on one line.

**User Response:** Change the number of arguments on the line for the directive.

**IMW2015E** Insufficient parameters for directive: *name*

**Explanation:** There were not enough arguments on the line.

**User Response:** Change the number of arguments on the line for the directive.

**IMW2016E** Invalid Number of subcaches: *string*.

**Explanation:** A negative number was specified for the ProxyNumTables directive in the javelin configuration file.

**User Response:** The number of subcaches must be equal to or larger than zero.

**IMW2017E** PICS rule group file couldn’t be read.

**Explanation:** The server could not read the referenced PICS rule containing the group file subdirective. Either the file does not exist or it contains unreadable information.

**User Response:** Change the referenced PICS rule to point to a valid file or ensure the file contains valid information.

**IMW2018E** PICS rule applies-to couldn’t be read.

**Explanation:** The referenced PICS rule contains an applies-to subdirective that is not valid.

**User Response:** Correct the PICS rule by entering a valid applies-to subdirective.

**IMW2019E** Cannot run cache agent because proxy server is not specified.

**Explanation:** On UNIX systems especially, the Proxy directive in the javelin configuration file must be specified. Otherwise, the server does not know whose cache to fill.

**User Response:** Specify the host and port for the proxy server whose cache should be filled in the Proxy directive in the javelin configuration file. Syntax: Proxy

**IMW2020E** NumClients too large. Downgraded number of cache refresh clients to 100.

**Explanation:** NumClients exceeded 100 and was decreased to the limit of 100 threads. This is to minimize network use and stress on the content servers that will be contacted.

**User Response:** Decrease the number of threads for the cache agent. See the NumClients directive in the javelin configuration file.

**IMW2021E** Could not open the cache agent configuration file *filename*.

**Explanation:** Unable to open the javelin configuration file.

**User Response:** Check the existence and the permissions of the javelin configuration file.

**IMW2022E** Not able to load files specified in cache log, check proxy server configuration.

**Explanation:** The cache agent was unable to load files that were specified in the cache access log. This can be caused by lack of a cache access log, inability to locate the server configuration file, or having caching set off.

**User Response:** Check the configuration files and system for the three symptoms listed above and correct it, as appropriate.

**IMW2023E** The proxy server is not configured for caching, specify CachingOn.

**Explanation:** In the server configuration file, Caching is set off.

**User Response:** Set the Caching directive in the server configuration on.

**IMW2024E** Proxy server did not specify CacheAccessLog directive.

**Explanation:** The CacheAccessLog directive wasn’t set by the server.

**User Response:** Set the CacheAccessLog in the server configuration file.

**IMW2025E** Could not open the proxy server’s CacheAccessLog file *filename*.

**Explanation:** The proxy server’s cache access log could not be opened.

**User Response:** Check that the CacheAccessLog directive is set. Check for the existence and permissions of the corresponding log file.
**Proxy Server messages**

**IMW2026E**  Forbidden - not allowed

**Explanation:** The user has issued a request which is being proxied to the origin server using reverse proxy. Access to the proxy server is protected at the proxy server, and the file being requested from the origin server is protected at the origin server. Protection at both the proxy server and the origin server at the same time is not supported in the reverse proxy environment.

**User Response:** To access the file using reverse proxy, remove the protection from either the proxy server or from the file at the origin server. If both access to the proxy server and access to the file at the origin server must be protected, then reverse proxy cannot be used to access the file. In that case, the file must be accessed using the proxy server as a normal proxy server and not a reverse proxy server.

**IMW3501I-3542E: CONSOLE Messages**

**Explanation of process descriptor in messages**

Several CONSOLE messages contain a `process_descriptor` variable, which includes `mode processid ServerIP:ServerPort serverSN serverAE`:

- **mode**
  - is one of:
    - `SA` - the server started as a standalone process (not managed by WLM)
    - `QM` - the server started as a WLM Queue Manager
    - `QS` - the server started as a WLM Queue Server

- **processid**
  - is the server’s process identifier as assigned by OS/390 UNIX System Services

- **ServerIP**
  - is the server’s IP address, in the form n.n.n.n, where n ranges from 0 to 255. If `ServerIP` is 0.0.0.0, then the server is not bound to any specific IP address (if neither the BindSpecific directive nor the -HN start parameter was used).

- **ServerPort**
  - is the primary port number that the server is listening on.

- **serverSN**
  - is the WLM subsystem name. If `mode` indicates SA, then `serverSN` will be ‘*’. Otherwise, it will reflect the value passed as the -SN start parameter.

- **serverAE**
  - is the WLM Application Environment name. If `mode` indicates SA or QM, then `serverAE` will be ‘*’. Otherwise, it will reflect the value passed as the -AE start parameter.

**Message descriptions**

**IMW3501I**  Config: Hostname: `hostname`, Port: `number`, SSL Port: `number`, Server Root: `serverroot`, current debug settings

**Explanation:** Displays the server’s hostname, normal port, and server root. The SSL port is specified if the server is running in secure mode. If the server is not running in secure mode, NONE is indicated for the SSL port. Debug settings that are turned on are listed. The SMF recording setting is also displayed.

**User Response:** None.
Console messages

IMW3502I  Stats: Threads running: number, Threads idle: number, Requests: number, Bytes rcvd: number, Bytes sent: number, Active In Conns: number, Active Out Conns: number.
Connections since last SMF: number

Explanation: Displays activity statistics for the Web server.

For more information on these statistics, see "Web server activity statistics" on page 114.

User Response: None.

IMW3503I  Do not recognize option, parameter, on -d option.

Explanation: You entered a parameter that is not valid on the -d option on the MODIFY command.

User Response: The message displays the valid options. Be sure you entered an acceptable option.

IMW3504I  Debug has been enabled for module, module_name.

Explanation: You entered -debug module_name on the MODIFY command. Debugging for the named module has been enabled.

User Response: None.

IMW3505I  Debug has been enabled for all modules.

Explanation: You entered -debug on the MODIFY command. Debugging has been enabled for all modules.

User Response: None.

IMW3506I  Unknown module name, module_name, specified on the -debug option.

Explanation: You specified -debug module_name with an unknown module_name.

User Response: Correct the module name and try again. (Valid module names are listed in message IMW3523I.)

IMW3507I  Debug has been disabled for module, module_name.

Explanation: You entered -nodebug module_name on the MODIFY command. Debugging for the named module has been disabled.

User Response: None.

IMW3508I  Debug has been disabled for all modules.

Explanation: You entered -nodebug on the MODIFY command. Debugging for all modules has been disabled.

User Response: None.

IMW3509I  Unknown module name, unknown_module_name, specified on -nodebug option. Option is: -nodebug module_name. Turns of debugging for all modules or optionally specify one of the following: list of module names

Explanation: The specified module name is not valid.

User Response: Correct the module name and try again. Valid module names are listed in the message.

IMW3510I  SMF recording has been enabled for record type, record_type.

Explanation: You entered -smf record_type on the operator console MODIFY command. SMF recording for the named record type has been enabled.

User Response: None.

IMW3511I  SMF recording has been enabled for all record types.

Explanation: You entered -smf on the operator console MODIFY command. SMF recording has been enabled.

User Response: None.

IMW3512I  Do not recognize option, record_type, on the -smf option.

Explanation: You specified -smf record_type with an unknown record_type.

User Response: Correct the record_type and try again. Valid record_types are listed in the message.
**IMW3513I**  SMF recording has been disabled for record type, record_type

**Explanation:** You entered -nosmf record_type on the operator console MODIFY command. SMF recording for the names record type has been disabled.

**User Response:** None.

**IMW3514I**  SMF recording has been disabled for all record types.

**Explanation:** You entered -nosmf on the MODIFY command. SMF recording has been turned off.

**User Response:** None.

**IMW3515I**  Do not recognize option, record_type, on the -nosmf option.

**Explanation:** You specified -nosmf record_type with an unknown record_type.

**User Response:** Correct the record_type and try again. (Valid record_types are listed in message IMW3526.)

**IMW3516I**  Version: version

**Explanation:** You entered -version on the MODIFY command. The server version is displayed.

**User Response:** None.

**IMW3517I**  First level tracing (-v) enabled.

**Explanation:** You entered -v on the MODIFY command. Verbose tracing has been enabled.

**User Response:** For guidance and additional information on Web server traces, contact the IBM Software Support Center.

**IMW3518I**  Second level tracing (-vv) enabled.

**Explanation:** You entered -vv on the MODIFY command. Very Verbose tracing has been enabled.

**User Response:** For guidance and additional information on Web server traces, contact the IBM Software Support Center.

**IMW3519I**  Third level tracing (-mtv) enabled.

**Explanation:** You entered -mtv on the MODIFY command. Much Too Verbose tracing has been enabled.

**User Response:** For guidance and additional information on Web server traces, contact the IBM Software Support Center.

**IMW3520I**  Cache tracing (-vc) enabled.

**Explanation:** You entered -vc on the MODIFY command. Verbose cache tracing has been enabled.

**User Response:** For guidance and additional information on Web server traces, contact the IBM Software Support Center.

**IMW3521I**  Do not recognize option, option, on the MODIFY command.

**Explanation:** You entered an unknown option on the MODIFY command.

**User Response:** Message IMW3527I identifies valid options. Try the command again with a correct option.

**IMW3522I**  No Help available for option, option.

**Explanation:** You specified -? option on the MODIFY command. There is no help available for option. Help is available for debug (-? debug) and nodebug (-? nodebug), smf (-? smf), and nosmf (-? nosmf).

**User Response:** Enter -?? to see more information.

**IMW3523I**  Option is: -debug module_name Enables debugging for all modules, or optionally specify one of the following: module_names listed

**Explanation:** You should receive this message when -? debug is specified on the MODIFY command.

**User Response:** None.

**IMW3524I**  Option is: -nodebug module_name Disables debugging for all modules, or optionally specify one of the following: config, perf.

**Explanation:** You should receive this message when -? nodebug is specified on the MODIFY command.

**User Response:** None.

**IMW3525I**  Option is: -smf SMF_record_type Enables all SMF recording, or optionally specify one of the following: config, perf.

**Explanation:** You should receive this message when -? smf is specified on the operator console MODIFY command.

**User Response:** None.
Console messages

IMW3526I Option is: -nosmf SMF_record_type
Disables all SMF recording, or
optionally specify one of the following: config, perf.

Explanation: You should receive this message when -? nosmf is specified on the operator console MODIFY command.

User Response: None.

IMW3527I Options: -d config -d stats -debug
module_name -nodebug module_name
-restart -smf option -nosmf option -v -vv
-mtv -vc -version -? -?? debug -?
nodebug -? smf -? nosmf

Explanation: You entered -? on the MODIFY command.

User Response: None.


Explanation: You have entered -?? on the MODIFY command.

User Response: None.

IMW3529E Error on console()..string. Internal error.
This is a software error. Call IBM Software Support.

Explanation: This is an internal error in the console support function.

User Response: Note the information in the message and contact the IBM Software Support Center for assistance.

IMW3530E Error on msg parameter to HTWTO(),
strlen(msg)=0, msg=number
n_padding=number, flags=hex_value

Explanation: This is an internal error in the console support function.

User Response: Note the information in the message and contact the IBM Software Support Center for assistance.

IMW3531E Error on msg parameter to HTWTO(),
strlen(msg)=0, msg=number
n_padding=number, flags=hex_value

Explanation: This is an internal error in the console support function.

User Response: Note the information in the message and contact the IBM Software Support Center for assistance.

IMW3532E Error on msg parameter to HTWTO(),
strlen(msg)=0, n_padding=number,
flags=hex_value

Explanation: This is an internal error in the console support function.

User Response: Note the information in the message and contact the IBM Software Support Center for assistance.

IMW3533E Error on n_padding parameter to HTWTO(),
n_padding=number,
HTWTConsoleWIDTH=number

Explanation: This is an internal error in the console support function.

User Response: Note the information in the message and contact the IBM Software Support Center for assistance.

IMW3534I PID: processid SERVER STARTING

Explanation: The server is initializing. processid is the server’s numeric process identifier assigned by OS/390 UNIX System Services. Message IMW3535E or IMW3536I should follow this message, indicating initialization failure or success.

User Response: None

IMW3535E PID: processid SERVER INITIALIZATION FAILED

Explanation: Server initialization started, but did not complete, due to one or more error conditions.

User Response: Review error messages that have been written to the job log (using DD STDERR) for a description of the error(s). processid is the server’s process identifier. Correct the errors and restart the server.

IMW3536E PID: processid SERVER
INITIALIZATION FAILED

Explanation: Server initialization started, but did not complete, due to one or more error conditions.

User Response: Review error messages that have been written to the job log (using DD STDERR) for a description of the error(s). processid is the server’s process identifier. Correct the errors and restart the server.
**IMW3536I  process_descriptor READY**

**Explanation:** The server has initialized successfully, and can process Internet requests. For more information, see “Explanation of process descriptor in messages” on page 459.

**User Response:** None.

**IMW3537I  process_descriptor RESTARTING**

**Explanation:** The server has been requested to terminate and restart its processing (possibly an operator request to restart). The configuration file is read, which causes some (not all) functions to be recycled. During restart, new connections are not allowed, and existing connections are quiesced. IMW3538I or IMW3539E should follow this message, indicating a successful restart or failure. For more information, see “Explanation of process descriptor in messages” on page 459.

**User Response:** None.

**IMW3538I  process_descriptor RESTART SUCCESSFUL**

**Explanation:** The server completed the restart successfully, and can process internet requests. For more information, see “Explanation of process descriptor in messages” on page 459.

**User Response:** None.

**IMW3539E  process_descriptor RESTART FAILED**

**Explanation:** The server began to restart, but did not complete successfully, due to one or more errors. For more information, see “Explanation of process descriptor in messages” on page 459.

**User Response:** Review the dump to diagnose the error. Start the server after correcting the problem.

**IMW3701E-3726E: HTCounter Program Messages**

**IMW3701E HTCounter: Counter file not found, tried the following: counterfile**

**Explanation:** A URL for this server specified the apicounter function available in the htcounter program. It tried to access a non-existent counter file.

**User Response:** Ask your Web administrator to create the counter file in the counters subdirectory of the server’s root directory with an initial value in it (typically 0) and set permissions to allow the server write access.

**IMW3702E HTCounter: strftime() call failed: TIMEFMT=’format_string’**

**Explanation:** A URL for this server specified the datetime function available in the htcounter program. The call to strftime() to format the current date and time failed. The message shows the format option used on the call to strftime().

**User Response:** Note the information in the message and contact the IBM Software Support Center for assistance.

**IMW3703E HTCounter: software error: strftime() call failed: result of localtime_r() call is 0.**

**Explanation:** A URL for this server specified the datetime function available in the htcounter program. The call to localtime_r() to determine the current date and time failed.

**User Response:** Contact the IBM Software Support Center for assistance.
HTCounter Program messages

IMW3704E  HTCounter: software error: strftime() call failed: result of gm_time_r() call is 0.
Explanation: A URL for this server specified the datetime function available in the htcounter program. The call to gm_time_r() to determine the current date and time failed.
User Response: Contact the IBM Software Support Center for assistance.

Explanation: A URL for this server specified the datetime function available in the htcounter program. This message indicates that an internal error occurred in the htcounter program.
User Response: Note the information in the message and contact the IBM Software Support Center for assistance.

IMW3706E  HTCounter: unable to open Counters/Fonts/FormsEtc.dat file..tried filename.
Explanation: An error occurred while the htcounter program was attempting to load the fonts files. The file, FormsEtc.dat, should be located under the Counters/Fonts subdirectory in the server root subdirectory. The message indicates where the htcounter program attempted to load the file from.
User Response: This error could occur if there was an installation problem, or if this file was inadvertently moved or deleted. Search for the file, FormsEtc.dat, and restore it to the subdirectory, <ServerRoot>/Counters/Fonts. It may be necessary to reinstall the server.
If necessary, contact the IBM Software Support Center for assistance.

IMW3707E  HTCounter: too many fonts specified in filename file.. maximum allowed=number_allowed., file asks for number_requested
Explanation: The FormsEtc.dat file, named filename, was read and there is an error in the contents.
User Response: This error could occur if there was an installation problem, or if this file was inadvertently changed. Attempt to restore the original contents of the FormsEtc.dat file. It may be necessary to reinstall the server.
If necessary, contact the IBM Software Support Center for assistance.

IMW3708E  HTCounter: unable to open Counters/Fonts/font_filename file..tried full_font_filename.
Explanation: An error occurred while the htcounter program was attempting to load one of the fonts files. The file, font_filename, should be located under the Counters/Fonts subdirectory in the server root subdirectory. The message indicates where the htcounter program attempted to load the file from.
User Response: This error could occur if there was an installation problem, or if this file was inadvertently moved or deleted. Search for the file, font_filename, and restore it to the subdirectory, <ServerRoot>/Counters/Fonts. It may be necessary to reinstall the server.
If necessary, contact the IBM Software Support Center for assistance.

IMW3709E  HTCounter: Error loading font file, font_name font_width x font_height, Character character_index character_value (0x: character_hex_value) is not printable.
Explanation: An error occurred while the htcounter program was attempting to load one of the fonts files. The font name, font size, and character in error are indicated in the message.
User Response: This error could occur if there was an installation problem, or if this file was inadvertently changed. Attempt to restore the original contents of the font files, Block1.dat and LCD.dat. It may be necessary to reinstall the server.
If necessary, contact the IBM Software Support Center for assistance.

IMW3710E  HTCounter: ...expecting ServerRoot to be: ServerRoot (based on your server configuration file).
Explanation: An error occurred where the htcounter program expected ServerRoot in the httpd.conf configuration file to be set as indicated in the message. This message should be proceeded by another message with additional information.
User Response: Refer to the accompanying message to determine the cause of the problem.

Explanation: The htcounter program relies on the setting of the ServerRoot directive in the httpd.conf configuration file to locate various files, such as the fonts files and the counter files. This message indicates that ServerRoot has not been set in the httpd configuration file.
User Response: Check the ServerRoot directive in the
httpd.conf file. Set it to the current working directory of the Web server. The initial configuration file setting is /usr/lpp/internet/server_root.

**IMW3712E** HTCounter: there is an error with the HTCounter font files, see your Web system administrator.

**Explanation:** This message is displayed in the browser. An error was encountered by the htc counter program when trying to read the fonts files. There may be other messages in the server error log which may help determine the cause of the problem.

**User Response:** See your Web administrator. Check the httpd error log for the string HTCounter: and for other error messages that may indicate the cause of the problem.

**IMW3713E** HTCounter: Error reading counter file.

**Explanation:** A URL for this server specified the apicounter function available in the htc counter program. It tried to access a counter file that did not contain a count (for example, the counter file was not initialized properly).

**User Response:** Ask your Web administrator to use a text editor to initialize the counter file to the initial value you want to use (for example, 0).

**IMW3714E** HTCounter: Error flushing counter file.

**Explanation:** This is an internal error related to closing and flushing the counter file.

**User Response:** Use a text editor to view the counter file. Identify any problems accessing the file.

If necessary, contact the IBM Software Support Center for assistance.

**IMW3715E** HTCounter: Error closing counter file after reading.

**Explanation:** This is an internal error relating to closing and flushing the counter file.

**User Response:** Use a text editor to view the counter file. Identify any problem accessing the file.

If necessary, contact the IBM Software Support Center for assistance.

**IMW3716E** HTCounter: Error opening counter file for write.

**Explanation:** A URL for this server specified the apicounter function available in the htc counter program. It tried to access a counter file that did not have the permissions set properly. The httpd server needs write access to the file.

**User Response:** Ask your Web administrator to give

---

**HTCounter Program messages**

**IMW3717E** HTCounter: Error writing counter file.

**Explanation:** This is an internal error, relating to writing the counter file.

**User Response:** Use a text editor to view the counter file. Identify any problems accessing the file.

If necessary, contact the IBM Software Support Center for assistance.

**IMW3718E** HTCounter: Error closing counter file after writing.

**Explanation:** This is an internal error relating to closing and flushing the counter file.

**User Response:** Use a text editor to view the counter file. Identify any problems accessing the file.

If necessary, contact the IBM Software Support Center for assistance.

**IMW3719E** HTCounter: Counter file does not exist, tried: counterfile

**Explanation:** A URL for this server specified the apicounter function available in the htc counter program. It tried to access a non-existent counter file.

**User Response:** Ask your Web administrator to create the counter file in the counters subdirectory of the server's root directory with an initial value in it (typically, 0) and set permissions to allow the server write access.

**IMW3720E** HTCounter: Software error in accessing counter file.

**Explanation:** This is an internal error, relating to writing the counter file.

**User Response:** Use a text editor to view the counter file. Identify any problems accessing the file.

If necessary, contact the IBM Software Support Center for assistance.

**IMW3721E** HTCounter: No counter file specified on apicounter URL.

**Explanation:** You did not specify a counter file on your URL when trying to use the apicounter function.

**User Response:** Specify a valid counter file name on the URL.
HTCounter Program messages

IMW3722E HTCounter: No text specified on text2gif URL.
Explanation: You did not specify a text string on your URL when trying to use the text2gif function.
User Response: Specify a text string on the TEXT option on the URL.

IMW3723E HTCounter: ....errno indicates: string
Explanation: This message gives supplementary information for a problem indicated by the message that should have preceded this one.
User Response: Refer to the accompanying message.

IMW3724E HTCounter: Not able to allocate memory for GIF tables.
Explanation: The htcouter program was not able to allocate storage for the temporary tables required to generate the gif image.
User Response: If the problem persists, there is a shortage of virtual storage. Attempt to use other utilities to determine if the server is getting enough storage. If the storage usage of the server continually increases over time, there may be a problem with the server.
Contact the IBM Software Support Center for assistance.

IMW3725E HTCounter: Not enough memory to allocate Image buffer..we were trying for number bytes.
Explanation: The htcouter program was not able to allocate storage for the temporary buffers required to generate the image.
User Response: If the problem persists, there is a shortage of virtual storage. Attempt to use other utilities to determine if the server is getting enough storage. If the storage usage of the server continually increases over time, there may be a problem with the server.
Contact the IBM Software Support Center for assistance.

IMW3726E HTCounter: Not enough memory to allocate GIF buffer..we were trying for number bytes.
Explanation: The htcouter program was not able to allocate storage for the temporary buffers required to generate the gif image.
User Response: If the problem persists, there is a shortage of virtual storage. Attempt to use other utilities to determine if the server is getting enough storage. If the storage usage of the server continually increases over time, there may be a problem with the server.
Contact the IBM Software Support Center for assistance.

IMW4000E-4018E: HTIMAGE Messages

IMW4000E A URL was not returned, nor was the default set for the picture.
Explanation: The coordinates returned from the client by clicking on an image map are not defined in the corresponding map file. There is no default action defined in the map file. The htimage program could not determine a redirection URL to send back to the client.
User Response: Add a default to the map file for this picture.

IMW4001E An error occurred while parsing picture configuration file.
Explanation: The htimage program encountered unrecognized data in the map file.
User Response: Correct the map file.

IMW4002E Expecting a closing parenthesis.
Explanation: The htimage program found a syntax error in the map file.
User Response: Correct the map file.

IMW4003E Expecting a comma separating the x and y.
Explanation: The htimage program found a syntax error in the map file.
User Response: Correct the map file.

IMW4004E Expecting a coordinate pair.
Explanation: The htimage program found a syntax error in the map file.
User Response: Correct the map file.
IMW4005E  Expecting a default URL.
Explanation: The htimage program found a syntax error in the map file.
User Response: Correct the map file.

IMW4006E  Expecting a field name.
Explanation: The htimage program found a syntax error in the map file.
User Response: Correct the map file.

IMW4007E  Expecting a first coordinate pair.
Explanation: The htimage program found a syntax error in the map file.
User Response: Correct the map file.

IMW4008E  Expecting a radius.
Explanation: The htimage program found a syntax error in the map file.
User Response: Correct the map file.

IMW4009E  Expecting a second coordinate pair.
Explanation: The htimage program found a syntax error in the map file.
User Response: Correct the map file.

IMW4010E  Expecting a URL.
Explanation: The htimage program found a syntax error in the map file.
User Response: Correct the map file.

IMW4011E  Expecting a y coordinate.
Explanation: The htimage program found a syntax error in the map file.
User Response: Correct the map file.

IMW4012E  Expecting an x coordinate.
Explanation: The htimage program found a syntax error in the map file.
User Response: Correct the map file.

IMW4013E  Picture configuration file was not found. Tried the following: string.
Explanation: The htimage program could not find the image map file referenced by the HTML page the client clicked on.

User Response: Correct the HTML or create the image map file.

IMW4014E  Syntax error at line
Explanation: The htimage program found a syntax error in the map file.
User Response: Correct the map file.

IMW4015E  The field name is not valid. Expecting string, 'default', 'rectangle', 'circle' or 'polygon'.
Explanation: The htimage program found a syntax error in the map file.
User Response: Correct the map file.

IMW4016E  The QUERY_STRING is not valid: string. expecting either x,y or x=x&y=y;
Explanation: The htimage program did not receive a valid coordinate pair for clicking on an image map. Verify that the query string was not damaged during the transmission of this request through your configuration file. Verify that the HTML page being viewed is correct. Verify that the browser program is operating correctly.
User Response: Correct any errors.

IMW4017E  You did not set either the PATH_INFO or the PATH_TRANSLATED environment variable.
Explanation: The htimage program did not receive a map file name. Verify that the path info is not lost while translating your request through your configuration file. Verify that the HTML page being viewed is correct. Verify that the browser program is operating correctly.
User Response: Correct any errors.

IMW4018E  You did not set the QUERY_STRING environment variable.
Explanation: The htimage program did not receive a QUERY_STRING. Verify that the query string is not lost while translating this request through your configuration file. Verify that the HTML page being viewed is correct. Verify that the browser program is operating correctly.
User Response: Correct any errors.

IMW5001E-5010E: HTADM Messages
HTADM messages

IMW5001E  Cannot open password file string.
Explanation: The htadm program was unable to open the specified password file. Verify that the file name is correct and that you have appropriate permissions.
User Response: Correct any problems found.

IMW5002I  Administrative tool for Server access authorization.
Explanation: Usage information for htadm program.
User Response: None.

IMW5003E  Cannot create file name.
Explanation: The htadm program could not create the password file. Verify the file system and permissions.
User Response: Correct any file system problems and retry.

IMW5004E  Cannot create temporary file name.
Explanation: The htadm program could not create the password file. Verify the file system and permissions.
User Response: Correct any file system problems and retry.

IMW5005E  Cannot find password file string.
Explanation: The htadm program could not find the referenced file. Verify the filename spelling and your permissions.
User Response: Correct any problems found and retry.

IMW5006I  Correct
Explanation: The password that you checked is correct.

IMW5007E  Incorrect
Explanation: The password that you checked is not correct.
User Response: None.

IMW5008E  Entry for user name string in password file.
Explanation: The htadm program found an error. For -adduser, the user name is already defined. For -deluser or -passwd, the user name is not found.
User Response: Correct the user name and retry.

IMW5009E  File name already exists.
Explanation: The htadm program can't create the requested password file.
User Response: Correct the name and retry.

IMW5010E  New file size size does not match expected size. Original password file left intact.
Explanation: The password file has been modified by someone else at the same time. Your change was not applied.
User Response: Try again.

IMW6102I-6805E: SSL Security Messages

IMW6102I  Key File Password: password
Explanation: The key file password has not been entered.
User Response: Enter key file password.

IMW6104E  Bad password
Explanation: The key file password you entered is not correct.
User Response: Verify that the password is correct and try entering it again. If the password cannot be remembered, a new key file has to be created.

IMW6304E  Key Data Base not read. Please check key file existence, permission and ownership. Using Default root keys.
Explanation: An error was detected when the Web server attempted to open the key file specified in the configuration file.
User Response: Verify the key file has the correct read and write permissions. The Web server must have permission to read the key file.
IMW6801E Both SSL and normal modes have been turned off. At least one of the two must be turned on. Normal mode will be turned on.

**Explanation:** An error occurred at server startup because both normal and SSL ports were disabled. At least one of the ports must be active to establish a TCP/IP socket connection.

**User Response:** For a secure network connection, set SSLMode on in the configuration file.

For more information, see [SSL Mode - Turn port on or off for SSL](#) on page 371 and [NormalMode - Turn port on or off for HTTP](#) on page 368.

IMW6802E SSL Handshake failed.

**Explanation:** An error occurred in the server to client SSL handshake. SSL session could not be established.

**User Response:** Verify that the server supports SSL and has a valid certificate. Try the transaction again.

IMW6803I SSL Port

**Explanation:** Informational message displaying the SSL TCP/IP listen port.

**User Response:** None.

IMW6804E The parameter to -sslport option is not valid: `parameter`

**Explanation:** An invalid parameter was entered for sslport option on server startup. Default value is 443, but any port greater than 1024 may be specified.

**User Response:** Verify the parameter entered for sslport and try again.

IMW6805E The parameter to -sslmode option is not valid: `parameter`

**Explanation:** An invalid parameter was entered for sslmode option on server startup. Valid values are ON and OFF.

**User Response:** Verify the parameter entered for sslmode and try again.
SSL Security messages
Appendix I. TCP/IP reference

Selecting data sets or files . . . . . . . . . . . . 471
Data set search order . . . . . . . . . . . . . . . . 471
OS/390 UNIX System Services data set environment. . . . . . . . . . . . . . 471
TCP/IP file placement configuration . . . . 472

Selecting data sets or files

Data sets and files are comparable terms. Data set is the non-OS/390 UNIX term; file is the OS/390 UNIX term. TCP/IP is a protocol that enables users to access data sets or files.

If you are familiar with OS/390, you probably use the term “data set” to describe a unit of data storage. More specifically, a data set is the major unit of data storage and retrieval, consisting of a collection of data in one of several prescribed arrangements and described by control information to which the system has access.

If you are familiar with UNIX, you probably use the term “file” to describe a named set of records stored or processed as a unit.

Because TCP/IP is a protocol, it uses a set of semantic and syntactic rules to perform its communication functions and to access data sets and files. The TCP/IP rules use a search order to access the data sets or files. TCP/IP also uses data sets or files in a specific order to perform its communication functions that access other data sets and files. The TCP/IP search order for the data sets or files used or accessed by the OE applications differs from the search order for the data sets or files used or accessed by non-OE applications.

Some of the data sets or files have special importance because of their functions. For example, certain data sets are used for configuration. The most fundamental data sets are the configuration data sets. In order, they are searched for the data set name of a service requested or a block of stored data. The search order differs between types of network configurations.

Data set search order

The table that follows illustrates OE search order of data sets. The new OpenEdition search rules are implemented in OS/390 UNIX.

In the following table, the abbreviations below are used:

hlq  Value of DATASETPREFIX from TCPIP.DATA. hlq defaults to TCPIP.

Note: OE does not support the TCPIP post install "zap" to change this hlq.

$X  Value of environment variable X used "as is" in fopen().

OS/390 UNIX System Services data set environment

The following table maps TCP/IP specific MVS data sets to their counterparts in the OS/390 UNIX environment.
## TCP/IP file placement configuration

The following table illustrates the recommended TCP/IP configuration file placement:

<table>
<thead>
<tr>
<th>TCP/IP Configuration File</th>
<th>OS/390 UNIX Applications</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCPIP.DATA share same file</td>
<td>SYS1.TCPPARMS(TCPDATA)</td>
</tr>
<tr>
<td>STANDARD.TCPXLBIN share same file</td>
<td>hlq.STANDARD.TCPXLBIN</td>
</tr>
<tr>
<td>HOSTS.SITEINFO share same file</td>
<td>HOSTS.SITEINFO</td>
</tr>
<tr>
<td></td>
<td>hlq.HOSTS.SITEINFO</td>
</tr>
<tr>
<td>HOSTS.ADDRINFO share same file</td>
<td>HOSTS.ADDRINFO</td>
</tr>
<tr>
<td></td>
<td>hlq.HOSTS.ADDRINFO</td>
</tr>
<tr>
<td>ETC.PROTO share same file</td>
<td>ETC.PROTO</td>
</tr>
<tr>
<td></td>
<td>hlq.ETC.PROTO</td>
</tr>
<tr>
<td>ETC.SERVICES must not share same</td>
<td>/etc/services</td>
</tr>
<tr>
<td>services file services are on</td>
<td></td>
</tr>
<tr>
<td>different ports</td>
<td></td>
</tr>
</tbody>
</table>
Glossary

For more information on terms used in this book, go to the *IBM Dictionary of Computing* on the IBM Web site at URL:

Bibliography

This bibliography lists the books related to Version 5.2 of the IBM HTTP Server for OS/390.

The HTTP Server books can be accessed from the Front Page of your server. They are also available from the HTTP Server Web site Library at URL: http://www.ibm.com/software/websphere/httpservers/

The HTTP Server books and most related publications are also available in BookManager format on the OS/390 CD-ROM Collection Kit and from the OS/390 Web site at URL: http://www.ibm.com/s390/os390/bkserv/

For a summary of available OS/390 books and online information, see the OS/390 Information Road Map.

HTTP Server publications

- HTTP Server Planning, Installing, and Using, SC31-8690-04
  This book explains how to plan for, install, configure, and use the Web server. The Programming section explains how to write external programs that interact with the Web server using the Common Gateway Interface (CGI), Go Webserver API (GWAPI), or LDAP API.

- Enterprise Web Serving with the IBM HTTP Server for OS/390, SG24-2074
  Use this Redbook along with the Planning, Installing, and Using book to help you plan for, install, tailor, and configure the Web server. This Redbook will also help you migrate from previous levels of the server.

- Troubleshooter
  This Web-based guide provides the most current troubleshooting hints and tips for the Web server. To access this information, go to the HTTP Server Web site Library at URL: http://www.ibm.com/software/websphere/httpservers/

- IBM Web Traffic Express for Multiplatforms User's Guide V1.0, GC31-8645-00
  This book describes IBM’s proxy server for the IBM HTTP Server.

Note: The Web Traffic Express guide does not contain OS/390-specific information.

Before using this guide, refer to the HTTP Server Planning, Installing, and Using book chapter on running your server as a proxy.

Application Server publications

- Application Server V1.2 Planning, Installing, and Using (GC34-4757-01)
  This book is available in HTML and PDF formats on the Application Server Web site at URL: http://www.ibm.com/software/websphere/appserv/library.html

- Application Server V1.1 Getting Started, (GC34-4757-00)

OS/390 publications

- OS/390 Information Road Map, GC28-1727
  This book describes available information for the elements and features in OS/390. It also explains how to order OS/390 documentation and how to access online OS/390 information.

- OS/390 Planning for Installation, GC28-1726
  This book lists the elements and features in OS/390. It explains how to get OS/390 up and running, and provides information about migration actions for specific elements of OS/390.

- OS/390 MVS Planning: Workload Management, GC28-1761
  This book explains Workload Management (WLM) concepts and interfaces, and includes the steps required for using WLM as well as its benefits.

- OS/390 UNIX System Services Planning, SC28-1890
  This book explains how to plan for and install OS/390 UNIX System Services.

- OS/390 UNIX System Services Command Reference, SC28-1892
  This book explains the commands used by OS/390 UNIX System Services.

- OS/390 MVS System Management Facilities, GC28-1783

475
Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user’s responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Mail Station P300
522 South Road
Poughkeepsie, NY 12601-5400
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR
PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain translations, therefore, this statement may not apply to you.

This publication documents intended Programming Interfaces that allow the customer to write programs to obtain the services of the IBM HTTP Server.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

# Trademarks

The following terms are trademarks of the IBM Corporation in the United States or other countries or both:

- AIX
- BookManager
- CICS
- CICS/ESA
- C/MVS
- DFSMS
- IBM
- IBMLink
- Language Environment
- Multiprise
- MVS/ESA
- OpenEdition
- OS/2
- OS/390
- RACF
- RMF
- S/390
- System/390
- VTAM
- WebSphere

Adobe, Acrobat, and PostScript are trademarks of Adobe Systems Incorporated.

CORBA is a trademark of the Object Management Group, Incorporated.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and/or other countries.
Lotus, Domino, Lotus Go Webserver, and Lotus Notes are trademarks of the Lotus Development Corporation in the United States, or other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States and/or other countries.

Netscape and Netscape Navigator are trademarks of the Netscape Communications Corporation in the United States, or other countries, or both.

OpenView is a trademark of the Hewlett-Packard Company.

POSIX is a trademark of the Institute of Electrical and Electronic Engineers.

Tivoli, TME, TME10, TME 10 NetView, TME 10 Distributed Monitoring, are trademarks of Tivoli Systems Inc. in the United States, or other countries, or both.

UNIX is a registered trademark in the United States and/or other countries licensed exclusively through X/Open Company Limited.

Other company, product or service names may be the trademarks or service marks of others.
Index

Special Characters
%%CERTIF%% 20, 286
%%CLIENT%% 20, 286
%%SERVER%% 20, 286

Numerics
0A2F (address bit dirty) errors 20

A
ABEND, Web server
CE dump requirements 3
IMW0085E 428
IMW0162E 434
planning considerations 3
Recovery directive 292
recovery options 292
Access Control directives and subdirectives
DefProt 275
Protect 278
Protection 282
Protection subdirectives
ACLOverride 283
AuthType 283
DeleteMask 283
GetMask 283
GroupFile 284
Mask 284
PasswordFile 284
PostMask 285
PutMask 285
ServerID 285
UserID 286
Access Control List (ACL) files 60
access control user IDs 20
access control using RACF or other SAF-based security products 4
access log maintenance options 74
Server Activity Monitor 117, 163
set filters 75
accessing the HTTP Server 35
AccessLog directive 324
AccessLogArchive directive 324
AccessLogExcludeMethod directive 326
AccessLogExcludeMimeType directive 326
AccessLogExcludeReturnCode directive 327
AccessLogExcludeURL directive 325
AccessLogExpire directive 327
AccessLogSizeLimit directive 328
AccessReportDescription directive 328
AccessReportDoDnsLookup directive 328
AccessReportExcludeHostName directive 330
AccessReportExcludeMethodName directive 331
AccessReportExcludeReturnCode directive 331
AccessReportExcludeURL directive 329
AccessReportIncludeHostName directive 331
AccessReportIncludeURL directive 329
AccessReportRoot directive 332
AccessReportTemplate directive 332
AccessReportTopList directive 333
ACLOverride subdirective 283
activity statistics
network 117
server 115
AddBlankIcon directive 295
AddCharSet directive 353
AddClient directive 357
AddDirIcon directive 296
AddEncoding directive 352
AddIcon directive 296
AddLanguage directive 352
AddParentIcon directive 297
Address, Multiple IP 165
AddType directive 353
AddUnknownIcon directive 297
authentication 48, 214
Authentication, GWAPI process step 201
Authorization, GWAPI process step 201
Authentication directive 313
authorization 215
Authentication, GWAPI process step 201
prototype 202
Authentication directive 313
authorization 215
Authorization, GWAPI process step 201
prototype 202
Authorization directive 314
authorizations for OS/390 UNIX System Services 7
AuthType subdirective 283
automatic browser detection 357
badrequest error code 307
baduser error code 307
badsript error code 306
B
backing up files 44
badredirect error code 307
badrequest error code 306
Cache Accelerator 69
Certificate and Administration Forms user interface 4
Global Server ID for 128-bit encryption 53
HTTP Server Certificate Authority user interface 4
browsers, tested 5
byrule error code 308
C
Cache Access Log directive 334
CacheLocalFile directive 384
CacheLocalMaxBytes directive 384
CacheLocalMaxFiles directive 385
central file management for PICS 147
certificates
acting as your own CA becoming a CA 233
buying CA software 55, 233
overview 233
processing certificates 236
HTTP Server CA utility creating a file for the database password 235
downloading a CA certificate 237
exporting CA keys 236
maximum number of certificates 233
migration considerations 13
overview 236
processing certificate requests 237
receiving an approved certificate 238
requesting a certificate 237
requirements for using 234
using the HTTP Server CA utility 236
using the IBM Key Management Utility 233
certificate authority, server utility 236
certificates
buying from a CA 54
certificates (continued)
digital 48
issuing your own 55
migrating 13
supported by the HTTP Server 54
CGI data, parsing 181
CGI Data Manipulation 182
cgi_error log 335
CGI example for OS/390 188
CGI example in REXX 193
CGI on OS/390, parsing 185
CGI response on OS/390 187
CGI scripts 171
CGI shell script example 195
CGI variables 184
CgiErrorLog directive 335
cgiparse command 241
cgutils 183
cgutils command 244
codepage, language
DefaultFsCp 294
DefaultNetCp 294
Codepage type 226
command line, running web usage
mining from 87
commands
cgiparse 241
cgutils 244
htadm 246
htimage 101
httpd 249
ikemony 418
IMWHTTPD program 253
Simple Network Management Protocol (SNMP) 118
wwwcmd 261
community names, SNMP 134, 381
compiling CGI Programs 171
compiling GWAPI programs 199
confidentiality, secure communications 47
Configuration and Administration Forms 42
configuration file
access control 275
basic server settings 288
directories and welcome page 295
error message customization 305
GWAPI application processing 311
logs and reports 323
meta-information 347
methods 348
performance settings 383
resource mapping 359
security 367
syntax 152
timeouts 382
user directories 304
configuring a proxy server
access control 163
proxy settings 163
port number 163
configuring the HTTP Server
controlling access to 44
editing the configuration file 43
using the Configuration and Administration Forms 42
CONNECT method 348
CONNECTION_POOL, httpd command 250
console commands
Workload Management 265
counter, how to display on a Web page 89
creating a home page 41
key database 412
key pair and certificate request 414
self-signed certificate 414
creating PICS labels 149
criteria for rating Web sites 144
customizing your Web site
displaying page count, date, and time 89
sending customized pages 357
using server-side includes 93
D
DAEMON SAF facility class 7
Data Filter, GWAPI process step 201
prototype 203
data set
configuration file 7
DataFilter directive 316
date and time, how to display on a Web page 89
DBLookup directive 316
Debug Tool, for C/C++ GWAPI programs 226
DebugToolAddr directive 312
default error messages 306
DefaultFsCp directive 294
DefaultNetCp directive 294
DefineLBService paragraph, PICS 151, 153
DefineService paragraph, PICS 151, 152
DebugToolAddr directive 312
default error messages 306
DefaultFsCp directive 294
DefaultNetCp directive 294
DefineLBService paragraph, PICS 151, 153
DefineService paragraph, PICS 151, 152
DefProt directive 275
DELETE method 348
DeleteMask subdirective 283
digital certificate 48
digital signature 48
DirAccess directive 298
dirbrowse error code 308
directives
AccessLog 324
AccessLogArchive 324
AccessLogExcludeMethod 326
AccessLogExcludeMimeType 326
AccessLogExcludeReturnCode 327
AccessLogExcludeURL 325
AccessLogExpire 327
AccessLogSizeLimit 328
AccessReportDescription 328
AccessReportDoDnsLookup 328
AccessReportExcludeHostName 330
AccessReportExcludeMethod 331
AccessReportExcludeReturnCode 331
AccessReportExcludeURL 329
AccessReportIncludeHostName 331
AccessReportIncludeURL 329
AccessReportRoot 332
AccessReportTemplate 332
AccessReportTopList 333
connections (continued)
digital 48
issuing your own 55
migrating 13
supported by the HTTP Server 54
CGI data, parsing 181
CGI Data Manipulation 182
cgi_error log 335
CGI example for OS/390 188
CGI example in REXX 193
CGI on OS/390, parsing 185
CGI response on OS/390 187
CGI scripts 171
CGI shell script example 195
CGI variables 184
CgiErrorLog directive 335
cgiparse command 241
cgutils 183
cgutils command 244
codepage, language
DefaultFsCp 294
DefaultNetCp 294
Codepage type 226
command line, running web usage
mining from 87
commands
cgiparse 241
cgutils 244
htadm 246
htimage 101
httpd 249
ikemony 418
IMWHTTPD program 253
Simple Network Management Protocol (SNMP) 118
wwwcmd 261
community names, SNMP 134, 381
compiling CGI Programs 171
compiling GWAPI programs 199
confidentiality, secure communications 47
Configuration and Administration Forms 42
configuration file
access control 275
basic server settings 288
directories and welcome page 295
error message customization 305
GWAPI application processing 311
logs and reports 323
meta-information 347
methods 348
performance settings 383
resource mapping 359
security 367
syntax 152
timeouts 382
user directories 304
configuring a proxy server
access control 163
proxy settings 163
port number 163
configuring the HTTP Server
controlling access to 44
editing the configuration file 43
using the Configuration and Administration Forms 42
CONNECT method 348
directives (continued)

FRCAWLParms 389
HostName 289
IconPath 302
imbeds 290
InheritEnv 366
InputTimeout 382
InstallPath 291
KeyFile 367
LDAPInfo 318
LiveLocalCache 390
Log 317
LogFormat 338
LoggingReportingDebugOutput 339
LoggingReportingProgram 339
LoggingReportingProgramOptions 339
LogTime 340
LogToSyslog 340
Map 362
MaxActiveThreads 390
MaxPersistRequest 390
MetaDir 347
MetaSuffix 348
NameTrans 313
NoLog 341
NormalMode 368
ObjectType 314
OutputTimeout 383
overview 274
Pass 364
PersistTimeout 391
PICSDBLookup 316
PIdFile 291
PluginDefault 378
PluginExclude 378
PluginInclude 378
Port 292
PostExit 318
PreExit 312
Protect 278
Protection 282
Protection subdirectives
ACLOverride 283
AuthType 283
DeleteMask 283
GetMask 283
GroupFile 284
Mask 284
PasswordFile 284
PostMask 285
PutMask 285
ServerID 285
UserID 286
ProxyAccessLog 163, 341
Recovery 292
Redirect 365
RefererLog 324, 342
ReportDataArchive 344
ReportDataCompressionProgram 342
ReportDataCompressionSuffix 343
ReportDataExpire 345
ReportDataSizeLimit 344
ReportDataUnCompressionProgram 342
ReportDataUnCompressionSuffix 343
ReportProcessOldLogs 343
ScriptTimeout 383
ServerInit 312
directives (continued)

ServerPriority 391
ServerRoot 293
ServerTerm 318
Service 315
SMF 346
SMFRecordingInterval 346
SNMP 381
SNMPCommunityName 381
SSLCipherSpec 369
SSLCertAuth 370
SSLMode 371
SSLPort 371
SSLServerCert 371
SSLV2Timeout 372
SSLV3Timeout 373
SSLX500CACert 373
SSLX500Host 373
SSLX500Password 375
SSLX500Port 374
SSLX500UserID 374
SuffixCaseSense 375
UseACLs 391
UseMetaFiles 392
UserDir 304
Userld 293
WebMasterEmail 382
Welcome 303
directory listings 41

DirReadme directive 299
DirShowBrackets directive 299
DirShowCase directive 300
DirShowDate directive 300
DirShowDescription directive 300
DirShowGroup directive 300
DirShowHidden directive 301
DirShowIcons directive 301
DirShowMaxDescrLength directive 301
DirShowMinLength directive 301
DirShowMode directive 302
DirShowOwner directive 302
DirShowSize directive 302
Disable directive 349
Distinguished Name (DN) 49
DNS-Lookup directive 289
domain name, specifying 289
DoReporting directive 335
dotdot error code 308
dynamic documents 171, 172, 182

E

EBCDIC 185, 187
EBCDIC conversion 185, 187
editing
PICS configuration file 151
server configuration file 43
Enable directive 349
EnableFRCA directive 385
ciphering
128-bit with Global Server ID 53
hardware 57
options 53
overview 47
SSL cipher specifications 56
support 55, 56
Error, GWAPI process step 201
prototype 204
error condition key words
badredirect 307
badrequest 306
badscript 306
baduser 307
byrule 308
defined 305
dirbrowse 308
dotdot 308
ipmask 308
ipmaskproxy 308
methoddisabled 308
multifail 309
noacl 308
noentry 308
noformat 311
notallowed 308
notauthorized 307
notmember 307
okredirect 306
openfailed 308
proxyfail 307
scriptio 310
scriptnotfound 310
scriptstart 311
service error 311
seterror 309
unknownmethod 307
Error directive 317
error logging
maintenance options 77
path 76
specify path 76
error recovery, Web server
CE dump requirements 3
IMW0085E 428
IMW0162E 434
planning considerations 3
Recovery directive 292
Index 483
error recovery, Web server (continued)
recovery options 292
ErrorLog directive 335
ErrorLogArchive directive 336
ErrorLogExpire directive 337
ErrorLogSizeLimit directive 338
ErrorPage directive 305
example, GWAPI REXX Service 224
example LDAP program 230
examples
accesses report (excluding beta and
alpha7 requests) 84
changing the server’s default
encryption settings 65
configuring log files 78
department server accesses report
(except internal addresses) 84
page hits report 82
protecting server resources 58
PUT requests to beta subdirectory
report 83
setting up secure communications 61
setting up SSL support for multiple IP
addresses 65
Exec directive 359
ExecDirPass directive 361
execution modes, WLM
defining application environments
with ISPF 107
establishing policies 107
PROC 112
running multiple servers 106
scalable server subsystem 106
standalone server 105
using ISPF panels to define
application environments. 107
export editions 55
exporting keys 415
External directive 175

F
Fail directive 361
Fast Response Cache Accelerator 69
FastCGI 173
configuration changes 173
file permissions 302
filters for access log
defaults 76
external hits information 76
overview 75
reduce log size 75
Website access information 76
forms, Configuration and
Administration 42
FRCA, Fast Response Cache
Accelerator 69
FRCAAccessLog directive 385
FRCACacheEntries directive 386
FRCACacheOnly directive 386
FRCACacheSize directive 386
FRCAFile directive 387
FRCAFileSize directive 387
FRCANoCaching directive 387
FRCAStackName directive 388
FRCAVirtualHost directive 388
FRCAWLMParms directive 389
Front Page
accessing after installation 35

G
GET method 348
GetMask subdirective 283
getting started 41
Global Server ID for 128-bit
encryption 53
global settings for logs 73
Go Webserver Application Programming
Interface (GWAPI) directives
Authentication 313
Authorization 314
DataFilter 316
DebugToolAddr 312
Error 317
Log 317
NameTrans 313
ObjectType 314
PICSDBLookup 316
overview 311, 370
plug-ins 378
PostExit 318
PreExit 312
ServerInit 312
ServerTerm 318
Service 315
group and user IDs
access control user IDs
%%CERTIF%% 20
%%CLIENT%% 20
%%SERVER%% 20
IMWEB group ID 18
surrogate user IDs, examples of
INTERNAL 20
PRIVATE 20
PUBLIC 20
WEBADM 20

H
handing, signal 252, 259
hardware encryption 57
HEAD method 348
headers
MIME headers 347
no-parse header programs 244
hints and tips
using the Server Activity
Monitor 163
home (welcome) page 41
HostName directive 289
How do I?
act as my own certificate authority for
a private network 233
configure the server 42
correct access to administration 44
customize the Web server 18
develop servlets for the Web server 9
protect server resources 58
rate Web sites 144
serve directory listings 61
set up secure communications 61
start serving pages 41
use the Configuration and
Administration Forms 151
use the HTTP Server CA utility 236
use the key and certificate
management utility 409
how to rate a web site 149
htadm command 246
HTCodePage_t 226
htimage command 101
HTML documents
embedding CGI programs 179
HTML pages, ASCII/EBCDIC
considerations 41
HTTP
disable methods 349
enable methods 349
headers 244
HTTP method handler 203
HTTP methods
CONNECT 348
DELETE 348
GET 348
HEAD 348
OPTIONS 348
POST 348
PUT 349
TRACE 349
HTTP return codes 205
HTTP Server
installing 17
managing 105
restarting, currently running
server 257
setting up security 57
stopping 37
viewing server’s Front Page 35
HTTP Server CA utility
creating a file for the database
password 235
exporting CA keys 236
HTTP Server CA utility (continued)
maximum number of certificates 233
overview 236
processing certificate requests 237
receiving an approved certificate 238
requesting a certificate 237
requirements for using 234
HTTPD_attributes() 206
HTTPD_authenticate() 206, 215, 216
httpd command 249
httpd.conf configuration file 267
HTTPD_exec() 208
HTTPD_extract() 207
HTTPD_file() 208
HTTPD_local_security() 210
HTTPD_log_access() 207
HTTPD_log_error() 209
HTTPD_log_trace() 209
HTTPD_reverse_translate() 207
HTTPD_set() 208
HTTPD_supply_label() 208
HTTPD_translate() 207
HTTPD_write() 209

IBM Key Management Utility
(IKEYMAN)
command 418
main menu 411
migrating existing key ring files 410
setting up your system environment 410
starting 411
task reference 411
using 411
IconPath directive 302
IDINSEX tag 184
IDs
access control user IDs
%%CERTIF%% 20
%%CLIENT%% 20
%%SERVER%% 20
IMWEB group ID 18
surrogate user IDs, examples of
PRIVATE 20
PUBLIC 20
WEBADM 20
WEBADM user ID 18
WEBSRV user ID 19
IKEYMAN, key and certificate management utility 409
Imbeds directive 30
importing keys 416
IMWEB group ID 18
IMWHTTPD program 253
includes, server-side
format 94
preparing to use 94
using 93
InheritEnv directive 366
InputTimeout directive 382
install path, Web server 18, 291
installing the HTTP Server
installing for first time 18
installing the HTTP Server (continued)
migrating 18
InstallPath directive 291
integrity, secure communications 47
IP Address, Multiple 165
IP address/host name protection 51
ipmask error code 308
ipmaskproxy error code 308
K
key and certificate management utility 409
key database
changing password 413
creating 412
default name 412
registering with the server 413
setting password 413
storing password in a file 413, 418
KeyFile directive 367
keys
asymmetric 48
key pair 48
key size 55, 56
private 48
public 48
symmetric 48
L
LABEL:CURRENT 230
LABEL:FIELD 230
LABELLIST 229
LabelsFor paragraph, PICS 151, 152
language and codepage
DefaultFsCp 294
DefaultNetCp 294
Languages for CGI programs 171, 185
LDAP
configuring 155, 158
overview 156
search filters 157
storing files on 161
using with htadm command 246, 248
LDAP API 271
LDAPInclude directive 323
LDAPInfo directive 318
LDAPInfo subdirectories 319
CacheTimeout 320
ClientAuthType 321
GroupNameFilter 322
GroupSearchBase 322
Host 319
IdleConnTimeout 320
KeyFileName 322
Port 320
SearchTimeout 320
ServerAuthType 320
ServerDN 321
ServerPasswordStashFile 321
Transport 319
UserCertFilter 322
UserNameFilter 321
UserSearchBase 321
WaitToRetryConnTime 320
LiveLocalCache directive 390
Local directive 174
Log, GWAPI process step 201
prototype 204
Log directive 317
log maintenance options
access log 74
error log 77
log paths, specifying for error log 76
for proxy log 74
LogFormat directive 338
logging and reporting
changing default directory 27
directives 323
logs
access log filters 75
access log maintenance 74
configuring scenario 78
error log maintenance 77
global settings 73
overview 71
path for error logs 76
path for proxy logs 74
reports
overview 78
sample scenarios 82
update templates 81
view default templates 82
web usage mining statistics 85
with system management facilities 88
Logging and Reporting directives
AccessLog 324
AccessLogArchive 324
AccessLogExcludeMethod 326
AccessLogExcludeMimeType 326
AccessLogExcludeReturnCode 327
AccessLogExcludeURL 325
AccessLogExpire 327
AccessLogFileLimit 328
AccessReportDescription 328
AccessReportDoDnsLookup 328
AccessReportExcludeHostName 330
AccessReportExcludeMethod 331
AccessReportExcludeReturnCode 331
AccessReportExcludeURL 329
AccessReportIncludeHostName 331
AccessReportIncludeURL 329
AccessReportRoot 332
AccessReportTemplate 332
AccessReportTopList 333
AgentLog 324, 333
CacheAccessLog 334
CgiErrorLog 335
DoReporting 335
Errorlog 335
ErrorLogArchive 336
ErrorLogExpire 337
ErrorLogSizeLimit 338
FRCAAccessLog 385
LoggingReportingDebugOutput 339
LoggingReportingProgram 339
LoggingReportingProgramOptions 339
LogTime 340
LogToSyslog 340
NoLog 341

Index 485
Logging and Reporting directives (continued)
ProxyAccessLog 163, 341
RefererLog 324, 342
ReportDataArchive 344
ReportDataCompressionProgram 342
ReportDataCompressionSuffix 343
ReportDataExpire 345
ReportDataSizeLimit 344
ReportDataUnCompressionProgram 342
ReportDataUnCompressionSuffix 343
ReportProcessOldLogs 343
SMF 346
SMFRecordingInterval 346

LoggingReportingDebugOutput directive 339
LoggingReportingProgram directive 339
LoggingReportingProgramOptions directive 339
LogTime directive 340
LogToSyslog 340
Lotus Notes adapter 210

management information base (MIB), SNMP 118
managing
   keys and certificates 416
   PICS labels 145
managing PICS from central files 147
Map directive 362
mapping, resource
   Exec 359
   ExecDirPass 361
   Fail 361
   InheritEnv 366
   Map 362
   Pass 364
   Redirect 365
   Mask subdirective 284
MaxActiveThreads directive 390
MaxPersistRequest directive 390
messages, customizing
   conditions, causes, and default messages 306
   customization 305
   ErrorPage directive 305
   key word
      badredirect 307
      badrequest 306
      bascript 306
      baduser 307
      byrule 308
      defined 305
      dirbrowse 308
      dotdot 308
      ipmask 308
      ipmaskproxy 308
      methoddisabiled 308
      multifail 309
      nextel 308
      noentry 308
      noformat 311
      notallowed 308
      notauthorized 307
      notmember 307
      okredirect 306
notauthorized error code 307
notmember error code 307
nph headers 183
nph-programs 244, 246

O
object IDs, SNMP MIB 118
ObjectType, GWAPI process step 201
   prototype 203
ObjectType directive 314
okredirect error code 306
openfailed error code 308
OPTIONS method 348
OS/390 conversion 185, 187
OS/390 Debug Tool, for C/C++ GWAPI programs 226
OS/390 GWAPI information 217
OS/390 UNIX System Services authorizations for 7
   superusers 7
OutputTimeout directive 383
Overview of LDAP 156
   security concepts, options, and support 45
   X.500 directory service 156
   overview of CGI 171
   overview of GWAPI 197

P
parsing CGI information 181
   parsing CGI information on OS/390 185
   Pass directive 364
   PasswordFile subdirective 284
   password
      authorized 42
      community name, SNMP 134, 381
      files 246
      key database
         changing 413
         specifying 413
      storing in a file 413, 418
      path-based statistics for web usage
         mining 86
      performance
         recommendations on method for starting the Web server 31
         suggestions if using RACF or another SAF-based security product 4
   PICS 316
   PICS for rating services and label bureaus 146
   PICS for web site administrators 145
   PICSDBLookup GWAPI process step 201
      prototype 203
   PidFile directive 291
   planning for installation 3
   Platform for Internet Content Selection (PICS) 143
      central files 147
      creating labels 149
      DefineLBService paragraph 151, 153
      DefineService paragraph 151, 152
      definition 143
PUT, HTTP method 349
PutMask subdirective 285

Q
QUERY_STRING 182

R
RACF 4
0A2F (address bit dirty) errors 20
defining
IMWEB group ID 18
SURROGAT class support for surrogate IDs 21
WEBADM user ID 18
WEBSRV user ID 19, 23
Lotus Notes Adapter function permission 22
performance suggestions 4
planning considerations 4
program control, turning on 19
SMF permission 19
WEBSRV permission 21
Workload Management permission 22
rating
criteria for Web sites 144
Web servers 143
rating file for PICS 147
record formats, SMF configuration 137
header 137
performance 139
Recovery directive 292
Redirect directive 365
referring CGI programs in HTML 179
RefererLog directive 324, 342
remote configuration and administration 42
report filters
external hits information 80
most visted Web pages 81
reduce report scope 80
Website access information 80
ReportDataArchive directive 344
ReportDataCompressionProgram directive 342
ReportDataCompressionSuffix directive 343
ReportDataExpire directive 345
ReportDataSizeLimit directive 344
ReportDataUnCompressionProgram directive 342
ReportDataUnCompressionSuffix directive 343
reporting and logging (continued)
logs (continued)
path for proxy logs 74
reports
overview 78
sample scenarios 82
update templates 81
view default templates 82
web usage mining statistics 85
with system management facilities 88
reporting program
DoReporting directive 335
examples 82
HTLOGREP 79, 339
LoggingReportingProgram directive 339
LoggingReportingProgramOptions directive 339
templates 81
third-party, using 78, 339
Web Usage Mining 79
ReportProcessOldLogs directive 343
reports
overview 78
sample scenarios 82
update templates 81
view default templates 82
web usage mining statistics 85
resource mapping
Exec 359
ExecDirPass 361
Fail 361
InheritEnv 366
Map 362
Pass 364
Redirect 365
response generation on OS/390 187
restarting
currently running server 257
return codes from predefined functions 211
returning CGI output 184
running webusage mining from the command line 87
S
scriptio error code 310
scriptnotfound error code 310
scriptstart error code 311
ScriptTimeout directive 383
secure communications, concepts 47
Secure Sockets Layer (SSL)
client authentication 50
cipher specifications 56
concepts
authentication 47
certificate revocation list (CRL) 49
cipher specifications 56
Index 487
Security (continued)
- confidentiality 47
- digital certificate 48
- digital signature or ID 48
- Distinguished Name 49
- encryption 47
- HTTPS protocol 50
- integrity 47
- Lightweight Directory Access Protocol (LDAP) 49
- overview 47
- Public Key Infrastructure (PKI) 49
- secure communication, characteristics 47
- configuring
  - encryption settings for the server 65
  - protection for server resources 58
  - secure communications 61
  - SSL support for multiple IP addresses 65
  - encryption support 55
  - Global Server ID for banking and financial institutions 53
  - hardware encryption support 57
- HTTP Server CA utility 236
- key and certificate management utility 409
- options
  - 128-bit encryption for export browsers 53
  - overview 51
  - protecting server resources 51
  - sending different certificates from each server IP address 52
  - setting up secure connections 52
  - specifying the encryption level to be used 53
  - using S/390 cryptographic hardware 57

Security directives
- KeyFile 367
- NormalMode 368
- SSLCipherSpec 369
- SSLClientAuth 370
- SSLMode 371
- SSLPort 371
- SSLServerCert 371
- SSLV2Timeout 372
- SSLV3Timeout 373
- SSLX500CARoot 373
- SSLX500Host 373
- SSLX500Password 375
- SSLX500Port 374
- SSLX500UserID 374
- self-signed certificates, example 62
- sending information 181
- server
  - installing 17
  - stopping 37
  - viewing server’s Front Page 35

Server Activity Monitor
- access log 117
- network activity statistics 117
- server activity statistics 115
- using 113

Server Initialization, GWAPI process
- step 201
  - prototype 202
  - server request process 199
- SERVER SAF facility class 7
- server-side includes
  - format 94
  - preparing to use 94
  - using 93
- Server Termination, GWAPI process
  - step 201
  - prototype 205
- ServerID subdirective 285
- ServerInit directive 312
- ServerPriority directive 391
- ServerTerm directive 318
- Service, GWAPI process step 201
  - prototype 203
  - Service directive 315
  - service unavailable 311
  - serving directory listings 41
  - serving pages 41
  - servlet support 12
  - setup.sh command
    - common error messages 26
  - format and options 24
  - setuperror error code 309
  - signal handling 252, 259
- Simple Network Management Protocol (SNMP)
  - commands and protocol 118
  - community names 134, 381
  - e-mail address for problem reports 134, 382
  - object IDs and variable names 118
  - object IDs for management information base (MIB) 118
  - overview 117
  - password 134, 381
  - problem reports, e-mail address 134, 382
  - receiving problem reports 134, 382
  - turning support on and off 135, 381
- SMF directive 346
- SMFReportingInterval directive 346
- SNMP directive 381
- SNMPCommunityName 381
- socksifying a proxy server 163
- software requirements, SMF 137
- specifying log paths
  - for error log 76
  - for proxy log 74
  - SSL 50
  - SSL_ClientAuth subdirective 287
  - SSLCipherSpec 369
  - SSLClientAuth directive 370
  - SSLMode directive 371
  - SSLPort directive 371
  - SSLServerCert 371
  - SSLV2Timeout 372
  - SSLV3Timeout 373
  - SSLX500CARoot 373
  - SSLX500Host 373
  - SSLX500Password 375
  - SSLX500Port 374
  - SSLX500UserID 374
- starting and stopping your server
  - starting at initialization 31
  - starting automatically 31
  - starting from httpd command 31
  - starting from OE shell 31
  - starting multiple instances 32
  - stopping 37
  - starting the key management utility 411
- static documents 172, 182
- storing files on server 148
- string comparison 186
- SuffixCaseSense directive 357
- superuser 7
- SUPERUSER SAF facility class 7
- surrogate user IDs 20
- System Management directives
  - ApplEnv 375
  - ApplEnvConfig 376
  - SNMP 381
  - SNMPCommunityName 381
  - WebMasterEmail 382
- System Management Facilities (SMF)
  - controlling logging interval 136
  - logging information with 88
  - overview 135
  - record formats
    - configuration 137
    - header 137
    - performance 139
  - software requirements 137
  - turning on and off 136
  - with the console MODIFY command 136

TCP/IP
- autostarting the Web server with 23
- customizing 22
- verifying name resolution 23
- tested browsers 5
- threads
  - hints and tips 390
  - MaxActiveThreads directive 390
  - server use of 383
- time specifiers 274
- Timeout directives
  - InputTimeout 382
  - OutputTimeout 383
  - PersistTimeout 391
  - ScriptTimeout 383
- TRACE method 349
- transience 221
- troubleshooting
  - error recovery options 292
- Tuning directives
  - CacheLocalFile 384
  - CacheLocalMaxBytes 384
  - CacheLocalMaxFiles 385
  - Fast Response Cache Accelerator (FRCA) directives
    - EnableFRCA 385
    - FRCAAccessLog 385
    - FRCACacheEntries 386
    - FRCACacheOnly 386
    - FRCACacheSize 386
    - FRCAMaxFileSize 387
Tuning directives (continued)

- FRCANoCaching 387
- FRCAStackName 388
- FRCAVirtualHost 388
- FRCAWLMParms 389
- LiveLocalCache 390
- MaxActiveThreads 390
- MaxPersistRequest 390
- PersistTimeout 391
- ServerPriority 391
- UseACLs 391
- UseMetaFiles 392

U

- unknownmethod error code 307
- updating PICS configuration file 151
- UseACLs directive 391
- UseMetaFiles directive 392
- user and group IDs
  - access control user IDs
    - %%CERTIF%% 20
    - %%CLIENT%% 20
    - %%SERVER%% 20
  - IMWEB group ID 18
  - surrogate user IDs, examples of
    - INTERNAL 20
    - PRIVATE 20
    - PUBLIC 20
  - WEBADM 20
  - WEBADM user ID 18
  - WEBSRV user ID 19
- user-based statistics for web usage mining 86
- user name and password protection 51
- UserDir directive 304
- UserID directive 293
- UserID subdirective 286
- uses for CGI 172
- user/lpp/internet, default Web server install path 18, 291
- user/lpp/internet/server_root, default Web server working directory 293

V

- variable names, SNMP MIB 118
- virtual hosts 165

W

- Web site rating criteria 144
- Web Traffic Express 163
- web usage mining report
  - group-based statistics 86
  - multiple servers 88
  - overview 85
  - path-based statistics 86
  - running from the command line 87
  - user-based statistics 86
- WEBADM user ID 18, 27
- WebMasterEmail directive 382
- WebSphere Application Server documentation 475
  - migration considerations 12
  - planning considerations 9
- WEBSRV user ID 19

welcome (home) page 41

welcome page and directory directives

- AddBlankIcon 295
- AddDirIcon 296
- AddIcon 296
- AddParentIcon 297
- AddUnknownIcon 297
- AlwaysWelcome 298
- DirAccess 298
- DirReadme 299
- DirShowBrackets 299
- DirShowBytes 299
- DirShowCase 300
- DirShowDate 300
- DirShowDescription 300
- DirShowGroup 300
- DirShowHidden 301
- DirShowIcons 301
- DirShowMaxDescrLength 301
- DirShowMaxLength 301
- DirShowMinLength 301
- DirShowMode 302
- DirShowOwner 302
- DirShowSize 302
- IconPath 302
- UserDir 304
- Welcome 303

Workload Management 22

- Workload management (WLM)
  - application environments, defining 110
  - classification rules, defining 110
  - completing the configuration process 112
  - console commands 265
  - defining application environments 107
  - execution modes
    - running multiple servers 106
    - scalable server subsystem 106
    - standalone server 105
    - overview 107
  - policies, establishing 107
  - service definition, choosing 107
  - tasks after defining application environments and classification rules 112
  - using ISPF panels 107
- writing GWAPI programs 198
- writing GWAPI REXX 220
- wwwcmd 261

X

- X.500 directory service 156