z/OS

Security Server PKI Services Guide and Reference

SA22-7693-00

z/OS

Security Server PKI Services Guide and Reference

SA22-7693-00

Note

Before using this information and the product it supports, be sure to read the general information under “Notices” on page 329.

First Edition, March 2002

This edition applies to Version 1 Release 3 of z/OS (5694-A01), and to subsequent releases and modifications until otherwise indicated in new editions.

Order publications through your IBM representative or the IBM branch office serving your locality. Publications are not stocked at the address below.

IBM welcomes your comments. A form for readers' comments may be provided at the back of this publication, or you may address your comments to the following address:

International Business Machines Corporation
Department 55JA, Mail Station P384
2455 South Road
Poughkeepsie, NY 12601-5400
United States of America

FAX (United States & Canada): 1+845+432-9405
FAX (Other Countries):
Your International Access Code +1+845+432-9405

IBMLink (United States customers only): IBMUSM10(MHVRCFS)
Internet e-mail: mhvrcfs@us.ibm.com
World Wide Web: http://www.ibm.com/servers/eserver/zseries/zos/webqs.html

If you would like a reply, be sure to include your name, address, telephone number, or FAX number.

Make sure to include the following in your comment or note:

- Title and order number of this book
- Page number or topic related to your comment

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Contents

Tables

Figures

About this book

This book contains information about PKI Services, which is part of the z/OS Security Server. The Security Server includes the following components:

- DCE Security Server
- Resource Access Control Facility (RACF)
- Lightweight Directory Access Protocol (LDAP) Server, which includes client and server functions
- Network Authentication Service
- Open Cryptographic Enhanced Plug-ins (OCEP)
- PKI Services
- z/OS Firewall Technologies

This book provides the information for planning, customizing, administering, and using the PKI Services component of the Security Server. For information about other components of the Security Server, see the publications related to those components.

PKI Services provides a certificate authority for the z/OS environment and enables you to issue and administer digital certificates, so that you do not have to purchase them from an external certificate authority. This book provides you with the information you need to become productive with PKI Services. It discusses the following topics:

- Procedures for setting up PKI Services on the z/OS platform.
- Using the PKI Services administration and user Web pages, you can easily issue digital certificates to trusted parties and control whether or not a certificate is renewed or revoked.
- Guidelines to help you plan for PKI Services, such as how to integrate PKI Services components with other products installed at your site.

Who should use this book

This book should be used by those who plan, install, customize, administer, and use PKI Services. It should also be used by those who install, configure, or provide support in the following areas:

- Integrated Cryptographic Service Facility (ICSF)
- Lightweight Directory Access Protocol (LDAP)
- Open Cryptographic Enhanced Plug-ins (OCEP)
- Open Cryptographic Services Facility (OCSF)
- Resource Access Control Facility (RACF)
- z/OS
- z/OS HTTP Server
- z/OS UNIX System Services

This book assumes that you have experience with installing and configuring products in a network environment. You should be knowledgeable about the following concepts and protocols:

- Hardware installation and configuration
- Internet communications protocols, in particular Transmission Control Protocol/Internet Protocol (TCP/IP) and Secure Sockets Layer (SSL)

- Public key infrastructure (PKI) technology, including Directory schemas, the X.509 version 3 standard, and the Lightweight Directory Access Protocol (LDAP)

How to use this book

This book contains several parts:

- Part 1, “Planning” on page 1 includes the following chapters:
  - Chapter 1, “Introducing PKI Services” on page 3 introduces PKI Services, describing its basic components and related products. It also describes supported standards, certificate types, fields and extensions.
  - Chapter 2, “Planning your implementation” on page 9 provides a planning overview for your implementation. It discusses the components that work with PKI Services and the team members you will need to implement PKI Services and the skills they will need.
  - Chapter 3, “Installing and configuring prerequisite products” on page 15 describes installing and configuring related products: the z/OS HTTP Server, OCSF and OCEP, LDAP, and optionally ICSF.
- Part 2, “Configuring your system for PKI Services” on page 21 describes the tasks your team members need to perform to configure PKI Services.
  - Chapter 4, “Running IKYSETUP to perform RACF administration” on page 23 describes how the RACF administrator updates and runs IKYSETUP, a REXX exec to perform RACF administration tasks, such as setting up the daemon user ID and giving accesses.
  - Chapter 5, “Configuring the UNIX runtime environment” on page 39 explains UNIX programmer tasks including how to copy files, update environment variables, update the PKI Services configuration file, and set up the /var/pkiserv HFS directory.
  - Chapter 6, “Tailoring LDAP configuration for PKI Services” on page 49 explains how the LDAP programmer updates LDAP configuration for PKI Services.
  - Chapter 7, “Updating z/OS HTTP Server configuration and starting the server” on page 51 explains how the Web server programmer updates the z/OS HTTP Server configuration files and starts the z/OS HTTP Server.
  - Chapter 8, “Tailoring the PKI Services configuration file for LDAP” on page 55 explains how the UNIX programmer updates the LDAP section of the PKI Services configuration file.
  - Chapter 9, “Creating VSAM data sets and starting and stopping PKI Services” on page 59 explains how the MVS programmer creates VSAM data sets and starts PKI Services.
- Part 3, “Customizing PKI Services” on page 63 explains how to customize end-user and administration Web pages and advanced customization using an exit.
  - Chapter 10, “Customizing the end-user Web pages” on page 65 provides an overview of the pkiserv.tmpl file, which contains the certificate templates, and explains how to customize the end-user Web pages.
  - Chapter 11, “Customizing the administration Web pages” on page 101 provides an overview of the CGI scripts and explains how to customize the administration Web pages.
  - Chapter 12, “Advanced customization” on page 107 explains how to use certificate policies, the signature algorithm, and the PKI exit.
- Part 4, “Using PKI Services” on page 123 explains using the end-user and administration Web pages.

Chapter 13, “Using the end-user Web pages” on page 125 shows the end-user Web pages and explains how to request a certificate, obtain the certificate, and renew or revoke a certificate.

Chapter 14, “Using the administration Web pages” on page 141 shows the administration Web pages and explains how to process certificate requests and certificates.

- Part 5, “Administering RACF for PKI Services” on page 165 explains how to perform many RACF administration tasks needed for PKI Services, such as authorizing users, administering extensions, locating your PKI Services certificate and key ring, and so on.
- Part 6, “Troubleshooting” on page 181 explains using logs and utilities:
  - Chapter 16, “Using information from SYS1.LOGREC” on page 183 describes ’SYS1.LOGREC’ — which is used to record unusual runtime events, such as an exception.
  - Chapter 17, “Using information from the PKI Services logs” on page 189 discusses using the PKI Services logs to debug problems and explains how to change logging options and display log options settings.
  - Chapter 18, “Using PKI Services utilities” on page 195 explains how to use PKI Services utilities: vosview displays the entries in the VSAM ObjectStore data set (request database), and iclview displays the entries in the issued certificate list (ICL).
- Part 7, “Reference information” on page 201 provides reference information including messages and important code samples.
  - Chapter 19, “Messages” on page 203 explains PKI Services messages.
  - Chapter 20, “File directory structure” on page 219 describes product and HFS directories for PKI Services and files contained in them.
  - Chapter 21, “The pkiserv.conf configuration file” on page 221 provides a code sample of the pkiserv.conf configuration file.
  - Chapter 22, “The pkiserv.tmpl certificate templates file” on page 223 provides a code sample of the pkiserv.tmpl file.
  - Chapter 23, “Environment variables” on page 265 explains the pkiserv.envars environment variables file and provides a code sample.
  - Chapter 24, “The IKYSETUP REXX exec” on page 269 explains the contents of the IKYSETUP REXX exec that performs RACF administration and provides a code sample.
  - Chapter 25, “Other code samples” on page 287 provides additional code samples.
  - Chapter 26, “The certificate validation service” on page 295 describes the certificate validation service.
- There are several appendixes, including the following:
  - Appendix A, “LDAP directory server requirements” on page 323 explains using a non-z/OS LDAP server.
  - Appendix B, “Using a gskkyman key database for your certificate store” on page 325 explains an alternative method for setting up your key database.

Where to find more information

Where necessary, this book references information in other books. For complete titles and order numbers for all elements of z/OS, see *z/OS Information Roadmap*.

Softcopy publications

The Security Server library is available on the following CD-ROMs. The CD-ROM online library collections include the IBM Library Reader, which is a program that enables you to view the softcopy books.

SK3T-4269 *z/OS Version 1 Release 3 Collection*

This collection contains the set of unlicensed books for the current release of z/OS in both BookManager and Portable Document Format (PDF) files. You can view or print the PDF files with the Adobe Acrobat reader.

SK3T-4272 *z/OS Security Server RACF Collection*

This softcopy collection kit contains the Security Server library for z/OS in both BookManager and Portable Document Format (PDF) files. You can view or print the PDF files with the Adobe Acrobat reader.

Using LookAt to look up message explanations

LookAt is an online facility that allows you to look up explanations for z/OS messages, system abends, and some codes. Using LookAt to find information is faster than a conventional search because in most cases LookAt goes directly to the message explanation.

You can access LookAt from the Internet at:

`http://www.ibm.com/servers/eserver/zseries/zos/bkserv/lookat/lookat.html`

or from anywhere in z/OS where you can access a TSO command line (for example, TSO prompt, ISPF, z/OS UNIX System Services running OMVS).

To find a message explanation on the Internet, go to the LookAt Web site and simply enter the message identifier (for example, `IAT1836` or `IAT*`). You can select a specific release to narrow your search. You can also download code from the *z/OS Collection*, SK3T-4269 and the LookAt Web site so you can access LookAt from a PalmPilot (Palm VIIx suggested).

To use LookAt as a TSO command, you must have LookAt installed on your host system. You can obtain the LookAt code for TSO from a disk on your *z/OS Collection*, SK3T-4269 or from the LookAt Web site. To obtain the code from the LookAt Web site, do the following:

1. Go to http://www.ibm.com/servers/eserver/zseries/zos/bkserv/lookat/lookat.html.
2. Click the **News** button.
3. Scroll to **Download LookAt Code for TSO and VM.**
4. Click the ftp link, which will take you to a list of operating systems. Select the appropriate operating system. Then select the appropriate release.
5. Find the **lookat.me** file and follow its detailed instructions.

To find a message explanation from a TSO command line, simply enter: **lookat** *message-id*. LookAt will display the message explanation for the message requested.

Note: Some messages have information in more than one book. For example, IEC192I has routing and descriptor codes listed in *z/OS MVS Routing and Descriptor Codes*. For such messages, LookAt prompts you to choose which book to open.

Accessing licensed books on the Web

z/OS licensed documentation in PDF format is available on the Internet at the IBM Resource Link Web site at:

`http://www.ibm.com/servers/resourcelink`

Licensed books are available only to customers with a z/OS license. Access to these books requires an IBM Resource Link Web userid and password, and a key code. With your z/OS order you received a memo that includes this key code.

To obtain your IBM Resource Link Web userid and password log on to:

`http://www.ibm.com/servers/resourcelink`

To register for access to the z/OS licensed books:

1. Log on to Resource Link using your Resource Link userid and password.
2. Click on **User Profiles** located on the left-hand navigation bar.
3. Click on **Access Profile.**
4. Click on **Request Access to Licensed books.**
5. Supply your key code where requested and click on the **Submit** button.

If you supplied the correct key code you will receive confirmation that your request is being processed. After your request is processed you will receive an e-mail confirmation.

Note: You cannot access the z/OS licensed books unless you have registered for access to them and received an e-mail confirmation informing you that your request has been processed.

To access the licensed books:

1. Log on to Resource Link using your Resource Link userid and password.
2. Click on **Library**.
3. Click on **zSeries**.
4. Click on **Software**.
5. Click on **z/OS**.
6. Access the licensed book by selecting the appropriate element.

Other sources of information

IBM provides customer-accessible discussion areas where PKI Services and RACF may be discussed by customer and IBM participants. Other information is also available through the Internet.

IBM discussion areas

IBM provides the following discussion areas for PKI Services, RACF and security-related topics.

- **MVSRACF**

  MVSRACF is available to customers through IBM's TalkLink offering. To access MVSRACF from TalkLink:

  1. Select S390 (the S/390 Developers' Association).
  2. Use the fastpath keyword: MVSRACF.

- **SECURITY**

SECURITY is available to customers through IBM's DialIBM offering, which may be known by other names in various countries. To access SECURITY:

1. Use the CONFER fastpath option.
2. Select the SECURITY CFORUM.

Contact your IBM representative for information on TalkLink, DialIBM, or equivalent offerings for your country and for more information on the availability of the MVSRACF and SECURITY discussions.

Internet sources

The following resources are available through the Internet to provide additional information about PKI Services, RACF, and other security-related topics:

- **Online library**

  To view and print online versions of the z/OS publications, use this address:

  `http://www.ibm.com/servers/eserver/zseries/zos/bkserv/`

- **Redbooks**

  The redbooks that are produced by the International Technical Support Organization (ITSO) are available at the following address:

  `http://www.ibm.com/redbooks/`

- **Enterprise systems security**

  For more information about security on the S/390 platform, OS/390, and z/OS, including the elements that comprise the Security Server, use this address:

  `http://www.ibm.com/servers/eserver/zseries/zos/security/`

- **RACF home page**

  You can visit the RACF home page on the World Wide Web using the following address. Check this site for future possible updates regarding PKI Services.

  `http://www.ibm.com/servers/eserver/zseries/zos/racf/`

- **RACF-L discussion list**

  Customers and IBM participants may also discuss RACF on the RACF-L discussion list. RACF-L is not operated or sponsored by IBM; it is run by the University of Georgia.

  To subscribe to the RACF-L discussion and receive postings, send a note to:

  `listserv@listserv.uga.edu`

  Include the following line in the body of the note, substituting your first name and last name as indicated:

  `subscribe racf-l` *`first_name last_name`*

  To post a question or response to RACF-L, send a note, including an appropriate `Subject:` line, to:

  `racf-l@listserv.uga.edu`

- **RACF sample code**

  You can get sample code, internally-developed tools, and exits to help you use RACF. This code works in our environment, at the time we make it available, but is not officially supported. Each tool or sample has a README file that describes the tool or sample and any restrictions on its use.

  To access this code from a Web browser, go to the RACF home page and select the "Downloads" topic from the navigation bar, or go to ftp://ftp.software.ibm.com/eserver/zseries/zos/racf/.

The code is also available from `ftp.software.ibm.com` through anonymous FTP. To get access:

1. Log in as user **anonymous**.
2. Change the directory, as follows, to find the subdirectories that contain the sample code or tool you want to download:

   ```
   cd eserver/zseries/zos/racf/
   ```

An announcement will be posted on RACF-L, MVSRACF, and SECURITY CFORUM whenever something is added.

Note: Some Web browsers and some FTP clients (especially those using a graphical interface) might have problems using `ftp.software.ibm.com` because of inconsistencies in the way they implement the FTP protocols. If you have problems, you can try the following:

- Try to get access by using a Web browser and the links from the RACF home page.
- Use a different FTP client. If necessary, use a client that is based on command line interfaces instead of graphical interfaces.
- If your FTP client has configuration parameters for the type of remote system, configure it as UNIX instead of MVS.

Restrictions

Because the sample code and tools are not officially supported,

- There are no guaranteed enhancements.
- No APARs can be accepted.

To request copies of IBM publications

Direct your request for copies of any IBM publication to your IBM representative or to the IBM branch office serving your locality.

There is also a toll-free customer support number (1-800-879-2755) available Monday through Friday from 6:30 a.m. through 5:00 p.m. Mountain Time. You can use this number to:

- Order or inquire about IBM publications
- Resolve any software manufacturing or delivery concerns
- Activate the program reorder form to provide faster and more convenient ordering of software updates

Part 1. Planning

The Planning part includes the following:

- Chapter 1, “Introducing PKI Services” on page 3 provides an overview of PKI Services, its components, and related concepts.
- Chapter 2, “Planning your implementation” on page 9 provides a planning overview for your implementation, including a discussion of the components that work with PKI Services. It also discusses the team members you will need to implement PKI Services and the skills they will need.
- Chapter 3, “Installing and configuring prerequisite products” on page 15 describes installing and configuring related products: the z/OS HTTP Server, OCSF and OCEP, LDAP, and optionally ICSF.

Chapter 1. Introducing PKI Services

This chapter provides an overview of PKI Services.

It covers the following topics:

What is PKI Services?

PKI Services allows you to establish a PKI infrastructure and serve as a certificate authority for your internal and external users, issuing and administering digital certificates in accordance with your own organization's policies. Your users can use a PKI Services application to request and obtain certificates through their own Web browsers, while your authorized PKI administrators approve, modify, or reject these requests through their own Web browsers. The Web applications provided with PKI Services are highly customizable, and a programming exit is also included for advanced customization. You can allow automatic approval for certificate requests from certain users and add host IDs, such as RACF user IDs, to certificates you issue for certain users to provide additional authentication. You can also issue your own certificates for browsers, servers, and other purposes, such as virtual private network (VPN) devices, smart cards, and secure e-mail.

PKI Services supports Public Key Infrastructure for X.509 version 3 (PKIX) and Common Data Security Architecture (CDSA) cryptographic standards. It also supports the following:

- The delivery of certificates through the Secure Sockets Layer (SSL) for use with applications that are accessed from a Web browser or Web server.
- The delivery of certificates that support the Internet Protocol Security standard (IPSEC) for use with secure VPN applications or IPSEC-enabled devices.
- The delivery of certificates that support Secure Multipurpose Internet Mail Extensions (S/MIME), for use with secure e-mail applications.

What is a certificate authority?

The certificate authority, commonly called a CA, acts as a trusted third party to ensure that users who engage in e-business can trust each other. A certificate authority vouches for the identity of each party through the certificates it issues. In addition to proving the identity of the user, each certificate includes a public key that enables the user to verify and encrypt communications.

The trustworthiness of the parties depends on the trust that is placed in the certificate authority that issued the certificates. To ensure the integrity of a certificate, the certificate authority digitally signs the certificate as part of creating it, using its signing private key. Attempts to alter a certificate will invalidate the signature and render it unusable.

The protection of the certificate authority's signing private key is critical to the integrity of the certificate authority. For this reason, you should consider using ICSF to securely store your PKI Services certificate authority's private key.

As a certificate authority using PKI Services, you can do the following:

- Track certificates you issue with an issued certificate list (ICL) that contains a copy of each certificate, indexed by serial number
- Track revoked certificates using certificate revocation lists (CRLs). When a certificate is revoked, PKI Services updates the CRL during the next periodic update. Just as it signs certificates, the CA digitally signs all CRLs to vouch for their integrity.

What is PKI?

The public key infrastructure (PKI) provides applications with a framework for performing the following types of security-related activities:

- Authenticate all parties that engage in electronic transactions
- Authorize access to sensitive systems and repositories
- Verify the author of each message through its digital signature
- Encrypt the content of all communications

The PKIX standard evolved from PKI to support the interoperability of applications that engage in e-business. Its primary advantage is that it enables organizations to conduct secure electronic transactions without regard for operating platform or application software package.

The PKIX implementation in PKI Services is based on the Common Data Security Architecture (CDSA) from Intel Corporation. CDSA supports multiple trust models, certificate formats, cryptographic algorithms, and certificate repositories. Its primary advantage is that it enables organizations to write PKI-compliant applications that support their business policies.

Basic components of PKI Services and related products

Table 1. Basic components of PKI Services and related products

Administration Web application	Assists authorized administrators to review requests for certificates, approve or reject requests, renew certificates, or revoke certificates through their own Web browsers. The application consists of sample screens that you can easily customize to display your organization's logo. It also supports the following tasks: • Reviewing pending certificate requests • Querying pending requests to process those that meet certain criteria • Displaying detailed information about a certificate or request • Monitoring certificate information, such as validity period • Annotating the reason for an administrative action
End-user Web application	Guides your users to request, obtain, and renew certificates through their Web browsers. The application consists of sample screens that you can easily customize to meet your organization's needs for certificate content and standards for appearance. It offers several certificate templates that you can use to create requests for a variety of certificate types, based on the certificate's intended purpose and validity period, and supports certificate requests that are automatically approved.

Table 1. Basic components of PKI Services and related products (continued)

Exit	Provides advanced customization for additional authorization checking, validating and changing parameters on calls to the R_PKIServ callable service (IRRSPX00), and capturing certificates for further processing. You can call this exit from the PKIServ CGIs and use its IRRSPX00 pre-processing and post-processing functions. A code sample in C language code is included.
ICSF (optional)	Securely stores the PKI Services certificate authority's private signing key.
LDAP	The directory that maintains information about the valid and revoked certificates that PKI Services issues in an LDAP-compliant format. You can use an LDAP server such as z/OS Security Server LDAP.
PKI Services daemon	The server daemon that acts as your certificate authority, confirming the identities of users and servers, verifying that they are entitled to certificates with the requested attributes, and approving and rejecting requests to issue and renew certificates. It includes support for: • An issued certificate list (ICL) to track issued certificates • Certificate revocation lists (CRLs) to track revoked certificates
R_PKIServ callable service (IRRSPX00)	The application programming interface (API) that allows authorized applications, such as servers, to programmatically request the functions of PKI Services to generate, retrieve and administer certificates.
RACF (or equivalent)	Controls who can use the functions of the R_PKIServ callable service and protects the components of your PKI Services system. RACF creates your certificate authority's certificate, key ring and private key. You can also use it to store the private key, if ICSF is not available.
z/OS HTTP Server	PKI Services uses the Web server to encrypt messages, authenticate requests, and transfer certificates to intended recipients.

Component diagram

Figure 1 on page 6 shows a typical PKI Services system.

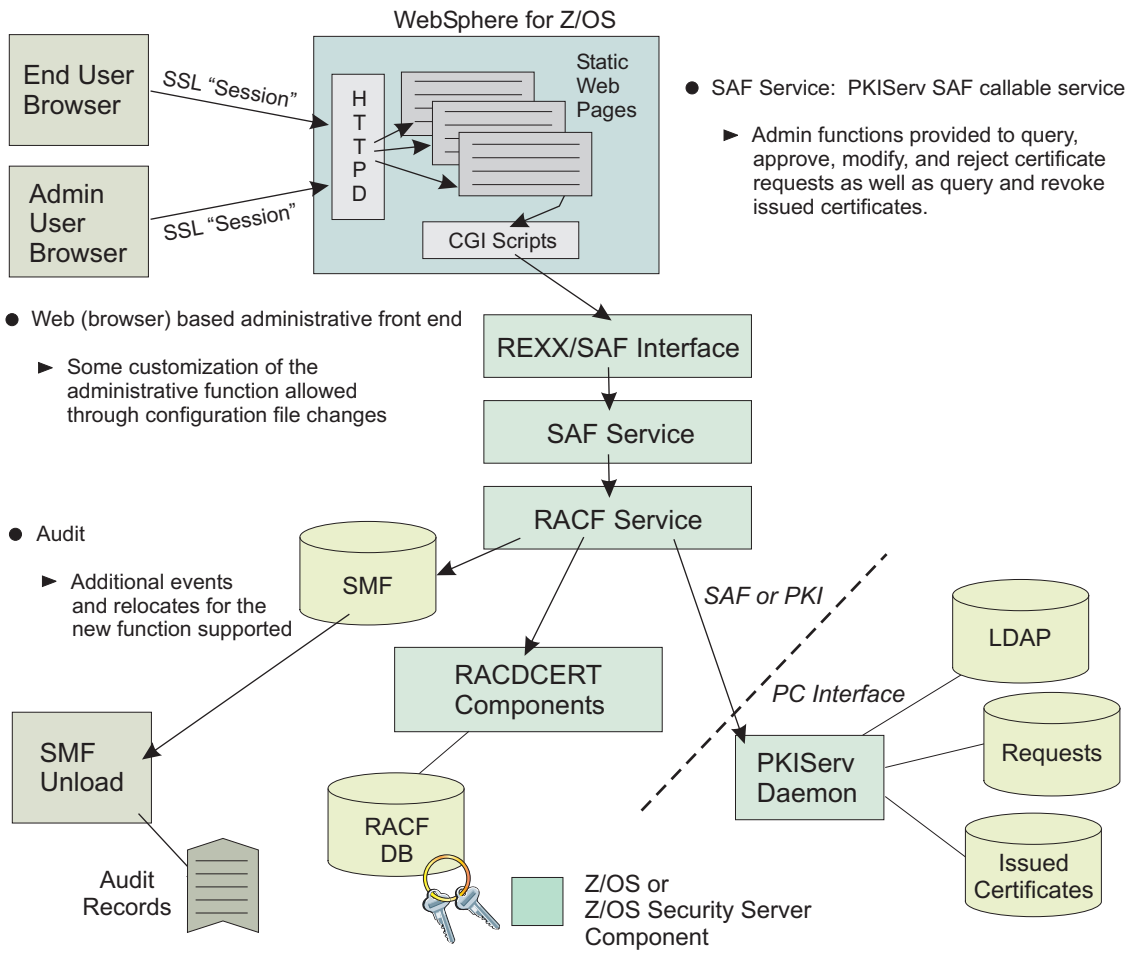


Figure 1. Component diagram of a typical PKI Services system

Supported standards

PKI Services supports the following standards for public key cryptography:

- Secure Sockets Layer (SSL) version 2 and version 3, with client authentication
- PKCS #10 browser and server certificate format, with a base64-encoded response
- IPSEC certificate format
- S/MIME certificate format
- Browser certificates for:
  - Microsoft Internet Explorer version 5.x
  - Netscape Navigator and Netscape Communicator version 4.x
- Server certificates
- LDAP standard for communications with the Directory
- X.509v3 certificates
- Certificate revocation lists (CRLv2)
- Key lengths up to 1024 bits for the CA signing private keys
- RSA algorithms for encryption and signing
- MD5 and SHA-1 hash algorithms

The LDAP standard that PKI Services supports is LDAP version 2. A directory using LDAP version 3 (with RFC 1779 syntax), is acceptable if it is backwardly compatible with version 2.

Supported certificate types

Table 2 lists the types of certificates that you can request, based on the certificate templates that are included with PKI Services. Certificate templates are samples of the most commonly requested certificate types. You can add, modify, and remove certificate templates to customize the variety of certificate types you offer to your users.

Table 2. Types of certificates you can request

Type of certificate	Use
One-year PKI SSL browser certificate	End-user client authentication using SSL
One-year PKI S/MIME browser certificate	Browser-based e-mail encryption
Two-year PKI browser certificate for authenticating to z/OS	End-user client authorization using SSL when logging onto z/OS
Five-year PKI SSL server certificate	SSL Web server certification
Five-year PKI IPSEC server (firewall) certificate	Firewall server identification and key exchange
Five-year PKI intermediate CA certificate	Subordinate (non-self-signed) Certificate Authority certification
One-year SAF browser certificate	End-user client authentication where RACF (not PKI Services) is the certificate provider
One-year SAF server certificate	Web server SSL certification where RACF (not PKI Services) is the certificate provider

Note: You can customize certificate templates to add, modify and remove certificate types.

Supported certificate fields and extensions

PKI Services certificates support most of the fields and extensions defined in the X.509 version 3 (X.509v3) standard. This support lets you use these certificates for most cryptographic purposes, such as SSL, IPSEC, VPN, and S/MIME.

PKI Services certificates can include the following types of extensions:

Standard extensions

The standard X.509v3 certificate extensions:

- authority key identifier
- basic constraints
- certificate policies
- key usage
- subject alternate name
- subject key identifier

Other extensions

Extensions that are unique to PKI Services, such as host identity mapping. This extension associates the subject of a certificate with a corresponding identity on a host system, such as with a RACF user ID.

To support your organization's policies, PKI Services also provides the means for you to customize and define certificate extensions. For example, you can change the extensions that are specified in the default certificate templates or create templates that return certificates with different extensions.

Chapter 2. Planning your implementation

The implementation of PKI Services requires the interaction of several software products, each with its own required skills. Therefore, it is important to understand the tasks involved and to plan your implementation.

This chapter provides the information you need to understand the task of implementing PKI Services, determine which skills are required to complete your implementation team, and create your own implementation plan.

This chapter covers the following topics:

Installing PKI Services

Your MVS programmer uses SMP/E to install PKI Services into an HFS directory. By default, PKI Services is installed in the /usr/lpp/pkiserv directory but the MVS programmer can determine whether to change the default for this and other directories. Before your team begins installing and configuring prerequisite products and setting up PKI Services, you will need to know which HFS directories were used so you can customize the install process.

Table 3 shows each HFS-related variable with its description and default value. Your MVS programmer should review the rightmost column of this table, crossing out any defaults that have changed and recording the correct directory names.

Table 3. HFS directory variables

Variable name	Description	Default value or customized value
variables-dir	The HFS directory where PKI Services creates working files.	**/var/pkiserv**
HFS-install-dir	The HFS directory where PKI Services is installed.	**/usr/lpp/pkiserv**
runtime-dir	The HFS directory where PKI Services looks for configuration files.	**/etc/pkiserv**

Determining prerequisite products

The installation and use of PKI Services requires the following products:

- z/OS HTTP Server
- LDAP directory server
- OCSF and OCEP
- ICSF (optional)

The installation and use of RACF, or an equivalent security product, is required.

z/OS HTTP Server

In a PKI Services system, the z/OS HTTP Server handles all requests that it receives from a Web browser. This includes requests for new certificates and requests to renew or revoke existing certificates. If needed, it performs authentication before allowing any exchange of information to take place.

z/OS HTTP Server must be installed on the same system where PKI Services is installed. SSL-enablement is required. If your HTTP server is SSL-enabled, your key file may be a RACF key ring, or a key file created by another product. For more information, see “Steps for installing and configuring the z/OS HTTP Server to work with PKI Services” on page 15.

LDAP directory server

Use of an LDAP server is required to maintain information about PKI Services certificates in a centralized location. The z/OS LDAP Server is recommended, but a non-z/OS LDAP server may be used if it can support the objectclasses and attributes used by PKI Services. Typical PKI Services usage requires an LDAP directory server that supports the LDAP (Version 2) protocol (and the PKIX schema), such as IBM z/OS LDAP. If you intend to use the z/OS LDAP server, you must configure it to use the TDBM backend.

Through the integration of IBM z/OS LDAP with DB2, the directory can support millions of directory entries. It also allows client applications, such as PKI Services, to perform database storage, update, and retrieval transactions. For more information, see “Steps for installing and configuring LDAP” on page 18.

OCSF and OCEP

PKI Services requires OCSF and OCEP to be installed and configured so that the user ID under which the PKI Services daemon runs can use required services. For more information, see “Installing and configuring OCSF and OCEP” on page 17.

ICSF (optional)

ICSF is recommended but not required. You can begin using PKI Services without installing ICSF and install it later without reinstalling PKI Services. ICSF is strongly recommended to store and protect your certificate authority’s private key. For more information, see “Installing and configuring ICSF (optional)” on page 20.

Identifying skill requirements

The implementation of PKI Services requires the interaction of several software products, each with its own required skills. This means that your team may consist of people from several different disciplines, particularly if you work with a large organization.

This section provides the information you need to determine which skills are required to complete your implementation. These skills are presented in terms of job titles for people who specialize in those skills. For example, a task requiring MVS skills is referred to as a task for an MVS programmer. Therefore, if some of your team members have multiple skills, you may require fewer individuals to complete your team.

Team members

Your team for installing and configuring prerequisite products and setting up PKI Services should include the following members:

- ICSF programmer
- LDAP programmer
- MVS programmer
- OCSF and OCEP programmer
- RACF administrator
- UNIX programmer
- Web server programmer

You may wish to include a Web page designer to customize your PKI Services Web applications. This task is listed in the chapter as a task for a Web server programmer.

One or more PKI administrators are needed to manage your ongoing operation as a certificate authority, once your PKI Services system is set up. The responsibilities of these administrators include approving, modifying and rejecting certificate requests and revoking certificates. It may be advisable to appoint a PKI administrator early, and involve this person in your planning.

Attention: PKI Services administrators play a very powerful role in your organization. The decisions they make when managing certificates and certificate requests determine who will access your computer systems and what privileges they will have when doing so. IBM recommends that you give this authority to only those individuals whom you trust with the RACF SPECIAL attribute. For more information on the RACF SPECIAL attribute, see the *z/OS Security Server RACF Security Administrator's Guide*.

Skills for setting up prerequisite products

The following table lists team members (alphabetically) and tasks and required skills needed for installing and configuring prerequisite products:

Table 4. Tasks and skills needed for installing prerequisite products

Role	Tasks	Required Skills	Documented in:
ICSF programmer	(Optionally) installing and configuring ICSF (if not already done)	ICSF installation and configuration skills	• *z/OS ICSF Administrator's Guide* • *z/OS ICSF Application Programmer's Guide* • *z/OS ICSF System Programmer's Guide*
LDAP programmer	Installing and configuring LDAP (if not already done) and recording information	LDAP installation and configuration skills	• *z/OS Security Server LDAP Server Administration and Use*
OCSF and OCEP programmer	Installing and configuring OCSF and OCEP (if not already done) and recording information	OCSF and OCEP installation and configuration skills	• *z/OS Open Cryptographic Services Facility Application Programming* • *z/OS SecureWay Security Server Open Cryptographic Enhanced Plug-ins Application Programming*

Table 4. Tasks and skills needed for installing prerequisite products (continued)

Role	Tasks	Required Skills	Documented in:
Web server programmer	Installing and configuring the z/OS HTTP Server (if not already configured for at least non-SSL pages) and recording information	z/OS HTTP Server installation and configuration skills	• *z/OS HTTP Server Planning, Installing, and Using*

Your team needs to install and configure prerequisite products before setting up PKI Services:

1. The Web server programmer installs and configures the z/OS HTTP Server.
2. The OCSF and OCEP programmer installs and configures the OCSF and OCEP.
3. The LDAP programmer installs and configures LDAP.
4. Optionally, the ICSF programmer install and configures ICSF.

See Chapter 3, “Installing and configuring prerequisite products” on page 15 for details about performing these tasks.

Skills for setting up PKI Services

The following table lists team members (alphabetically) and the tasks and skills needed for setting up PKI Services:

Table 5. Roles, tasks, and skills for setting up PKI Services

Role	Tasks	Required Skills	Documented in:
LDAP programmer	• Customizes LDAP configuration for PKI Services	• LDAP customization skills	• *z/OS Security Server LDAP Server Administration and Use*
MVS programmer	• Creates VSAM object store and ICL data sets and indexes • Starts the PKI Services daemon	• Basic MVS skills – Editing a data set – ISPF COPY command – MVS console start command • JCL knowledge to change job card • Basic Web and browser skills • REXX skills (for working with IKYSETUP REXX exec)	• *z/OS MVS System Commands*

Table 5. Roles, tasks, and skills for setting up PKI Services (continued)

Role	Tasks	Required Skills	Documented in:
RACF administrator	• Adds groups and user IDs • Sets up access control • Creates certificates • Sets up daemon security	• RACF administration • RACF commands such as the following: – ADDGROUP – ADDSD – ADDUSER – RACDCERT – RDEFINE – PERMIT – SETROPTS • TSO skills	• *z/OS TSO/E REXX Reference* • *z/OS UNIX System Services Planning* • *z/OS Security Server RACF Security Administrator's Guide*
UNIX programmer	• Copies files • (Optionally) customizes environment variables • (Optionally) customizes (non-LDAP sections of) pkiserv.conf configuration file • Sets up /var/pkiserv directory • Updates the LDAP section of the pkiserv.conf configuration file	• Basic UNIX commands, such as the **cp** (copy) command • Getting superuser authority	• *z/OS UNIX System Services Command Reference* • *z/OS UNIX System Services Planning*
Web server programmer	• Helps set up PKI Services – Updates the z/OS HTTP Server's configuration files – Starts the z/OS HTTP Server • Customizes the PKI Services Web pages	• z/OS HTTP Server customization skills • Editing configuration files • Customizing the PKI Services Web pages	• *z/OS HTTP Server Planning, Installing, and Using*

Creating an implementation plan

Your implementation plan should include major subtasks, responsible parties, and a realistic estimate of time and effort required. The major tasks for implementing PKI Services are provided here as a basis for you to build your own plan.

Task roadmap for implementing PKI Services

Table 6 shows the subtasks and associated procedures for implementing PKI Services. These tasks will comprise the major part of your implementation plan.

Table 6. Task roadmap for implementing PKI Services

Subtask	Associated procedure (See ...)
Installing and configuring prerequisite products:	Chapter 3, "Installing and configuring prerequisite products" on page 15

Table 6. Task roadmap for implementing PKI Services (continued)

Subtask	Associated procedure (See ...)
• z/OS HTTP Server	• "Steps for installing and configuring the z/OS HTTP Server to work with PKI Services" on page 15
• OCSF and OCEP	• "Steps for installing and configuring OCSF and OCEP to work with PKI Services" on page 17
• LDAP directory server	• "Steps for installing and configuring LDAP" on page 18
• ICSF (optional)	• "Installing and configuring ICSF (optional)" on page 20
Configuring your system for PKI Services:	Part 2, "Configuring your system for PKI Services" on page 21
• RACF	• Chapter 4, "Running IKYSETUP to perform RACF administration" on page 23
• z/OS UNIX	• Chapter 5, "Configuring the UNIX runtime environment" on page 39
• LDAP configuration	• Chapter 6, "Tailoring LDAP configuration for PKI Services" on page 49
• z/OS HTTP Server	• Chapter 7, "Updating z/OS HTTP Server configuration and starting the server" on page 51
• LDAP	• Chapter 8, "Tailoring the PKI Services configuration file for LDAP" on page 55
• VSAM	• Chapter 9, "Creating VSAM data sets and starting and stopping PKI Services" on page 59
Customizing PKI Services:	Part 3, "Customizing PKI Services" on page 63
• Customizing end-user Web pages	• Chapter 10, "Customizing the end-user Web pages" on page 65
• Customizing administration Web pages	• Chapter 11, "Customizing the administration Web pages" on page 101
• Advanced customizing	• Chapter 12, "Advanced customization" on page 107
Testing PKI Services:	Part 4, "Using PKI Services" on page 123
• Using end-user Web pages	• Chapter 13, "Using the end-user Web pages" on page 125
• Using administration Web pages	• Chapter 14, "Using the administration Web pages" on page 141
Administering PKI Services:	Part 5, "Administering RACF for PKI Services" on page 165
• RACF	• Chapter 15, "RACF administration for PKI Services" on page 167

Chapter 3. Installing and configuring prerequisite products

After the MVS programmer installs PKI Services using SMP/E (but before team members set up PKI Services — see Chapter 4, “Running IKYSETUP to perform RACF administration” on page 23 through Chapter 9, “Creating VSAM data sets and starting and stopping PKI Services” on page 59), your team needs to set up prerequisite products:

- z/OS HTTP Server
- OCSF and OCEP
- LDAP
- ICSF (optional)

Tasks to perform before setting up PKI Services

Before you can set up PKI Services, your team needs to set up prerequisite software products by completing the following tasks, if not already done:

1. “Installing and configuring the z/OS HTTP Server”
2. “Installing and configuring OCSF and OCEP” on page 17
3. “Installing and configuring LDAP” on page 17
4. “Installing and configuring ICSF (optional)” on page 20

This chapter explains these tasks in more detail.

Installing and configuring the z/OS HTTP Server

PKI Services requires that you have the z/OS HTTP Server installed and configured for at least non-SSL page retrieval. (Tasks of other team members, such as the RACF administrator and Web server programmer — see Chapter 4, “Running IKYSETUP to perform RACF administration” on page 23 and Chapter 7, “Updating z/OS HTTP Server configuration and starting the server” on page 51 — assume that this is already done.)

Steps for installing and configuring the z/OS HTTP Server to work with PKI Services

Before you begin:

1. You will need Web server programming skills to complete this procedure.
2. You may need to refer to the following publication:

   z/OS HTTP Server Planning, Installing, and Using

Perform the following steps to install and configure the z/OS HTTP Server to work with PKI Services:

1. Use the following table to decide what you need to do:

If ...	Then ...	Notes
The z/OS HTTP Server is not installed and configured...	Install and configure z/OS HTTP Server by following the instructions in the installation section of *z/OS HTTP Server Planning, Installing, and Using*.	**Recommendation:** For PKI Services, when you install the z/OS HTTP Server, do not use a password file.

If ...	Then ...	Notes
The z/OS HTTP Server is installed but not configured for SSL...	Fill in the missing values in the table in the next step. (The RACF programmer needs information for setting up PKI Services; see Chapter 4, “Running IKYSETUP to perform RACF administration” on page 23.)	
The z/OS HTTP Server is installed and configured for SSL using a RACF key ring...	Fill in the missing values in the table in the next step. (The RACF programmer needs information for setting up PKI Services; see Chapter 4, “Running IKYSETUP to perform RACF administration” on page 23.)	
The z/OS HTTP Server is installed and configured for SSL using gskkyman...	Fill in the missing values in the table in the next step. (The RACF programmer needs information for setting up PKI Services; see Chapter 4, “Running IKYSETUP to perform RACF administration” on page 23. The RACF programmer also needs to add your CA certificate to an existing keyfile; see Appendix B, “Using a gskkyman key database for your certificate store” on page 325 for information about gskkyman steps.)	

You can now perform the steps for the decision you have made.

---

2. Fill in the rightmost column of the following table with information from the configuration:

Table 7. z/OS HTTP Server information you need to record

z/OS HTTP Server information	Explanation	Value
z/OS HTTP Server fully qualified domain name	A fully qualified domain name is the name of a host system. It includes a series of subnames (each of which is a domain name). For example, `ralvm7.vnet.ibm.com` is a fully qualified domain name that includes the domain names `ibm.com` and `vnet.ibm.com`. (The RACF administrator needs to know the fully qualified domain name when setting up PKI Services.)	
The full UNIX pathname of your httpd.conf configuration file.	(The Web server programmer needs to know the full UNIX pathname when updating the httpd.conf configuration file to support PKI Services.)	

Installing and configuring OCSF and OCEP

PKI Services requires OCSF and OCEP to be installed and configured so that the user ID under which the PKI Services daemon runs can use required services. The OCSF and OCEP programmer also needs to record some information.

Steps for installing and configuring OCSF and OCEP to work with PKI Services

Before you begin:

1. Although the base feature of z/OS includes OCSF and ICSF, if you are in the United States or Canada, make sure you have ordered and installed the additional OCSF Security Level 3 feature. (There is no charge for this feature.)
2. You will need OCSF and OCEP programming skills to complete this procedure.
3. You may need to refer to the configuration information in the following publications:
   - *z/OS Open Cryptographic Services Facility Application Programming*
   - *z/OS SecureWay Security Server Open Cryptographic Enhanced Plug-ins Application Programming*

   These books contain:
   - Instructions on how to set up the necessary security authorizations using RACF
   - Information on the RACF program control definitions necessary for OCSF
   - Instructions on how to run the installation scripts necessary to use OCSF and OCEP.

Perform the following steps to install and configure OCSF and OCEP to work with PKI Services:

1. If OCSF and OCEP are not already installed and configured, follow the instructions for how to do so in the previously listed publications.
2. If the value set for the registry directory differs from the default of '/var/ocsf', record the new value in the following table. (If it differs from the default, the UNIX programmer will need to update the OCSFREGDIR environment variable in the PKI Services environment variables file, pkiserv.envars.)

Table 8. OCSF information you need to record

OCSF information	Explanation	Default value or customized value
Value set for the registry directory	This is the location of the OCSF registry. The default is '/var/ocsf'.	**'/var/ocsf'**

A later chapter, Chapter 26, "The certificate validation service" on page 295, provides information about the PKI Services OCSF Trust Policy, PKITP. For information about configuring this, see "Configuring and getting started with PKITP" on page 299.

Installing and configuring LDAP

The LDAP programmer installs and configures LDAP for the TDBM DB2 backend and records values that will be needed later.

Steps for installing and configuring LDAP

Although it may be configured otherwise, typical PKI Services usage requires access to an LDAP directory server. Install the LDAP directory server separately from PKI Services. After the installation is complete, LDAP needs to be configured for PKI Services. The directory stores issued certificates and certification revocation lists. The z/OS LDAP Server is recommended but not required. You can use a non-z/OS LDAP server if it can support the object classes and attributes that PKI Services uses. For information about using a non-z/OS LDAP server, see Appendix A, “LDAP directory server requirements” on page 323. The remainder of this chapter assumes you will use the z/OS LDAP Server.

Before you begin:

1. You will need LDAP programming skills to complete this procedure.
2. You will need to refer to the following publication:

   z/OS Security Server LDAP Server Administration and Use

Perform the following steps to install and configure LDAP to work with PKI Services:

1. Use the following table to decide what you need to do:

If ...	Then...	Notes
You do not have LDAP installed and configured...	Follow the instructions in the Administration section of *z/OS Security Server LDAP Server Administration and Use*.	**Note:** It is not necessary to set up the LDAP server for SSL because PKI Services does not use SSL to communicate with the LDAP server.
You have LDAP installed and configured but not for the TDBM backend...	You need to migrate to the TDBM backend. See *z/OS Security Server LDAP Server Administration and Use* for details about how to do this.	
You have LDAP installed and configured for the TDBM backend...	Go to the next step.	

   You can now perform the steps for the decision you have made.

   _____

2. Record the values from the LDAP configuration step in the following table. (Your team will need this information when setting up PKI Services.)

Table 9. LDAP information you need to record

LDAP information	Explanation	Value

Administrator's distinguished name	This is the distinguished name to use for LDAP binding. A distinguished name is the unique name of a data entry that identifies its position in the hierarchical structure of the directory. A distinguished name consists of the relative distinguished name (RDN) concatenated with the names of its ancestor entries. For example, an entry for Tim Jones could have an RDN of `CN=Tim Jones` and a DN of:
	`CN=Tim Jones,O=IBM,C=US`
	
	CAs typically have distinguished names in the following form:
	`OU=your-CA's-friendly-name,O=your-organization, C=your-country-abbreviation`
	
	The LDAP administrator defines the administrator's distinguished name with the **adminDN** keyword in the /etc/ldap/slapd.conf configuration file. For example, the value is ″cn=Admin″ in the following:
	`adminDN="cn=Admin"`	

Administrator password	This is the password to use for LDAP binding. The LDAP programmer can set this in several ways, for example: • By specifying the password as a TDBM entry by using the **userPassword** attribute in the **ldif2tdbm** load utility • (Not recommended) by using the **adminPW** keyword in the slapd.conf configuration file.	
LDAP fully qualified domain name and port	This is the IP address and port on which the LDAP server is listening. For example, for `ldap.widgets.com:389`, the fully qualified domain name is `ldap.widgets.com` and the port is `389`. See Table 7 on page 16 for a definition of fully qualified domain name.	

Suffix	A suffix in LDAP is the top-level name of the subtree. For example, for the following distinguished name:
	`OU=your-CA's-friendly-name,O=your-organization, C=your-country-abbreviation`
	
	the suffix could be either "`O=your-company,C=your-country-abbreviation`" or "`C=your-country-abbreviation`".
	
	The suffix value is specified after the suffix keyword in the slapd.conf file:
	`suffix "O=your-company,C=your-country-abbreviation"`
	
	Note: If you have more than one suffix, record the suffix you intend to use as the root for storing the PKI Services CA certificate.	

3. The chapters that follow require the LDAP server to be running. Follow the instructions in the chapter about running the LDAP server in *z/OS Security Server LDAP Server Administration and Use*.

Installing and configuring ICSF (optional)

Using ICSF is recommended but not required. RACF can use ICSF's Public Key Data Set (PKDS) to securely store the PKI Services CA signing key if directed to do so. For this to be successful, the ICSF programmer must install and configure ICSF for Public Key Algorithms (PKA), and ICSF must be running. (The RACF administrator uses the IKYSETUP REXX exec to set up any RACF profiles needed to control access to ICSF services and keys. For more information, see Chapter 4, "Running IKYSETUP to perform RACF administration" on page 23.)

Note: You do not have to choose whether or not to install ICSF and perform the installation and configuration at this point. You can do so later in the process.

Before you begin:

- You will need ICSF programming skills to complete this procedure.
- You may need to refer to the following publication:

  z/OS ICSF Administrator's Guide

  This publication provides information about managing cryptographic keys, setting up and maintaining the PKDS, controlling who can use cryptographic keys and services, and general information about ICSF and cryptographic keys.

If ICSF is not already installed and configured for PKA, do this by following the instructions in *z/OS ICSF Administrator's Guide*.

Part 2. Configuring your system for PKI Services

After the MVS programmer installs PKI Services into the HFS directory, your team needs to perform additional tasks to configure PKI Services, including the following:

- Chapter 4, “Running IKYSETUP to perform RACF administration” on page 23 describes how the RACF administrator updates and runs IKYSETUP, a REXX exec to perform RACF administration tasks, such as setting up the daemon user ID and giving accesses.
- Chapter 5, “Configuring the UNIX runtime environment” on page 39 explains:
  - Copying files, such as the PKI Services configuration file
  - Updating environment variables
  - Updating the PKI Services configuration file
  - Setting up the /var/pkiserv HFS directory.
- Chapter 6, “Tailoring LDAP configuration for PKI Services” on page 49 explains how to update your LDAP configuration (performed earlier — see “Installing and configuring LDAP” on page 17) for PKI Services.
- Chapter 7, “Updating z/OS HTTP Server configuration and starting the server” on page 51 describes updating the z/OS HTTP Server configuration files and starting the z/OS HTTP Server.
- Chapter 8, “Tailoring the PKI Services configuration file for LDAP” on page 55 explains how to update the LDAP section of the PKI Services configuration file.
- Chapter 9, “Creating VSAM data sets and starting and stopping PKI Services” on page 59 explains how to create VSAM data sets and start PKI Services.

Chapter 4. Running IKYSETUP to perform RACF administration

PKI Services provides SYS1.SAMPLIB(IKYSETUP), a REXX exec, to perform RACF administration tasks for setting up PKI Services. The RACF administrator updates and runs this REXX exec, which issues RACF commands to perform the following tasks:

- Adding groups and user IDs
  - Setting up the PKI Services administration group
  - Creating the PKI Services daemon user ID
  - Giving appropriate access to the RACF group
  - Creating the surrogate user ID and giving the surrogate user ID authority to generate certificates

    (A surrogate user ID is the identity assigned to client processes when they are requesting certificate services. A surrogate user ID is required for external clients. For simplicity IBM recommends that you use surrogate user IDs for internal clients as well, rather than allowing them to access PKI Services under their own identities.)
  - Associating the user ID with the PKI Services started procedure.
- Setting up access control to protect end-user and administrative functions of PKI Services:
  - Authorizing the PKI Services daemon user ID for CA functions
  - Giving administrators access to VSAM data sets
  - Optionally authorizing PKI Services for ICSF resources.
- Creating CA and SSL certificates:
  - Creating a CA certificate and private key
  - Backing them up to a password-protected MVS data set
  - Optionally migrating the private key to ICSF
  - Creating a SAF key ring and associating it with the certificate
  - Exporting the CA certificate to an MVS data set and HFS file
  - Generating a server certificate signed by the new CA
  - Creating a key ring for the Web server
  - Associating the Web server and any trusted CA certificates to the key ring.
- Setting up the z/OS HTTP Server for surrogate operation.

Overview of IKYSETUP

IKYSETUP consists of several parts:

- Configurable section — This section assigns values to variables.
- A section that issues RACF commands to perform RACF administration tasks (see “Actions IKYSETUP performs by issuing RACF commands” on page 269 for details about the actions that various sections of code perform)
- A section that writes information (such as the name of the PKI Services administration group) to the log data set. The log itself consists of two parts: commands issued and other information. (See Figure 3 on page 36.)

Note: By default, IKYSETUP creates the log. You can disable recording information to the log by changing the value of one of the variables in IKYSETUP (`log_dsn`) to null.

The configurable section contains three parts:

- Values you must change (by making them specific to your company, such as your company's name)
- Values you might change depending on how you want PKI Services set up (for example, whether your setup will include ICSF)
- Values you can optionally change (these defaults are acceptable without change, but you might want to change them to make them more specific to your company, for example the name of the PKI Services administration group, which by default is PKIGRP)

The following table illustrates the structure and divisions of IKYSETUP:

Table 10. IKYSETUP — Structure and divisions

Configurable section — assigns values to variables • Values you must change to customize (see Table 11 on page 25) • Values you might change that are related to setup (see Table 16 on page 29) • Values you can optionally change (see Table 17 on page 32)
Issues RACF commands
Records information in the log data set

Before you begin

You need to collect the following publications:

- *z/OS Security Server RACF Command Language Reference*
- *z/OS Security Server RACF Security Administrator's Guide*
- *z/OS TSO/E REXX Reference*

The RACF administrator needs to decide the values of variables in IKYSETUP and to record these values for future reference. Review and update as necessary the following three variables tables.

Note: There are three tables because there are three categories of variables:

- Variables whose values you are **required** to change, such as ones containing your company name
- Variables whose values you might want to change, depending based on how you are setting up PKI Services
- Variables whose values you can optionally change.

There is some overlap between the three types of variables, for example, if you are already using the RACF sample Web application, PKISERV.

Recommendation: If you are running IKYSETUP for the first time, to get PKI Services up and running quickly, you can complete only the following:

- Table 11 on page 25
- Table 15 on page 28
- The rows of Table 16 on page 29 concerning z/OS UNIX level security:
  - unix_sec

– (If z/OS level security is already set up:) bpx_userid. and pgmcntl_dsn.

Variables whose values must change

Fill in the blank lines in the rightmost column with your company's information (and cross out the defaults in these cells).

Table 11. IKYSETUP variables whose values must change

Variable name	Description	Referenced elsewhere	Default value and your company's information
`ca_dn`	The CA's distinguished name. (For a definition of distinguished name, see Table 9 on page 19.) If you already have your CA certificate and private key set up in RACF, set `ca_dn=""`, set `ca_label` (in the following row) to the value of your CA's label, and update `ca_expires` and `web_expires` (in Table 17 on page 32) to reflect the expiration date of your CA certificate. If you do not already have your CA certificate and private key set up in RACF, cross out the default in the rightmost cell of this row and record the information for your company-specific information for distinguished name on the blank line.	The suffix of the PKI Services CA's distinguished name must match the LDAP suffix. (The LDAP suffix is in the LDAP configuration file, slapd.conf. See Table 9 on page 19 for a definition of suffix.)	**OU('Human Resources Certificate Authority')** **O('Your Company')** **C('Your Country 2 Letter Abbreviation')** _____
`ca_label`	The CA certificate label. If you already have your CA certificate and private key set up in RACF (and your CA certificate's label differs from the default), you need to set `ca_label` to your CA certificate's label.	No	**Local PKI CA** (Replace this default if you already have your CA certificate and private key set up in RACF.) _____
`daemon_uid`	The z/OS UNIX user identifier (UID) associated with the PKI Services daemon user ID.	No	**554** _____
`pki_gid`	The z/OS UNIX group identifier (GID) for the PKI Services administration group.	No	**655** _____

Table 11. IKYSETUP variables whose values must change (continued)

Variable name	Description	Referenced elsewhere	Default value and your company's information
pkigroup_mem.	Members of the PKI Services administration group are responsible for administering PKI Services functions. **Note:** IBM recommends that you restrict PKI Services administration tasks to those with the RACF SPECIAL attribute. See page 11 for more information. pkigroup_mem. is a list in which pkigroup_mem.0 is the number of members in the list and the rest of the entries are their user IDs. You must change the pkigroup_mem.0 to at least 1, and change pkigroup_mem.1 through pkigroup_mem.n to the member user IDs.	No	**0** (default for pkigroup_mem.0, the number of member user IDs) _____ **Note:** You must change the default to at least 1. (Record the member IDs:) _____ _____ _____ _____ _____
surrog_uid	The UID associated with the surrogate user ID.	No	**555** _____
web_dn	Your Web server's distinguished name. (For a definition of distinguished name, see Table 9 on page 19.) **Notes:** 1. The RACF administrator copies the fully qualified domain name from an earlier table: Table 7 on page 16. 2. If you already have your Web server configured for SSL: • Set web_dn="" • Update the web_ring row (You need to connect your PKI Services CA certificate to your key ring. See the web_ring row for directions.)	The value of the Web server's common name (CN), which is your server's symbol IP address, for example, www.*yourcompany*.com), must match your Web server's fully qualified domain name.	**CN('www.YourCompany.com')** **O('Your Company')** **L('Your City')** **SP('Your Full State or Province Name')** **C('Your Country 2 Letter Abbreviation')** _____

Table 11. IKYSETUP variables whose values must change (continued)

Variable name	Description	Referenced elsewhere	Default value and your company's information
web_ring	The name of the Web server's SAF key ring. If your Web server is configured for SSL and you are using a RACF key ring, set web_ring to the value of the RACF key ring. If your Web server is configured for SSL and you are using gskkyman, set web_ring="" and see Appendix B, "Using a gskkyman key database for your certificate store" on page 325 for additional directions.	httpd*.conf - KeyFile directive	**SSLring** _____

Variables whose values may change depending on setup

To help in completing the next table of variables (see Table 16 on page 29) fill out the following four decision tables:

Decision table for restrict_surrog

Use the following decision table to determine the value of restrict_surrog in Table 16 on page 29. The restrict_surrog variable determines if the RESTRICTED attribute is assigned to the surrogate user ID. The RESTRICTED attribute limits the resources available to this user ID.

Recommendation: By default, IKYSETUP does not assign the RESTRICTED attribute to the surrogate user ID. IBM recommends that you do not change the default the first time you run IKYSETUP (but do change it before going into a production environment). For more information, see the chapter about defining groups and users in *z/OS Security Server RACF Security Administrator's Guide*.

Table 12. Deciding the value of restrict_surrog

If ...	Then ...
You want to assign the RESTRICTED attribute to the surrogate user ID	Set restrict_surrog=1
You do **not** want to assign the RESTRICTED attribute to the surrogate user ID	Do not change the default restrict_surrog=0

Decision table for use_icsf

Use the following decision table to determine the value of use_icsf in Table 16 on page 29. The use_icsf variable determines whether you are using ICSF for private key protection.

Recommendation: By default, IKYSETUP does not use ICSF. IBM recommends that you do not change the default the first time you run IKYSETUP but that you change it before going into a production environment. (For information about installing and configuring ICSF, see "Installing and configuring ICSF (optional)" on page 20.)

Table 13. Deciding the value of use_icsf

If ...	Then ...
You want to use ICSF for private key protection	Set use_icsf=1 You also need to review and possibly change the following additional variables in Table 16 on page 29: • csfkeys_profile • csfserv_profile • csfusers_grp
You do **not** want to use ICSF	Do not change the default use_icsf=0

Decision table for key_backup

Use the following decision table to determine the value of key_backup in Table 16 on page 29. The key_backup variable determines whether the PKI Services CA certificate and private key should be backed up to an encrypted data set.

Table 14. Deciding the value of key_backup

If ...	Then ...	Notes
You want to back up your CA's certificate and private key to a passphrase encrypted data set	Do not change the default key_backup=1	**Note:** When you use IKYSETUP, you need to enter a passphrase whose display is not inhibited — it appears on the screen in the clear.
You do **not** want to back up your CA's certificate and private key to a passphrase encrypted data set	Set key_backup=0	

Decision table for unix_sec

Use the following decision table to determine the value of unix_sec in Table 16 on page 29. The unix_sec variable determines whether you want to use z/OS UNIX security, which is a higher level of security. z/OS UNIX provides two levels of security:

UNIX level security
: This is a less stringent level of security than z/OS UNIX level security. It is for installations where system programmers have been granted superuser authority. Programs that run with superuser authority have daemon level authority and can issue MVS identity-changing services without entering a _passwd() for the target user ID. With this level of security, the BPX.DAEMON profile in the FACILITY class is not defined.

z/OS UNIX level security
: This is a higher level of security than z/OS UNIX level security. It lets your system exercise more control over superusers. With this level of security, the BPX.DAEMON profile in the FACILITY class is defined.

Table 15. Deciding the value of unix_sec

If...	Then ...	Notes
You already have z/OS UNIX security set up	Set unix_sec=1	–

Table 15. Deciding the value of unix_sec (continued)

If...	Then ...	Notes
You do not have z/OS UNIX security set up and you do not want to set it up	Do not change the default of unix_sec=0	–
You do not have z/OS UNIX security set up and you want to set it up for the first time	Set unix_sec=2	**Notes:** 1. For information about additional manual configuration, see the section about establishing UNIX security in the *z/OS UNIX System Services Planning*. 2. If you are setting unix_sec=2, you also need to review and possibly update the following variables: • bpx_userid. • pgmcntl_dsn.

Update the following table based on your answers in the preceding decision tables. If you have decided to change any of the defaults in the rightmost column, cross out the defaults and enter your company's information:

Table 16. IKYSETUP variables you might want to change depending on setup

Variable name	Description	Referenced elsewhere	Default value or your company's information
`bpx_userid.`	A list of user IDs with daemon and server authority. The `bpx_userid.0` is the number of items in the list and the rest of the entries are the bpx user IDs. (This is non-applicable if `unix_sec ¬=2`.)	No	**1**(default for number of items) **OMVSKERN**
`csfkeys_profile`	A profile to protect the PKI Services key in ICSF. (This is non-applicable if `use_icsf= 0`.) If you do not want IKYSETUP to create the profile, set `csfkeys_profile=""`. **Note:** When RACF stores the private key in the PKDS, it generates the label as: `'IRR.DIGTCERT.CERTIFAUTH.`*`unique-time-stamp`*`'`	No	**IRR.DIGTCERT.CERTIFAUTH.***
`csfserv_profile`	A profile to protect ICSF services. (This is non-applicable if `use_icsf=0`.)	No	**CSF***

Table 16. IKYSETUP variables you might want to change depending on setup (continued)

Variable name	Description	Referenced elsewhere	Default value or your company's information
csfusers_grp	A group of authorized ICSF service users. (This is non-applicable if `use_icsf=0`.)	No	
`key_backup`	Specifies whether the PKI Services CA certificate and private key should be backed up to an encrypted data set. The value can be: • 1 (yes — the default) • 0 (no). **Note:** When you use IKYSETUP, you need to enter a passphrase whose display is not inhibited — it appears on the screen in the clear.	No	**1** (yes)
`pgmcntl_dsn.`	A list in which pgmcntl_dsn.0 is the number of items in the list and the rest of the entries are a list of load libraries to be program controlled. If you set `unix_sec=2`, you probably need to update the list of data sets. (This is non-applicable if `unix_sec¬=2`.)	No	**9** (default for number of items) • '**CEE.SCEERUN**' • '**CBC.SCLBDLL**' • '**GLD.SGLDLNK**' • '**GSK.SGSKLOAD**' • '**SYS1.CSSLIB**' • '**TCPIP.SEZALINK**' • '**SYS1.LINKLIB**' • '**CSF.SCSFMOD0**' • '**CSF.SCSFMOD1**'
`restrict_surrog`	Specifies whether the surrogate user ID should be marked restricted. The value can be: • 0 (no — the default) • 1 (yes) **Recommendation:** Do not change the default the first time you run IKYSETUP, but change it before going into a production environment.	No	**0** (no)

Table 16. IKYSETUP variables you might want to change depending on setup (continued)

Variable name	Description	Referenced elsewhere	Default value or your company's information
`unix_sec`	Specifies whether to set up z/OS UNIX level security. (See page 28 for a definition of z/OS UNIX level security.) The value can be: • 0 (do not set up — the default) • 1 (is already set up) • 2 (add this level of security) If you are changing unix_sec to 1 or 2, you also need to review and possibly update the bpx_userid. and pgmcntl_dsn. rows. **Recommendation:** Do not set unix_sec=2 the first time you are running IKYSETUP.	For `unix_sec=2`, the names of the load libraries need to change.	**0** (no)
`use_icsf`	Specifies whether PKI Services should use ICSF for private key operations. The value can be: • 0 (no — the default) • 1 (yes). If you are changing use_icsf to 1, see also the csfkeys_profile, csfserv_profile, and csfusers_grp rows. **Recommendation:** Do not change the default the first time you run IKYSETUP, but change it before going into a production environment.	For this to be successful, ICSF must be configured for RSA (PKA) operations and running.	**0** (no)

Variables you can optionally change

Review the values of the following variables to determine if you want to change any of the defaults in the rightmost column. (You should probably change at least the values for `ca_expires` and `web_expires`.) If you decide to change any value, cross out the default in the rightmost column and record your company's information.

Running IKYSETUP

Table 17. IKYSETUP variables you can optionally change

Variable name	Description	Referenced elsewhere	Default value or your company's information
`backup_dsn`	The data set that will contain a backup copy of the PKI Services certificate and private key.	No	'*daemon*.**PRIVATE.KEY.BACKUP.P12BIN**' **Note:** The *daemon* refers to the `daemon` variable in this table.
`ca_expires`	The date the PKI Services CA certificate expires.	No	**2020/01/01** **Note:** You should update this value to the expiration date of your CA certificate.
`ca_ring`	The name of the PKI Services SAF key ring.	pkiserv.conf - [SAF] KeyRing value	**CAring**
`daemon`	The PKI Services daemon user ID.	pkiserv.conf - [SAF] KeyRing value	**PKISRVD**
`export_dsn`	The data set that will contain the PKI Services certificate for copying to HFS.	No	'*daemon*.**PRIVATE.CACERT.DERBIN'** **Note:** The *daemon* refers to the `daemon` variable in this table.
`log_dsn`	The log data set name.	No	'*your-id*.**PRIVATE.IKYSETUP.LOG'** **Notes:** 1. The *your-id* refers to the RACF ID of the person running IKYSETUP. (You do not need to add this; MVS adds this for you.) 2. Changing the default is not recommended.
`pkigroup`	The PKI Services administration group. This is a RACF group containing the list of user IDs that are authorized to use PKI Services administration functions.	No	**PKIGRP**
`surrog`	The surrogate user ID for PKI Services. **Note:** This cannot be an existing user ID (because IKYSETUP creates the user ID with the NOPASSWORD attribute).	httpd*.conf - Surrogate user ID	**PKISERV**
`vsamhlq`	The high-level qualifier of the VSAM data sets for PKI Services. **Note:** The RACF administrator gets this information from the MVS programmer	• pkiserv.conf - [ObjectStore] *DSN values • IKYCVSAM - Data sets names	Same as the `daemon` variable earlier in this table.
`web_expires`	The date the Web server certificate expires.	No	**2020/01/01** **Note:** You should update this value to the expiration date of your CA certificate.
`web_label`	The label for the Web server's certificate.	No	**SSL Cert**

Table 17. IKYSETUP variables you can optionally change (continued)

Variable name	Description	Referenced elsewhere	Default value or your company's information
`webserver`	The Web server's daemon user ID.	See Web server documentation.	**WEBSRV**

Steps for performing RACF tasks using IKYSETUP

You can use the following directions to run IKYSETUP with minimal changes or to extensively customize it.

Recommendation: If this is your first attempt to use IKYSETUP, you are recommended to change only the IKYSETUP variables in the section ″Things you must change.″ You can refine IKYSETUP later, after you are familiar with the process of updating and running it.

The following flowchart illustrates the iterative nature of the process of updating IKYSETUP:

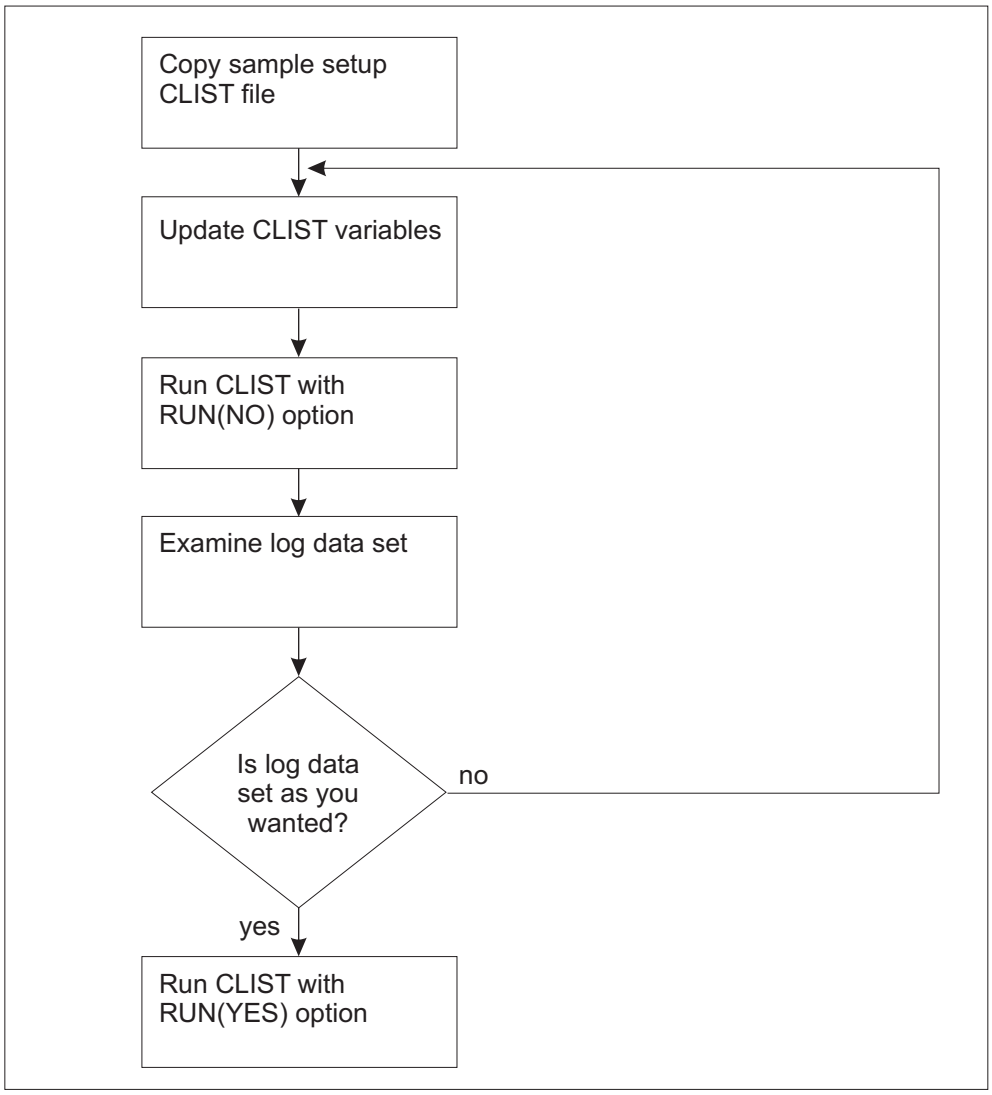


Figure 2. Flowchart of the process of updating IKYSETUP

Perform the following steps to use IKYSETUP to perform RACF administration tasks:

1. Copy 'SYS1.SAMPLIB(IKYSETUP)' to a data set you are permitted to edit.

2. Edit the IKYSETUP code to update the values of variables you changed in Table 11 on page 25.

   The following example shows how to change the `pkigroup_mem.` variables. (Remember that for `pkigroup_mem.`, you set `pkigroup_mem.0` to the number of items in the list and `pkigroup_mem.1` through `pkigroup_mem.n` to the PKI Services administration group member IDs.)

   Example:

```
pkigroup_mem.0=3       /* Number of pkigroup members to connect */
pkigroup_mem.1="TOM"
pkigroup_mem.2="DICK"
pkigroup_mem.3="HARRY"
```

3. If necessary, update the values of variables you changed in Table 16 on page 29 .

   The following example shows how to change the `use_icsf` variable.

   Example:

   ```
   use_icsf=1
   ```

---

4. Optionally update any variables you changed in Table 17 on page 32.

   The following example shows how to change the `log_dsn` variable.

   Example:

   ```
   log_dsn="PRIVATE.IKYSETUP.LOG"
   ```

---

5. Run IKYSETUP by entering the following command:

   ```
   EX 'data-set-name(IKYSETUP)' 'RUN(NO)'
   ```

   Notes:

   a. The user ID that runs IKYSETUP must be a RACF SPECIAL user ID.

   b. When IKYSETUP runs, it prompts you to enter your secret passphrase. (This is for encrypting the backup copy of your CA certificate and private key.) Be aware that asterisks do not replace the secret passphrase; it appears on the screen in the clear. **Make a note of this passphrase.** (If you forget it, your backup will be useless.)

   c. The NO option in the command specifies displaying the commands only. (This creates a log data set listing the commands and other information. Alternative parameters are: YES, which indicates running IKYSETUP as is, and PROMPT, indicates prompting the user before running each command.)

---

6. Review the log data set. (See Figure 3 on page 36 for an example of the data that appears on your display when you are running IKYSETUP; this is similar to the contents of the log data set.) The top part identifies the tasks and shows the commands that run to perform those tasks. Review this to ensure that the issued commands match your expectations. (For more information about these commands, see “Actions IKYSETUP performs by issuing RACF commands” on page 269.) The bottom part provides a record of important information that you will need for later steps, such the name of your daemon user ID. Review this information to ensure that the values are the ones you want.

   If you want to change any of the commands or information in the log data set, you need to change additional values in IKYSETUP. Remember to record any additional changes in Table 11 on page 25, Table 16 on page 29, and Table 17 on page 32. Then go back to step 3.

---

7. If the log data set includes the commands and information you want, rerun the IKYSETUP code by entering the following command:

   ```
   EX 'data-set-name(IKYSETUP)' 'RUN(YES)'
   ```

---

8. After running IKYSETUP with RUN(YES), examine the results recorded in the log data set. Investigate and rerun (potentially by hand) any failing commands. Investigate informational messages and make any necessary corrections. (Informational messages usually indicate a set-up problem that may affect

operations later. For example, any informational message from the RACDCERT commands that indicate that the certificate has been marked ″NO TRUST″ is an error.)

The following figure shows an example of the data that appears when you run IKYSETUP.

```
Creating users and groups ...
ADDUSER  PKISRVD name('PKI Srvs Daemon')  nopassword  omvs(uid(554)  assize(256000000)  threads(512))
ADDUSER  PKISERV nopassword   omvs(uid(555))  name('PKI Srvs Surrogate')
SETROPTS EGN GENERIC(DATASET)
ADDSD 'PKISRVD.**' UACC(NONE)
ADDGROUP  PKIGRP OMVS(GID(655))
Allowing administrators to access PKI databases ...
PERMIT 'PKISRVD.**' ID(PKIGRP) ACCESS(CONTROL)
SETROPTS GENERIC(DATASET) REFRESH
Creating the CA certificate ...
RACDCERT GENCERT CERTAUTH SUBJECTSDN(OU('Human Resources Certificate Authority')
 O('Your Company') C('Your Country 2 Letter Abbreviation'))
 WITHLABEL('Local PKI CA') NOTAFTER(DATE(2020/01/01))
Backing up the CA certificate ...
RACDCERT CERTAUTH EXPORT(LABEL('Local PKI CA'))  DSN('PKISRVD.PRIVATE.KEY.BACKUP.P12BIN')
FORMAT(PKCS12DER)  PASSWORD('********')
Marking CA certificate as HIGHTRUST ...
RACDCERT CERTAUTH ALTER(LABEL('Local PKI CA')) HIGHTRUST
Saving the CA certificate to a data set for OPUT ...
RACDCERT CERTAUTH EXPORT(LABEL('Local PKI CA'))  DSN('PKISRVD.PRIVATE.CACERT.DERBIN') FORMAT(CERTDER)
Creating the PKI Services keyring ...
RACDCERT ADDRING(CAring) ID(PKISRVD)
RACDCERT ID(PKISRVD) CONNECT(CERTAUTH  LABEL('Local PKI CA')  RING(CAring) USAGE
(PERSONAL) DEFAULT)
Creating the Webserver SSL certificate and keyring ...
RACDCERT GENCERT ID(WEBSRV) SIGNWITH(CERTAUTH  LABEL('Local PKI CA'))  WITHLABEL
('SSL Cert') SUBJECTSDN(CN('www.YourCompany.com') O('Your Company') L('Your City')
   SP('Your Full State or Province Name') C('Your Country 2 Letter Abbreviation'))
   NOTAFTER(DATE(2020/01/01))
RACDCERT ADDRING(SSLring) ID(WEBSRV)
RACDCERT  ID(WEBSRV) CONNECT(CERTAUTH  LABEL('Local PKI CA') RING(SSLring))
RACDCERT  ID(WEBSRV) CONNECT(ID(WEBSRV)  LABEL('SSL Cert') RING(SSLring)
USAGE(PERSONAL) DEFAULT)
Giving PKISRVD access to BPX.SERVER ...
RDEFINE FACILITY BPX.SERVER
PERMIT BPX.SERVER CLASS(FACILITY)  ID(PKISRVD) ACCESS(READ)
Allowing the PKI Services daemon to act as a CA ...
RDEFINE FACILITY IRR.DIGTCERT.GENCERT
RDEFINE FACILITY IRR.DIGTCERT.LISTRING
RDEFINE FACILITY IRR.DIGTCERT.LIST
PERMIT IRR.DIGTCERT.GENCERT CLASS(FACILITY)  ID(PKISRVD) ACCESS(CONTROL)
PERMIT IRR.DIGTCERT.LISTRING CLASS(FACILITY)  ID(PKISRVD) ACCESS(READ)
PERMIT IRR.DIGTCERT.LIST CLASS(FACILITY)  ID(PKISRVD) ACCESS(READ)
Allowing the Webserver to access its keyring ...
PERMIT IRR.DIGTCERT.LISTRING CLASS(FACILITY)  ID(WEBSRV) ACCESS(READ)
PERMIT IRR.DIGTCERT.LIST CLASS(FACILITY)  ID(WEBSRV) ACCESS(READ)
Allowing the Webserver to switch identity to PKISERV ...
SETROPTS CLASSACT(SURROGAT)
RDEFINE SURROGAT BPX.SRV.PKISERV
PERMIT BPX.SRV.PKISERV CLASS(SURROGAT)  ID(WEBSRV) ACCESS(READ)
SETROPTS RACLIST(SURROGAT) REFRESH
```

Figure 3. Sample log data set (Part 1 of 2)

```
Creating the STARTED class profile for the daemon ...
RDEFINE STARTED PKISRVD.* STDATA(USER(PKISRVD))
SETROPTS CLASSACT(STARTED) RACLIST(STARTED)
SETROPTS RACLIST(STARTED) REFRESH
Allowing PKISERV to request certificate functions ...
SETR GENERIC(FACILITY)
RDEFINE FACILITY IRR.RPKISERV.**
PERMIT IRR.RPKISERV.** CLASS(FACILITY) ID(PKISERV)  ACCESS(CONTROL)
Creating the profile to protect PKI Admin functions ...
RDEFINE FACILITY IRR.RPKISERV.PKIADMIN
PERMIT IRR.RPKISERV.PKIADMIN CLASS(FACILITY)  ID(PKIGRP) ACCESS(UPDATE)
PERMIT IRR.RPKISERV.PKIADMIN CLASS(FACILITY)  ID(PKISERV) ACCESS(NONE)
SETROPTS RACLIST(FACILITY) REFRESH

-----------------------------------------------
Information needed for PKI Services UNIX set up:
-----------------------------------------------

The daemon user ID is:
  PKISRVD

The VSAM high level qualifier is:
  PKISRVD
This is needed for the [ObjectStore] section in pkiserv.conf

The PKI Services' DER encoded certificate is in data set:
  'PKISRVD.PRIVATE.CACERT.DERBIN'
This must be OPUT to /var/pkiserv/cacert.der with the BINARY option

The fully qualified PKI Services' SAF keyring is:
  PKISRVD/CAring
This is needed for the [SAF] section in pkiserv.conf

The PKI Services CA DN is:
  OU=Human Resources Certificate Authority,O=Your Company,C=Your Country 2 Letter Abbreviation
The suffix must match the LDAP suffix in slapd.conf

The webserver's SAF keyring is:
  SSLring
This is needed for the KeyFile directive in httpd*.conf files

The Webserver's DN is:
  CN=www.YourCompany.com,O=Your Company,L=Your City,ST=Your Full State or
  Province Name,C=Your Country 2 Letter Abbreviation
The left most RDN must be the webserver's fully qualified domain name
```

Figure 3. Sample log data set (Part 2 of 2)

Chapter 5. Configuring the UNIX runtime environment

After the RACF administrator performs the tasks necessary to set up PKI Services, the UNIX programmer needs to copy and update certain files and set up the PKI Services variables directories. The following table summarizes information about copying and updating files:

Table 18. Deciding which files to copy and change

File	Purpose	Need to copy?	Need to change?
pkiserv.conf	Configuration file. Contains various settings and values PKI Services needs.	Yes	The UNIX programmer needs to update the LDAP section of this file, but IBM recommends doing this later (see Chapter 8, "Tailoring the PKI Services configuration file for LDAP" on page 55). **Recommendation:** Do not change the other (non-LDAP) sections. However, if you choose to do so, there are certain variables (identified in Table 19 on page 43) you should not change except when you are updating the pkiserv.tmpl certificate template file.
pkiserv.tmpl	Certificate templates file. Contains HTML-style code that builds the Web pages underlying certificate requests.	Yes	**Recommendation:** Make no changes to this file until later. See Chapter 11, "Customizing the administration Web pages" on page 101 for details about making changes.
pkiserv.envars	The environment variables file.	Only if the file needs changes	UNIX programmer may have to update this file. See "Optionally updating PKI Services environment variables" on page 40.

(To view the contents of any of these files, see Chapter 25, "Other code samples" on page 287.)

The UNIX programmer performs the following tasks:

- Copies the configuration and certificate template files (and the environment variables file, if it needs any changes) from their source HFS directory (by default, /usr/lpp/pkiserv/samples) to their runtime directory (by default, /etc/pkiserv)
- If necessary, updates the environment variables file
- If necessary, updates the configuration file
- Sets up the /var/pkiserv directory.

Steps for copying files

Before you begin:

- You need to obtain the following publication:

  z/OS UNIX System Services Planning
- You need to know the HFS directory where the MVS programmer installed PKI Services and the runtime directory, *HFS-install-dir* and *runtime-dir* in the commands that follow. The defaults are /usr/lpp/pkiserv/ and /etc/pkiserv respectively. The MVS programmer was asked to record any changes to these defaults; see Table 3 on page 9.

Configuring the UNIX runtime environment

- The user ID you use for copying files must have superuser authority.

Perform the following steps to copy the files:

1. Copy the configuration and template files by entering the following commands from the UNIX command line.

   Note: To use these commands, your user ID must have super user authority.

   ```
   cp -p /HFS-install-dir/samples/pkiserv.conf runtime-dir
   cp -p /HFS-install-dir/samples/pkiserv.tmpl runtime-dir
   ```

   ---

2. Examine the values in the environment variables file (by default, pkiserv.envars). If any values need to change (such as the OCSFREGDIR, the environment variable for the OCSF registry directory — see step 2 on page 17), copy this file by entering the following command:

   ```
   cp -p /HFS-install-dir/samples/pkiserv.envars runtime-dir
   ```

Optionally updating PKI Services environment variables

You need to define certain environment variables (such as LIBPATH) for the PKI Services daemon to run. There are two files related to environment variables.

- A sample environment variables file, pkiserv.envars (by default in /usr/lpp/pkiserv/samples/)
- SYS1.PROCLIB member PKISERVD (You can use the ENVAR parameter to point to the environment variables file.)

You can use pkiserv.envars to set environment variables for the PKI Services daemon. This file contains most of the environment variables needed to run the daemon. You may be able to use the file without changing it if you meet the following conditions:

- You used the default value for the install directory
- You want to use the default message level
- You are using the default location for the OCSF Registry directory (/var/ocsf).

Recommendation: If you need to make changes to the pkiserv.envars file, copy the file another directory (such as /etc/pkiserv) and make changes only to the copy.

PKISERVD is the sample procedure to start PKI Services. (For sample code, see “PKISERVD sample procedure to start PKI Services daemon” on page 292.) PKISERVD sets the TZ (time zone) environment variable because it is very likely that the value of this variable needs to change. PKISERVD also includes parameters specifying the directory containing the environment variables file (DIR) and the file name of the environment variables file (FN). If you make a copy of pkiserv.envars as recommended, you also need to change the name of the directory in PKISERVD (for example, `DIR="/etc/pkiserv"`) and possibly the file name (for example, `FN="pki.env"`).

Note: You can change all of the following on the start command:

- environment variables directory
- file name
- job output class
- region size

- stdout
- stderr
- time zone

See “Steps for starting the PKI Services daemon” on page 60.

Because of the limitation of the number of characters allowed in the PARM=*operand* on the JCL EXEC card, take care to ensure that the total length of the environment variables directory and file name, TZ value, and stdout and stderr redirection values do not exceed the 100 character maximum.

You must specify any environment variables that PKI Services requires either in the PKISERVD procedure or in the environment variables file (pkiserv.envars). IBM recommends making additions and changes to the environment variables file.

(Optional) Steps for updating PKI Services environment variables

Perform the following steps to update PKI Services environment variables:

1. Examine the values in the environment variables file (by default, pkiserv.envars) and update the file as necessary. (See “Environment variables in the environment variables file” on page 265 for a description of the environment variables and “The pkiserv.envars environment variables file” on page 267 for a code sample of the environment variables file.)

   Note: If the value set for the OCSF registry directory differs from the default value of ’/var/ocsf’, you need to update the OCSFREGDIR environment variable.

2. Make any needed changes to PKISERVD, such as updating the pathname of the environment variables file (FN and DIR parameters). (See “PKISERVD sample procedure to start PKI Services daemon” on page 292 for a code sample of the PKISERVD proc.)

(Optional) Steps for updating the configuration file

The pkiserv.conf configuration file for the PKI Services daemon consists of sections of name-value pairs. **Everything in the pkiserv.conf file — including section names, keys, and values — is case-sensitive.** Each section of the pkiserv.conf configuration file has a title enclosed in square brackets. The configuration file includes the following sections:

[OIDs]

The OIDs section specifies the object identifiers for various nicknames PKI Services uses internally. The OIDs are specified in the following form:

```
<name>=<dotted-decimal>
```

Example:

```
[OIDs]
:
MyPolicy=1.2.3.4
```

[ObjectStore]

The ObjectStore section specifies operational

information for various files and data sets. The following is an example of the ObjectStore section:

Example:

```
[ObjectStore]
Name=pkica
⋮
```

[CertPolicy]

The CertPolicy section is for CA policy information. The following is an example of the CertPolicy section:

Example:

```
[CertPolicy]
SigAlg1=sha-1WithRSAEncryption
⋮
```

[General]

The General section is for general information. The following is an example of the General section:

Example:

```
[General]

InitialThreadCount=10
⋮
```

[SAF]

The SAF section is for information about the SAF (RACF) key ring that is used for CA certificate and private key storage. The following is an example of the SAF section:

Example:

```
[SAF]
KeyRing=PKISRVD/CAring
```

[LDAP]

The LDAP section contains information about the LDAP server for posting certificates and CRLs. The following is an example of the LDAP section:

Example:

```
[LDAP]
NumServers=1
⋮
```

The UNIX programmer needs to update the LDAP section of this file, but IBM recommends doing this later (see Chapter 8, “Tailoring the PKI Services configuration file for LDAP” on page 55). For sections other than LDAP, the configuration file contains the values that IBM recommends. Therefore, IBM recommends that you do not change these sections. However, you can change them if you wish.

Before you begin: The following table provides information about parameters in the pkiserv.conf configuration file. (It omits parameters for the LDAP section. For information about these parameters, see Table 23 on page 55.) Read the parameter descriptions, and examine the default values in the rightmost column to ensure that the values meet your company’s requirements. As necessary, cross out the defaults and enter the information appropriate to your own company’s needs and policies.

Table 19. Information needed for updating the configuration file

Parameter	Information needed	Where to get this information	Default value or customized value
OIDs section			
`MyPolicy=`	A registered Object ID identifying your organization's usage policy, for example: `1.2.3.4`	Do not change this information until you are performing advanced customization. See "Steps for creating the CertificatePolicies extension" on page 107 for more information.	**1.2.3.4** If you need to use the CertificatePolicies extension, replace `1.2.3.4` with the value of your Object ID: _____
ObjectStore section			
`Name=`	File-name prefix PKI Services uses when creating the full file names.	UNIX programmer decides this value.	**pkica**
`Path=`	HFS path to the working directory.	See variables-dir in Table 3 on page 9.	**/var/pkiserv**
`ObjectDSN=`	VSAM data set name for ObjectStore data. This is the request database. Each VSAM request record consists of a fixed header followed by a variable-length section containing the BER-encoded request.	For the high-level qualifier before the period, see the *vsamhlq* variable in Table 17 on page 32. The name of the file (after the period) can change; the MVS programmer who creates the VSAM data sets usually decides these names.	'**pkisrvd**.**vsam.ost**' Note that this begins with the VSAM high-level qualifier.
`ObjectTidDSN=`	VSAM data set name for the ObjectStore alternate index.	For the high-level qualifier before the period, see the *vsamhlq* variable in Table 17 on page 32. The name of the file (after the period) can change; the MVS programmer who creates the VSAM data sets usually decides these names.	'**pkisrvd**.**vsam.ost.path**' Note that this begins with the VSAM high-level qualifier.
`ICLDSN=`	VSAM data set name for ICL data. This contains the certificates that have been issued. Each VSAM ICL record consists of a fixed header followed by a variable-length section containing the BER-encoded request.	For the high-level qualifier before the period, see the *vsamhlq* variable in Table 17 on page 32. The name of the file (after the period) can change; the MVS programmer who creates the VSAM data sets usually decides these names.	'**pkisrvd**.**vsam.icl**' Note that this begins with the VSAM high-level qualifier.
`RemoveCompletedReqs=`	Time period that completed certificate requests remain in the ObjectStore before automatic deletion. This is a number followed by `d` (days) or `w` (weeks).	UNIX programmer decides this value.	**1w**

Table 19. Information needed for updating the configuration file (continued)

Parameter	Information needed	Where to get this information	Default value or customized value
`RemoveInactiveReqs=`	Time period that incomplete, inactive certificate requests remain in the ObjectStore before automatic deletion. This is a number followed by `d` (days) or `w` (weeks).	UNIX programmer decides this value.	**4w**
CertPolicy section			
`SigAlg1=`	The Object ID for the signature algorithm. This must be an RSA signature algorithm: • **`sha-1WithRSA Encryption`** `(the default)` • `md-5WithRSAEncryption` • `md-2WithRSAEncryption` **Note:** Changing the default also requires adding a line in the OIDs section. See "Updating the signature algorithm" on page 109.	Do not change this information until you are performing advanced customization. See "Updating the signature algorithm" on page 109 for more information.	**sha-1WithRSA Encryption**
`CreateInterval=`	How often the certificate creation thread scans the database for approved requests. This is a number followed by `w` (weeks), `d` (days), `h` (hours), `m` (minutes), or `s` (seconds).	UNIX programmer decides this value.	**3m**
`TimeBetweenCRLs=`	The time between certificate revocation lists. This is a number followed by `w` (weeks), `d` (days), `h` (hours), `m` (minutes), or `s` (seconds).	UNIX programmer decides this value.	**1d**
`CRLDuration=`	The amount of time that a certificate revocation list is valid. This is a number followed by `w` (weeks), `d` (days), `h` (hours), `m` (minutes), or `s` (seconds).	UNIX programmer decides this value.	**2d**
`PolicyRequired=`	Whether the CertificatePolicies extension is included in the certificate. This is `T` (True) or `F` (False). **Unless you change this to T, the following fields in the CertPolicy section are ignored.**	UNIX programmer decides this value. It should be `T` if you are using the CertificatePolicies extension or `F` otherwise. Do not change this information until you are performing advanced customization. See "Steps for creating the CertificatePolicies extension" on page 107 for more information.	**F**

Table 19. Information needed for updating the configuration file (continued)

Parameter	Information needed	Where to get this information	Default value or customized value
`PolicyCritical=`	Whether the CertificatePolicies extension is created with the critical flag turned on. This is `T` (True) or `F` (False).	UNIX programmer decides this value. It should be `T` if you are using the CertificatePolicies extension or `F` otherwise. Do not change this information until you are performing advanced customization. See “Steps for creating the CertificatePolicies extension” on page 107 for more information.	**F**
`PolicyName1=`	The Object ID for the policy. (This is the same value that is in the MyPolicy parameter of the OIDs section.)	Do not change this information until you are performing advanced customization. See “Steps for creating the CertificatePolicies extension” on page 107 for more information.	*MyPolicy* If you changed PolicyRequired=F to PolicyRequired=T, replace the variable *MyPolicy* with the same value that is in the MyPolicy parameter of the OIDs section. _____
`Policy1Org=`	This is the organization name for the CertificatePolicies extension, for example, `International Business Machines, Inc.`	Do not change this information until you are performing advanced customization. See “Steps for creating the CertificatePolicies extension” on page 107 for more information.	**My Company, Inc.** If you are changing PolicyRequired=F to PolicyRequired=T, you need to specify your own value for this: _____
`Policy1Notice1=`	The first company notice number.	Do not change this information until you are performing advanced customization. See “Steps for creating the CertificatePolicies extension” on page 107 for more information.	**1** If you are changing PolicyRequired=F to PolicyRequired=T, you need to specify your own value for this: _____
`Policy1Notice2=`	The second company notice number.	Do not change this information until you are performing advanced customization. See “Steps for creating the CertificatePolicies extension” on page 107 for more information.	**17** If you are changing PolicyRequired=F to PolicyRequired=T, you need to specify your own value for this: _____

Table 19. Information needed for updating the configuration file (continued)

Parameter	Information needed	Where to get this information	Default value or customized value
`UserNoticeText1=`	A legal statement about certificate issuance and use, for example: `Certificate for IBM internal use only`	Do not change this information until you are performing advanced customization. See "Steps for creating the CertificatePolicies extension" on page 107 for more information.	*statement* If you are changing PolicyRequired=F to PolicyRequired=T, you need to replace the variable *statement* with your own value for this: _____
`CPS1=`	The Uniform Resource Identifier (URI) where your organization's Certification Practice Statement (CPS) is located. This is in the form: `http://www.`*`mycompany`*`.com/cps.html`	Do not change this information until you are performing advanced customization. See "Steps for creating the CertificatePolicies extension" on page 107 for more information.	**http://www.***mycompany***.com/cps.html** If you are changing PolicyRequired=F to PolicyRequired=T, you need to replace the variable *mycompany* with your own value for this: **http://www._____.com/cps.html**
General section			
`InitialThreadCount=`	Number of threads (at least 2 and no more than 100) the PKI Services daemon should create at program initialization.	UNIX programmer decides this value.	**10**
SAF section			
`KeyRing=`	The fully qualified name of the SAF key ring for PKI Services to use. (This must consist of an uppercase user ID ″/″ case-sensitive ring name.)	See the earlier table: Table 17 on page 32	**PKISRVD/CAring**
LDAP section — For information about the LDAP section, see Table 23 on page 55.			

Perform the following steps to update the pkiserv.conf configuration file:

Note: Keep in mind that everything in the pkiserv.conf file — including section names, keys, and values — is case-sensitive.

1. Optionally, update the values of the parameters in the ObjectStore section.
   a. If necessary, change **pkica** in the following line to the value in the Name= row in the preceding table:

      `Name=`**`pkica`**

   b. If necessary, change **/var/pkiserv** in the following line to the value in the Path= row in the preceding table:

      `Path=`**`/var/pkiserv`**

   c. If necessary, change **pkisrvd** in the following lines to the value of the VSAM high-level qualifier in the ObjectDSN=, ObjectTidDSN=, and ICLDSN= rows of the preceding table; if you changed the file names after the period, replace these values also:

```
ObjectDSN='pkisrvd.vsam.ost'
ObjectTidDSN='pkisrvd.vsam.ost.path'
ICLDSN='pkisrvd.vsam.icl'
```

Note: The high-level qualifier of the VSAM data set names must match the name of the RACF user ID assigned to the PKI Services daemon (by default, PKISRVD). If you change from the default to another user ID, you need to change the high-level qualifier in the configuration file as well. If the MVS programmer changes the data set names (see Step 2c on page 60), you must make equivalent changes in the pkiserv.conf file.

d. If necessary, change **1w** in the following line to the value in the RemoveCompletedReqs= row of the preceding table:

```
RemoveCompletedReqs=1w
```

e. If necessary, change **4w** in the following line to the value in the RemoveInactiveReqs= row of the preceding table:

```
RemoveInactiveReqs=4w
```

---

2. If necessary, update the values of the parameters in the CertPolicy section.

   a. If necessary, change **3m** in the following line to the value in the CreateInterval= row of the preceding table:

   ```
   CreateInterval=3m
   ```

   b. If necessary, change **1d** in the following line to the value in the TimeBetweenCRLs= row of the preceding table:

   ```
   TimeBetweenCRLs=1d
   ```

   c. If necessary, change **2d** in the following line to the value in the CRLDuration= row of the preceding table:

   ```
   CRLDuration=2d
   ```

   d. If necessary, change **F** in the following line to the value in the PolicyRequired= row of the preceding table:

   ```
   PolicyRequired=F
   ```

---

3. Optionally, update the values of the parameters in the General section. If necessary, change **10** in the following line to the value in the InitialThreadCount= row of the preceding table:

   ```
   InitialThreadCount=10
   ```

---

4. If necessary, change **PKISRVD/CAring** in the following line to the value in the KeyRing= row of the preceding table:

   ```
   KeyRing=PKISRVD/CAring
   ```

Steps for setting up the /var/pkiserv directory

PKI Services needs to set up HFS files in a directory. (The default location is /var/pkiserv.) You need to set up this location and make the PKI Services daemon (by default, PKISRVD) the owner.

Configuring the UNIX runtime environment

Then you copy the CA certificate from its MVS data set to the cacert.der in the directory (the default location is /var/pkiserv) and change its permission settings. (The data set was created earlier. See “Before you begin” on page 24; the default name of the data set is 'pkisrvd.private.cacert.derbin'.)

Perform the following steps to set up the /var/pkiserv directory:

1. Change the ownership of the directory to PKISRVD by entering the following command from the UNIX command line:

   ```
   chown PKISRVD /var/pkiserv
   ```

2. Copy the CA certificate from its MVS data set to cacert.der in the /var/pkiserv directory by entering the following command from the UNIX command line:

   ```
   cp "//'pkisrvd.private.cacert.derbin'" /var/pkiserv/cacert.der
   ```

3. Change the permission settings of the file by entering the following command from the UNIX command line:

   ```
   chmod 755 /var/pkiserv/cacert.der
   ```

4. Change the ownership of the file by entering the following command from the UNIX command line:

   ```
   chown pkisrvd /var/pkiserv/*
   ```

Chapter 6. Tailoring LDAP configuration for PKI Services

The directions in this section are for tailoring the configuration of the z/OS Security Server LDAP for PKI Services. If you are using a different LDAP product, you need to refer to the documentation for this product. See Appendix A, “LDAP directory server requirements” on page 323 for information about installing a non-z/OS LDAP.

The LDAP programmer needs to tailor LDAP configuration for PKI Services by loading the user schema file so that the LDAP server understands the format of entries that will be stored in the directory.

Steps for tailoring LDAP configuration for PKI Services

Before you begin:

- You will need LDAP programming skills to complete this procedure.
- Make sure that the LDAP server is started before beginning these steps. If you are unsure about this, see “Steps for installing and configuring LDAP” on page 18.
- You need to know the following information from LDAP installation. Copy the information into the following table from (completed) Table 9 on page 19:

Table 20. LDAP information you need for tailoring LDAP configuration

LDAP information	Explanation	Value
Administrator’s distinguished name	This is the distinguished name to use for LDAP binding. (For a definition of distinguished name, see Table 9 on page 19. The LDAP administrator defines the administrator’s distinguished name with the **adminDN** keyword in the /etc/ldap/slapd.conf configuration file. For example, the value is ″cn=Admin″ in the following: `adminDN="cn=Admin"`	
Administrator password	This is the password to use for LDAP binding. The LDAP programmer can set this in several ways, for example: – By specifying the password as a TDBM entry by using the **userPassword** attribute in the **ldif2tdbm** load utility – (Not recommended) by using the **adminPW** keyword in the slapd.conf configuration file.	
LDAP fully qualified domain name and port	This is the IP address and port on which the LDAP server is listening. For example, for `ldap.widgets.com:389`, the fully qualified domain name is `ldap.widgets.com` and the port is `389`. See Table 7 on page 16 for a definition of fully qualified domain name.	
Suffix	(For a definition of suffix, see Table 9 on page 19.) The suffix value is specified after the suffix keyword in the slapd.conf file. `suffix "o=`*`your-company`*`,c=`*`your-country-abbreviation`*`"`	

Perform the following steps to tailor LDAP configuration for PKI Services:

1. Copy the /usr/lpp/ldap/etc/schema.user.ldif file to the directory from which you are working by entering the following z/OS UNIX shell command:

   ```
   cp /usr/lpp/ldap/etc/schema.user.ldif  .
   ```

2. Edit the schema.user.ldif file in the current directory, ensuring that the "dn:" line (the first line in the file) has the following form and replacing *Your Company Suffix* with the suffix from the preceding table:

   ```
   dn: cn=schema, Your Company Suffix
   ```

   ---

3. Load the schema defined in the schema.user.ldif file into the directory by entering the following command, replacing *admindn* and *passwd* with the adminDN and adminPW values from the preceding table:

   ```
   ldapmodify -D admindn -w passwd -V 3 -f schema.user.ldif
   ```

Chapter 7. Updating z/OS HTTP Server configuration and starting the server

Starting the Web server requires having a configuration file for it. This chapter describes how the Web server programmer performs the following tasks:

- Updating the z/OS HTTP Server's configuration files by cutting and pasting directives from the PKI Services samples directory into them
- Starting the z/OS HTTP Server.

Before you begin:

- The z/OS HTTP Server must have already been configured.
- It would be helpful to have available a copy of *z/OS HTTP Server Planning, Installing, and Using*.

Steps for updating the z/OS HTTP Server's configuration files

PKI Services uses two modes of SSL, and these two modes require running two instances of the z/OS HTTP Server. Although the two instances share a single server certificate and private key, they use two different configuration files.

- The first configuration file is your existing configuration file (created earlier — see "Steps for installing and configuring the z/OS HTTP Server to work with PKI Services" on page 15). It specifies port 80 for normal HTTP traffic and port 443 for the SSL traffic port.
- The second configuration file, /etc/httpd1443.conf, specifies SSL traffic only on port 1443, with client authentication. (If this file does not exist, you create it by copying the first file.)

The following table summarizes the configuration and usage of each Web server:

Table 21. Summary of configuration and usage of each Web server instance

Server instance	Protocol	SSL	Server authentication	Client authentication	Port number
First instance	HTTP	No	No	No	80
First instance	HTTPS	Yes	Yes	No	443
Second instance	HTTPS	Yes	Yes	Yes	1443

Before you begin:

- You need to know the HFS install directory (the HFS directory where the MVS programmer installed PKI Services), called *HFS-install-dir* in the commands that follow. The default is /usr/lpp/pkiserv/. The MVS programmer was asked to record any changes to the defaults; see Table 3 on page 9.
- You need to know the following LDAP information. Record the information in the rightmost row of the following table:

Updating z/OS HTTP Server configuration and starting the server

Table 22. LDAP information you need for tailoring z/OS HTTP Server configuration

LDAP information	Explanation	Value
Administrator's distinguished name	This is the distinguished name to use for LDAP binding. (For a definition of distinguished name, see Table 9 on page 19.) The LDAP administrator defines the administrator's distinguished name with the **adminDN** keyword in the /etc/ldap/slapd.conf configuration file. For example, the value is ″cn=Admin″ in the following: `adminDN="cn=Admin"`	
Administrator password	This is the password to use for LDAP binding. The LDAP programmer can set this in several ways, for example: – By specifying the password as a TDBM entry by using the **userPassword** attribute in the **ldif2tdbm** load utility – (Not recommended) by using the **adminPW** keyword in the slapd.conf configuration file.	
LDAP fully qualified domain name	This is the IP address on which the LDAP server is listening, for example, for `ldap.widgets.com`. See Table 7 on page 16 for a definition of fully qualified domain name.	
LDAP port	This is the port for LDAP, for example, `389` in `ldap.widgets.com:389`	

Perform the following steps to update the z/OS HTTP Server's configuration files:

1. If the second configuration file does not yet exist, create it by copying the first configuration file with the following command:

   ```
   cp -p /etc/httpd.conf /etc/httpd1443.conf
   ```

   ---

2. Copy the first set of sample z/OS HTTP Server configuration directives (from the PKI Services samples directory, /*HFS-install-dir*/samples/httpd.conf file) into the default configuration file, /etc/httpd.conf.

   Note: The *HFS-install-dir*, your HFS installation directory, by default is /usr/lpp/pkiserv. The MVS programmer determines whether to change this default (see Table 3 on page 9).

   a. Copy the keyfile, sslmode, sslport, and normalmode directives as is, replacing any existing values.

   b. If your organization customized the value of web_ring (see Table 11 on page 25), change `SSLring` in the keyfile directive in the following line to the customized value:

      ```
      keyfile SSLring SAF
      ```

   c. Optionally, copy the userID directive as is, replacing any existing value.

      Recommendation:

      You are recommended to copy the userID directive (as shown in the following) into your file as is. However, if you already have a value in your file for this, you are not required to change it.

      ```
      UserId %%CLIENT%%
      ```

   d. Copy the protection and protect directives after any protection and protect directives you already have. Do not change the order in which these directives appear.

   e. Copy the redirect directives after any redirect directives you already have. Do not change the order in which these directives appear.

   f. Copy the pass and exec directives before any pass and exec directives you already have.

g. Add the addtype directives to your list of addtypes if they don't already exist.

h. Change all instances of *server-domain-name* to your Web server's fully qualified domain name, for example, www.ibm.com. (For information about your Web server's fully qualified domain name, see Table 7 on page 16.)

i. Change all instances of *application-root* to your HFS installation directory, which is usr/lpp/pkiserv by default.

Note: Your HFS installation directory by default is /usr/lpp/pkiserv. The MVS programmer determines whether to change this default (see Table 3 on page 9).

---

3. Copy the second set of z/OS HTTP Server configuration directives (from the PKI Services samples directory, /*HFS-install-dir*/samples/httpd2.conf) into the /etc/httpd1443.conf file.

Note: The *HFS-install-dir*, your HFS installation directory, by default is /usr/lpp/pkiserv. The MVS programmer determines whether to change this default (see Table 3 on page 9).

a. If you created this file by copying the first httpd.conf file, delete all existing protection, protect, redirect, pass, and exec directives.

b. Copy the userId, keyfile, sslmode, sslport, sslclientauth, normalmode, and SSLX500CARoots directives as is, replacing any existing values.

c. If your organization customized the value of web_ring (see Table 11 on page 25), change `SSLring` in the keyfile directive in the following line to the customized value:

```
keyfile SSLring SAF
```

d. Add the following directives after the SSLX500CARoots directive:

- SSLX500Host
- SSLX500Port
- SSLX500UserID
- SSLX500Password

Replace the <> placeholders with the actual values from Table 22 on page 52.

e. Copy the protection and protect directives after any protection and protect directives you already have. Do not change the order in which these directives appear.

f. Copy the redirect directives after any redirect directives you already have. Do not change the order in which these directives appear.

g. Copy the exec directives before any pass and exec directives you already have.

h. Change all instances of *server-domain-name* to your Web server's fully qualified domain name, for example, www.ibm.com. (For information about your Web server's fully qualified domain name, see Table 7 on page 16.)

i. Change all instances of *application-root* to your HFS installation directory.

Note: Your HFS installation directory by default is /usr/lpp/pkiserv. The MVS programmer determines whether to change this default (see Table 3 on page 9).

j. If you created httpd1443.conf by copying httpd.conf, optionally change the directories in httpd1443.conf for the report, log, and pid files. (IBM

recommends that you do this to ensure the two servers are not using the same files at the same time.) To do this:

1) Create a new directory for the httpd1443 files by using the following command:

```
mkdir /etc/internet/logs1443
```

2) Assign ownership to WEBSRV with the following command:

```
chown websrv /etc/internet/logs1443
```

3) Edit the *Log directives in the new httpd1443.conf file to provide unique path names.

   For example, if the first httpd.conf file has the following:

```
AccessLog     /etc/internet/logs/httpd-log
AgentLog      /etc/internet/logs/agent-log
RefererLog    /etc/internet/logs/referer-log
ErrorLog      /etc/internet/logs/httpd-errors
CgiErrorLog   /etc/internet/logs/cgi-errors
```

   change the httpd1443.conf *Logs to the following:

```
AccessLog     /etc/internet/logs1443/httpd-log
AgentLog      /etc/internet/logs1443/agent-log
RefererLog    /etc/internet/logs1443/referer-log
ErrorLog      /etc/internet/logs1443/httpd-errors
CgiErrorLog   /etc/internet/logs1443/cgi-errors
```

Steps for starting the z/OS HTTP Server

Perform the following steps to start the z/OS HTTP Server:

1. Make sure that the LDAP server is started. (If you are unsure about this, see “Steps for installing and configuring LDAP” on page 18.)

---

2. Enter the following commands from the UNIX command line:

```
httpd
httpd -r /etc/httpd1443.conf
```

   Alternately, if you are using the IMWEBSRV started procedure as shipped with the Web server, you can start the two instances by entering the following MVS console commands:

```
S IMWEBSRV
S IMWEBSRV,ICSPARM='-r /etc/httpd1443.conf'
```

Chapter 8. Tailoring the PKI Services configuration file for LDAP

Chapter 5, “Configuring the UNIX runtime environment” on page 39 describes tasks the UNIX programmer performs. Then other team members perform additional tasks before the UNIX programmer updates the LDAP section of the pkiserv.conf configuration file.

Steps for tailoring the LDAP section of the configuration file

Before you begin:

- You will need UNIX programming skills to complete this procedure.
- Table 23 lists some parameters that are in the LDAP section of the pkiserv.conf configuration file. The rightmost column lists the default values. You need to change some of these values. Fill in the blank lines with your company’s information (and cross out these defaults). If you decide to change any of the other defaults, cross out these values and record your company’s information.

Table 23. Information needed for updating the LDAP section of the configuration file

Parameter	Information needed	Where to get this information	Default value and your company’s information
NumServers=	The number of available LDAP servers. These are replicas that can post certificates and CRLs.	From LDAP programmer	**1**
PostInterval=	How often the posting thread should scan the database for items to post in weeks (w), days (d), hours (h), minutes (m), or seconds (s).	UNIX programmer decides this. Specify a number followed by h (hours), m (minutes) or s (seconds). Example: `6m`	**5m**
Server1=	The fully qualified domain name (IP address and port) for the first LDAP server. For example, for `ldap.widgets.com:389`, the fully qualified domain name is `ldap.widgets.com` and the port is `389`. (See Table 7 on page 16 for a definition of fully qualified domain name.)	Copy this information from the earlier (completed) table: Table 20 on page 49	*myldapserver.mycompany*.com:389 (See note at end of table.)

Table 23. Information needed for updating the LDAP section of the configuration file (continued)

Parameter	Information needed	Where to get this information	Default value and your company's information
AuthName1=	The distinguished name to use for LDAP binding. (See Table 9 on page 19 for a definition of distinguished name.) The LDAP administrator defines the administrator's distinguished name with the **adminDN** keyword in the /etc/ldap/slapd.conf configuration file. For example, the value is ″cn=Admin″ in the following: `adminDN="cn=Admin"`	Copy this information from the earlier (completed) table: Table 20 on page 49	*CN=root* (See note at end of table.)
AuthPwd1=	The password to use for LDAP binding. The LDAP programmer sets this.	Copy this information from the earlier (completed) table: Table 20 on page 49	*root* (See note at end of table.) _____
CreateOUValue=	Value to use for the OU attribute when creating LDAP entries under the objectclass organizationalUnit (see Table 66 on page 323). This is used only when no OU value is specified in the relative distinguished name.	UNIX programmer decides this (after consulting with LDAP programmer)	**Created by PKI Services**
RetryMissingSuffix=	True (T) or False (F) setting that indicates whether LDAP post requests should be retried later if the distinguished name suffix does not exist. When set to F, LDAP post requests that fail because of a missing suffix are discarded.	UNIX programmer decides this (after consulting with LDAP programmer)	**T**
Note: If the number of servers (row 1) is greater than one, you need this value for **each** server.			

Perform the following steps to update the LDAP section of the pkiserv.conf configuration file:

1. If necessary, change `1` (the default) in the following line to the number of available LDAP servers listed in the preceding table:

   ```
   NumServers=1
   ```

   _____

2. Optionally change `5m` in the following line to the posting interval in the preceding table:

```
PostInterval=5m
```

---

3. Change *your-ldap-server-address:port* to your fully qualified domain name and port as listed in the preceding table:

   ```
   Server1=your-ldap-server-address:port
   ```

   Note: If the value in the NumServers line is greater than 1, add another line like the preceding for each additional server. (Increment the number in each additional line, for example, Server2.)

---

4. Change *CN=root* in the following line to the value of the administrator distinguished name in the preceding table:

   ```
   AuthName1=CN=root
   ```

   Note: If the value in the NumServers line is greater than 1, add another line like the preceding for each additional server. (Increment the number in each additional line, for example, AuthName2.)

---

5. Change *root* in the following line to the value of the administrator password in the preceding table:

   ```
   AuthPwd1=root
   ```

   Note: If the value in the NumServers line is greater than 1, add another line like the preceding for each additional server. (Increment the number in each additional line, for example, AuthPwd2.)

---

6. If necessary, change 'Created by PKI Services' in the following line to the OU attribute value in the preceding table:

   ```
   CreateOUValue=Created by PKI Services
   ```

---

7. If necessary, change 'T' in the following line to the value in the last row of the preceding table:

   ```
   RetryMissingSuffix=T
   ```

Tailoring the PKI Services configuration file for LDAP

Chapter 9. Creating VSAM data sets and starting and stopping PKI Services

The MVS programmer performs the following tasks:

- Creating the VSAM object store and ICL data sets and indexes
- Starting the PKI Services daemon
- If necessary, stopping the PKI Services daemon

Space considerations for creating VSAM data sets

The MVS programmer uses the IKYCVSAM sample JCL to create two VSAM data sets (clusters):

- A data set for the request database (object store)
- A data set for the Issued Certificate List (ICL).

The IKYCVSAM sample JCL contains default values for the primary and secondary extent allocations. Both are 50 (RECORDS(50 50)). You need to update these values based on your anticipated future needs. Use the following guidelines to update the RECORDS parameter for the DEFINE CLUSTER statements. Keep in mind that IDCAMS allocates extents on a track basis. (For more information about IDCAMS, see *z/OS DFSMS Access Method Services for Catalogs*.) After determining the size of the extent desired, IDCAMS rounds up to the next whole track. This may increase the actual size of the extents allocated.

Determining storage needs for ICL

The ICL maintains a permanent record for each certificate PKI Services issues. There is one ICL record for each issued certificate. The ICL grows over time as more certificates are issued. Assuming average size certificates, one ICL record will occupy 1024 bytes of storage. However, IDCAMS allocates based on the record's maximum size, 32756. Therefore, one allocation record will hold 31 certificates (32756 / 1024 = 31). If you use the default RECORDS(50 50), each extent will hold 50 allocation records or 1599 certificates (32756 X 50 / 1024 = 1599). Because VSAM data sets can have up to 128 extents, the total number of certificates that can be stored using RECORDS(50 50) is 204672.

Summary of storage considerations for ICL

1 RECORD = 31 certificates
1 50 RECORD EXTEND = 1599 certificates
Entire data set using RECORDS(50 50) = 204672 certificates

If your anticipated needs different greatly from the above values, you need to adjust the RECORDS parameter on the DEFINE CLUSTER statement for the ICL. (This is the second DEFINE CLUSTER statement in IKYCVSAM. See "IKYCVSAM" on page 289 for a code sample of this file.)

Determining storage needs for the object store

The object store holds records to track active certificate requests. There is one object store record for each active certificate request and potentially another record to post the certificate to the LDAP directory. Object store records are not permanent. They are deleted when they are no longer needed. Unlike the ICL, the object store does not grow beyond a certain point, unless there is a spike in certificate request activity. Assuming average size certificate requests, one object store record and its companion posting record will occupy a total of 2560 bytes of

storage. IDCAMS allocates based on the maximum size, 32756. Therefore, one allocation record will hold 12 concurrent certificate requests (32756 / 2560 = 12). If you use the default RECORDS(50 50), each extent will hold 50 allocation records or 639 concurrent certificate requests (32756 X 50 / 2560 = 639). Because VSAM data sets can have up to 128 extents, the total number of concurrent certificate requests that can be stored using RECORDS(50 50) is 81792.

Summary of storage considerations for the object store

1 RECORD = 12 concurrent certificate requests
1 50 RECORD EXTEND = 639 concurrent certificate requests
Entire data set using RECORDS(50 50) = 81792 certificates

If your anticipated needs different greatly from the preceding values, you need to adjust the RECORDS parameter on the DEFINE CLUSTER statement for the object store. (This is the first DEFINE CLUSTER statement in IKYCVSAM. See "IKYCVSAM" on page 289 for a code sample of this file.)

Steps for creating the VSAM object store and ICL data sets and indexes

PKI Services uses VSAM data sets to store requests in progress and issued certificates. You need to create these data sets manually.

Perform the following steps to create the VSAM object store and ICL data sets and indexes:

1. Copy the sample JCL in 'SYS1.SAMPLIB(IKYCVSAM)' to your JCL data set. (See "IKYCVSAM" on page 289 for a code sample of this file.)

   _____

2. Update your data set as the instructions in the prolog of the sample JCL direct.
   a. You need to change the JOB card.
   b. You need to change the VOL statements.
   c. You can optionally change the data set names but must remember to make equivalent changes in the pkiserv.conf file if you do so. (See Step 1c on page 46.)
   d. Update the primary and secondary extent allocations (both are 50 by default) based on your anticipated future needs. (See "Space considerations for creating VSAM data sets" on page 59 for guidelines on determining the space you will need.) The following line shows these allocations:

      ```
      RECORDS(50 50)
      ```

   e. Do **not** change any numeric values, such as CISZ(512), other than the primary and secondary values specified for RECORDS.

   _____

3. Submit the job when your changes are complete.

   _____

Steps for starting the PKI Services daemon

Before you begin:

- Your z/OS HTTP Server should be SSL-enabled (see Chapter 7, "Updating z/OS HTTP Server configuration and starting the server" on page 51) and the uncustomized PKISERV application ready for use.

- You need to know the runtime directory, called *runtime-dir* in the command that follows. The default is /etc/pkiserv/. The MVS programmer was asked to record any changes to the default; see Table 3 on page 9.

Perform the following steps to start the PKI Services daemon and view your Web pages:

1. If you have not done so already, start the Web server and the LDAP server.

---

2. Start the PKI Services daemon from the MVS console by entering the following command:

   ```
   S PKISERVD
   ```

   Note: Depending on the amount of customization you did, there are various versions of this command to start the PKI Services daemon. For example, if you changed the pkiserv.envars file (see step 2 on page 40), you need to specify its new location as a parameter in the start command:

   ```
   S PKISERVD,DIR='runtime-dir'
   ```

   (Single quotation marks are required to maintain the character case of the values being assigned to the substitution parameters.)

   The command in the following example specifies the runtime directory and the file name of the environment variables file:

   Example:

   ```
    S PKISERVD,DIR='/etc/pkiserv',FN='pkiserv.envars'
   ```

   The default time zone is EST5EDT. If you need to change this, you can supply the new value as a parameter, as in the following examples:

   Examples:

   ```
   S PKISERVD,TZ=PST8PDT
   S PKISERVD,DIR='/etc/pkiserv',FN='pkiserv.envars',TZ=PST8PDT
   ```

---

3. Go to your Web pages by entering the following URL from your browser:

   ```
   http://webserver-fully-qualified-domain-name/PKIServ/public-cgi/camain.rexx
   ```

   The *webserver-fully-qualified-domain-name* is the common name (CN) portion of the Web server's distinguished name; see Table 11 on page 25.

   You should be able to go through your Web pages to request, retrieve, and revoke a certificate of type ″PKI browser certificate for authenticating to z/OS.″ Ensure you can do this before trying to customize the application.

Stopping the PKI Services daemon

To stop the PKI Services daemon, enter one of the following two commands. You can use either the following MODIFY (or F) console command:

```
F PKISERVD,STOP
```

or the STOP (P) command:

```
P PKISERVD
```

Part 3. Customizing PKI Services

This part of the book includes the following:

- Chapter 10, “Customizing the end-user Web pages” on page 65 provides an overview of the pkiserv.tmpl file, which contains the certificate templates, and explains customizing the end-user Web pages.
- Chapter 11, “Customizing the administration Web pages” on page 101 provides an overview of the CGI scripts and explains how to customize the administration Web pages.
- Chapter 12, “Advanced customization” on page 107 explains:
  - Using certificate policies
  - Updating the signature algorithm
  - Using the PKI exit.

Chapter 10. Customizing the end-user Web pages

This chapter describes the pkiserv.tmpl certificate templates file and explains how to use it to customize the end-user Web pages. It also explains the relationship between CGIs and the certificate templates file.

Before you begin to customize Web pages, you need to understand the pkiserv.tmpl certificate templates file. This file contains certificate templates, which define the fields that comprise a specific certificate request. (See Chapter 22, “The pkiserv.tmpl certificate templates file” on page 223 for a code sample of the templates file.) This chapter describes the main sections and subsections and explains the contents of the pkiserv.tmpl certificate templates file. It explains the minimal changes you need to make in the pkiserv.tmpl certificate templates file and how to perform additional customization to tailor the certificate templates to make them more appropriate for your organization.

Contents of the pkiserv.tmpl certificates templates file

The pkiserv.tmpl certificate templates file contains certificate templates that define the fields that comprise a specific certificate request. The file contains a mixture of true HTML and HTML-like tags. The HTML can contain JavaScript for input field verification.

The main sections of the pkiserv.tmpl certificate templates file are listed in Table 24:

Table 24. pkiserv.tmpl — Structure and main divisions

A prolog section of comments explaining main sections, subsections, named fields, and substitution variables. (To examine these comments, see Chapter 22, “The pkiserv.tmpl certificate templates file” on page 223.)
APPLICATION section The APPLICATION section contains subsections, which produce certain Web pages, such as the PKI Services Home page (see Figure 4 on page 126). For details, see “The APPLICATION section” on page 71.
TEMPLATE sections These are the certificate templates (models) that contain the HTML to produce certificate request forms. They also define the fields that are permissible in the certificate. For details, see “TEMPLATE sections” on page 74.
INSERT sections These contain HTML for certain Web pages (for example, the ″Request submitted successfully″ Web page) and certificate field dialogs (for example, text entry boxes (the common name INSERT produces a text box where the user enters this information) and drop-downs). For details, see Figure 7 on page 132.

The pkiserv.tmpl file begins with a prolog. This is a section of comments that explains the main sections and subsections of the file. Any line with a # in column 1 is a comment.

Only the APPLICATION section and TEMPLATE sections can contain subsections, but all three can contain named fields and substitution variables.

What are substitution variables?

A substitution variable holds a value that HTML code can reference. At run time, the actual value replaces a substitution variable.

You use square brackets to delineate a substitution variable.

Example:

```
[base64cert]
```

Notes:

1. Substitution variables are case-sensitive.
2. Depending on the section where a substitution variable is present, it may not have a valid meaning. For example, the base64cert substitution variable is meaningless before the certificate is retrieved. Therefore, in this case, the value of [base64cert] would be the null string (an empty string).

The following table summarizes valid substitution variables:

Table 25. Substitution variables

Substitution variable	Description
base64cert	The requested certificate, base64-encoded.
browsertype	A special substitution variable to qualify named fields only. It enables the different browsers, Netscape and Internet Explorer, to perform browser-specific operations, such as generating a public and private key pair. To do this, Netscape uses a KEYGEN HTML tag while Internet Explorer uses ActiveX controls. For example, suppose you specify `%%PublicKey[browsertype]%%` in a TEMPLATE CONTENT section. If the user referencing this section uses the Netscape Navigator browser, then INSERT PublicKeyNS is included. If the user's browser is Microsoft Internet Explorer, INSERT PublicKeyIE is included.
iecert	The requested certificate in a form that Microsoft Internet Explorer accepts.
optfield	A special substitution variable that should be placed in any certificate field name INSERT where the end user can supply the value. It makes the input field optional.
printablecert	This contains the certificate details so that the end user can confirm that the certificate is the correct one to renew or revoke. The displayed data is extracted from the ICL entry.
tmplname	A certificate template name. This is primed from the HTML tag `<SELECT NAME="Template">` in the `<APPLICATION NAME=PKISERV>` section. The end user selects it on the first Web page.
transactionid	A unique value returned from a certificate request.

What are named fields?

Named fields insert common HTML code, such as a common input field or a page header or footer, in a Web page. (Each named field refers to a corresponding INSERT section.) A named field is delineated with `%%`.

Examples:

```
%%Country%%
%%-pagefooter%%
```

Note: Named fields are case-sensitive.

A named field can include or not include a dash. A named field without a dash, such as `%%Label%%` may have a special meaning as a certificate field. Its special meaning depends on the section in which it appears. (See “Relationship between CGIs and the pkiserv.tmpl file” on page 88 for more information.)

A named field with a dash, such as `%%-pagefooter%%`, has no special meaning. PKISERV treats it simply as HTML code to insert. Any special meaning the named field might have, based on the section in which it is contained, is ignored. For example, in a TEMPLATE CONTENT section (see “TEMPLATE sections” on page 74) if you specify `%%-pagefooter%%`, `-pagefooter` is not considered a certificate field name. However, the INSERT section with the name `-pagefooter` is included in the HTML page displayed to the end user.

INSERT sections

Although the INSERT sections are at the end of the pkiserv.tmpl certificate templates file, they are explained first because of their relationship to named fields. As previously indicated, any named field used in the pkiserv.tmpl file must be defined in a corresponding INSERT section.

Unlike the APPLICATION section and TEMPLATE sections, INSERT sections can have no subsections. The following is the format of an INSERT section:

<INSERT NAME=*insert-name***>...</INSERT >**

An INSERT contains HTML that either:

- defines a certificate field
- defines other common HTML that can be referenced in other sections.

The following example of an INSERT defines a certificate field.

Example:

```
<INSERT NAME=Country>
<p> Country [optfield] <BR>
<INPUT NAME="Country" TYPE="text" SIZE=2 maxlength="2">
</INSERT>
```

The next example defines other common HTML:

Example:

```
<INSERT NAME=-pagefooter>
<p>email: webmaster@your_company.com
</INSERT>
```

To reference an INSERT, you use a named field of the form `%%insert-name%%`, for example `%%Country%%` or `%%-pagefooter%%`.

The pkiserv.tmpl certificate templates file contains INSERT sections of several main types:

- Sample INSERTs, which are includable code inserts (This is common HTML for Web page content as listed in Table 26 on page 68)
- Certificate fields that are defined in INSERT sections. (See Table 27 on page 68.) These include:

– X.509 fields (for example, OrgUnit)
– non-X.509 fields (for example UserId).

Table 26. Sample INSERTs

INSERT NAME	Contents
-AdditionalHeadIE	ActiveX controls to enable Internet Explorer to generate a key pair
-requestok	HTML for the Web page ″Request submitted successfully″ after a successful certificate request (for both original requests and renewals). (For a sample of this Web page, see Figure 7 on page 132.)
-requestbad	HTML for the Web page that says, ″Request was not successful″
-renewrevokeok	HTML for the Web page that says, ″Request submitted successfully″ after a successful attempt to revoke a certificate (see Figure 12 on page 138 for a sample of the Web page to renew or revoke a certificate)
-renewrevokebad	HTML for the Web page that says, ″Request was not successful″ after an unsuccessful attempt to renew or revoke a certificate (see Figure 12 on page 138 for a sample of the Web page to renew or revoke a certificate)
-returnpkcs10cert	This returns a PKCS #10 certificate.
returnbrowsercertNS	This contains [base64cert], which is the base64 substitution variable.
returnbrowsercertIE	This contains a script for producing a popup window installing your certificate (if you are using the Microsoft Internet Explorer browser). See Figure 9 on page 134 for a sample of this Web page.

Named fields in INSERT sections

Most of the following fields are X.509 fields. The following table summarizes the named fields in INSERT sections:

Table 27. Named fields in INSERT sections

Field	Description
AltDomain	The host name of the machine where a certificate will be installed. This is a text field of up to 100 characters. **Note:** The value is one of the list of subject's alternate names that is saved in the subject alternate name extension in the certificate.
AltEmail	The user's e-mail address, including the @ character and any periods (.). This is a text field of up to 100 characters. **Note:** The value is one of the list of subject's alternate names that is saved in the subject alternate name extension in the certificate.
AltIPAddr	The unique IP version 4 address that specifies the location of the server or device on the Internet, for example, 9.67.97.103. (PKI Services supports only IP version 4 addresses.) The IP address is in dotted decimal format and is a text field of up to 15 characters. **Note:** The value is one of the list of subject's alternate names that is saved in the subject alternate name extension in the certificate.
AltURI	A name or address referring to an internet resource; a URL is one kind of uniform resource identifier. This is a text field of up to 100 characters. **Note:** The value is one of the list of subject's alternate names that is saved in the subject alternate name extension in the certificate.

Table 27. Named fields in INSERT sections (continued)

Field	Description
ChallengePassPhrase	The passphrase the user entered when requesting a certificate. The user types the same passphrase, exactly as entered on the request form. This is a case-sensitive text field of up to 32 characters.
CommonName	For browser certificates, this is your name, such as John Smith. (You can use your first and last name, in that order.) For server certificates, this is name by which the server's administrator wants it to be known. For SSL servers, the SSL protocol requires the CommonName to be the fully qualified domain name of the server, for example, `www.ibm.com`. CommonName is a text field of up to 64 characters. See the note on page 82 for more information about this field.
Country	The country where your organization is located. This is a 2-character text field.

HostIdMap	This is the user ID for authorization purposes, in an e-mail type of format:
	`subject-id@host-name`
	
	For example, this could be `dsmith@ibm.com`. This is a text field of up to 100 characters.
	
	There are three ways to use %%HostIdMap%%:
	• If you place it in the CONTENT section, the end user can specify the value (or values since it may be repeated).
	• You can also place it in the APPL section that the application provides. If you do so, it should have the following form:
	`%%HostIdMap=@host-name%%`
	
	The `host-name` is the hardcoded system name for the current system.
	
	The application provides the user ID as the user entered it when prompted for user ID and password. Note that, for this to function properly, the z/OS HTTP Server protection scheme for the request must force a prompt for user ID and password. Thus, only one HostIdMap is provided using this method.
	• A third way to specify HostIdMap is to place `%%HostIdMap%%` in the ADMINAPPROVE section. This allows the administrator to fill in the value when approving the certificate request. See "Administering HostIdMappings extensions" on page 168 for more information.

KeyProt	(This is for the Internet Explorer browser only.) This asks if the user wants to enable strong private key protection. The drop-down choices are Yes and No.
KeyUsage	The intended purpose of the certificate. Possible values are: • handshake — Protocol handshaking (for example, SSL) • dataencrypt — Data encryption • certsign — Certificate signing • docsign — Document signing
Label	The label assigned to the requested certificate. This is a text field of up to 32 characters.
Locality	The city or municipality where your organization is located, such as Pittsburgh or Paris. This is a text field of up to 64 characters.
NotBefore	Number of days (0 or 30) before the certificate becomes valid.
NotAfter	Length of time that the certificate is current. This is 365 days (1 year) or 720 days (2 years).
Org	Organization. The legally registered name (or trademark name, for example, IBM) of your organization. This is a text field of up to 64 characters.
OrgUnit	The name of your division or department. This is a text field of up to 64 characters.

Table 27. Named fields in INSERT sections (continued)

Field	Description
OrgUnit2	The name of your division or department. (There can be more than one organizational unit field on a request form. For example, one could be for your department and another for your division.) This is a text field of up to 64 characters.
PassPhrase	The user decides this and enters and then reenters it when requesting a certificate (and must later supply this value when retrieving the certificate). This is a case-sensitive text field of up to 32 characters. There is no minimum number of characters, and the user can use any characters, but alphanumeric characters (A–Z, a–z, and 0–9) are recommended.
PublicKey	The base64-encoded PKCS #10 certificate request. (This is for server or device enrollment only.) You create a certificate request on behalf of another server (which could be a z/OS server or other type of server) or device for which you are requesting a certificate. You use software specific to that server to generate the PKCS#10 request before going to the PKI Services Web site. Save the request in a file. Then open the file in a text editor such as Windows Notepad and copy the and paste the contents into the text box on the enrollment form. A text area of 70 columns and 12 rows is allocated for this certificate request. Here is an example of the certificate request: `MIIBiDCB8gIBADAZMRcwFQYDVQQDEw5Kb2huIFEuIFB1YmxpYzCBnzANBgkqhkiG 9w0BAQEFAAOBjQAwgYkCgYEAsCT1cJHAGPqi6OjAyL+xNbt8z5ngmvqO2V0O3oYu /mEnQtRM96e+2jbmDCRo5tWVklG4OYf9ZVB5biURMJFLztfa4AVdEVtun8DH2pwc wiNIZZcC1Zym5adurUmyDk64PgiiIPMQS/t0ttG4c5U8uWSK0b1J4V4f7ps+t1aG t+cCAwEAAaAwMC4GCSqGSIb3DQEJDjEhMB8wHQYDVR0OBBYEFAlKTovBBvnFqDAO 1oIhtRinwRC9MA0GCSqGSIb3DQEBBQUAA4GBAIbCVpwYvppIX3HHmpKZPNY8Snsz AJrDsgAEH51WOIRGywhqKcLLxa9htoQai6cdc8RpFVTwk6UfdCOGxMn4aFb34Tk3 5WYdzOiHXg8MhHiB3EruwdWs+S7Fv3JhU3FLwU6lFLfAjbVi+35iEWQymOR6mE5W CathprmGfKRsDE5E`</br>
PublicKeyIE	(This is for the Internet Explorer browser only.) This is the cryptographic service provider. The user selects a value from a drop-down list (Microsoft Base Cryptographic Provider or Microsoft Enhanced Cryptographic Provider).
PublicKeyNS	(This is for the Netscape browser only.) This is the key size for your public/private key pair. The user selects a value from the drop-down list. Larger keys are more secure, but they also increase the time needed for connecting to a secure session.
Requestor	The user's name, used for tracking the request. This can be in any format, for example, John Smith or John. J. Smith. (This can differ from the common name, especially if the request is for a server certificate.) The value is saved with the request and issued certificate, but it is not a field in the created certificate. The default value is taken from the leftmost RDN in the subject's distinguished name, truncated to 32 characters.

SignWith	For PKI the component and for SAF the component and key-label used to sign this certificate, indicating the provider for certificate generation. This is a text field of up to 45 characters. It can be SAF or PKI Services, as shown in the following examples.
	Examples:
	`"SAF:CERTAUTH/Local CA Cert"`
	`"PKI:"`
	For SAF, the label of the signing certificate must be included. The first example shows the SignWith field in a SAF template. It includes the signing certificate, a CERTAUTH certificate labeled 'Local CA Cert'.
	For PKI, it is an error to include the signing certificate. The second example shows the SignWith field in a PKI template. Notice that this contains no signing certificate.

Table 27. Named fields in INSERT sections (continued)

Field	Description
StateProv	The state or province where your organization is located. Your registration policies determine whether you spell out the full name of the state or province or use an abbreviation. This is a text field of up to 64 characters.
Title	Job title. This is a text field of up to 64 characters.
TransactionId	PKISERV Web pages assign this after the user requests a certificate. When it is displayed, the user needs to record this number. This is a text field of up to 56 characters.
UserId	The owning SAF user ID. This is a text field of up to 8 characters.

The APPLICATION section

The APPLICATION section identifies the applications that will use PKI Services. The following is the format of the APPLICATION section:

<APPLICATION NAME=appl-name>...</APPLICATION>

The product ships with one application, PKISERV. Therefore, the pkiserv.tmpl certificate templates file that ships with PKI Services contains the following line:

Example:

```
<APPLICATION NAME="PKISERV">
```

The APPLICATION section can contain the following subsections:

- CONTENT
- RECONTENT
- RESUCCESSCONTENT
- REFAILURECONTENT
- ADMINHEADER
- ADMINFOOTER

<CONTENT> ...</CONTENT>

This subsection contains the HTML to display the PKI Services home Web page to the end user who is requesting and retrieving certificates. (See Figure 4 on page 126 for a sample Web page.) This subsection should contain one or more named fields (see "What are named fields?" on page 66) identifying certificate templates to use for requesting or managing certificates through this application. These template names should match the HTML selection value associated with them.

<RECONTENT> ...</RECONTENT>

This subsection contains the HTML to display information about the certificate so that the end user can confirm that this is the correct certificate to renew or revoke. (See Figure 12 on page 138 for a sample Web page.) This subsection uses the substitution variable [printablecert], which contains the data extracted from the ICL entry. (See "What are substitution variables?" on page 66.)

<RESUCCESSCONTENT> ...</RESUCCESSCONTENT>

This subsection contains the HTML to display a Web page to the end user when the revocation request is successful. Any named fields in this subsection are interpreted as HTML content inserts (for example, a page footer) that INSERT sections define. For

PKISERV, the INSERT sections are included as part of the HTML for the Web page displayed to the end user.

<REFAILURECONTENT> ...</REFAILURECONTENT>
This subsection contains the HTML to display a Web page to the end user when renewal or revocation request is unsuccessful. Any named fields in this subsection are interpreted as content inserts (for example, a page footer) that INSERT sections define. For PKISERV, the INSERT sections are included as part of the HTML for the Web page displayed to the end user.

<ADMINHEADER>...</ADMINHEADER>
This subsection contains the general installation-specific HTML content for the header of all administration Web pages. See “Steps for customizing the administration Web pages” on page 103 for more information.

<ADMINFOOTER>...</ADMINFOOTER>
This subsection contains the general installation-specific HTML content for the footer of all administration Web pages. See “Steps for customizing the administration Web pages” on page 103 for more information.

The following table summarizes the contents (Web pages) that the subsections of the APPLICATION section generate.

Table 28. Subsections of the APPLICATION section

Section or subsection	Contents
CONTENT	HTML for the Web page ″PKISERV certificate generation application.″ (For a sample of this Web page, see Figure 4 on page 126.)
RECONTENT	HTML for the Web page ″Renew or revoke a browser certificate.″ (For a sample of this Web page, see Figure 12 on page 138.)
RESUCCESSCONTENT	Contains only the named field %%-renewrevokeok%% (whose associated INSERT contains HTML for the Web page ″Request submitted successfully″).
REFAILURECONTENT	Contains only the named field %%-renewrevokebad%% (whose associated INSERT contains HTML for the Web page ″Request was not successful″).
ADMINHEADER	This is for an administration page; see “Customizing the administration Web pages” on page 102 for more information.
ADMINFOOTER	This is for an administration page; see “Customizing the administration Web pages” on page 102 for more information.

Templates that PKI Services provides

PKI Services provides the templates to request the following certificates:

- One-year SAF server certificate
- One-year SAF browser certificate
- One-year PKI SSL browser certificate (See Figure 6 on page 131 to see a sample of this Web page.)
- One-year PKI SSL S/MIME browser certificate

- Two-year PKI browser certificate for authenticating to z/OS
- Five-year PKI SSL server certificate
- Five-year PKI IPSEC server (firewall) certificate
- Five-year PKI intermediate CA certificate

The following table describes the certificate templates that PKI Services provides:

Table 29. Certificate templates PKI Services provides

Certificate template	Description
One-year SAF server certificate	The template allows end users to request certificates for servers, using native SAF certificate generation facilities (rather than PKI Services certificate generation facilities). The certificate is used for handshaking only (for example, SSL). This certificate is auto-approved.
One-year SAF browser certificate	This template is for requesting a browser certificate. SAF certificate generation facilities (rather than PKI Services certificate generation facilities) create the certificate. The requestor must input a label (see Table 27 on page 68 for descriptions of fields) because the certificate is stored in a RACF database. This certificate is auto-approved.
One-year PKI SSL browser certificate	A template for requesting a browser certificate that PKI Services generates. The end user enters the common name. (See Table 27 on page 68 for descriptions of fields.) This template contains an ADMINAPPROVE section. Therefore, certificates requested using this template require administrator approval before being issued. The user ID and password are not required but the passphrase is required.
One-year PKI S/MIME browser certificate	A template for requesting a browser certificate that PKI Services generates. This is similar to the one-year PKI SSL browser certificate except the end user selects AltEmail.
Two-year PKI browser certificate for authenticating to z/OS	A template for requesting a browser certificate that PKI Services generates. This is similar to the one-year PKI SSL browser certificate except this includes the %%HostIdMap%% INSERT and this certificate is auto-approved. %%HostIdMap%% is intended as a replacement for adding (and mapping) the certificate to a RACF user ID. This template specifies `%%HostIdMap=@ host-name%%` and `%%UserId%%` in the APPL section. This template does not require administrator approval but has protection through the user ID and password. (For more information about %%HostIdMap%%, see the HostIdMap field in Table 27 on page 68.)
Five-year PKI SSL server certificate	A template for requesting a server certificate that PKI Services generates. This is similar to the SAF server template except that this template contains an ADMINAPPROVE section. Therefore, certificates requested using this template require administrator approval before being issued. The user ID and password are not required but the passphrase is required.
Five-year PKI IPSEC server (firewall) certificate	A template for requesting a server certificate that PKI Services generates. This is similar to the five-year PKI SSL server certificate except that keyusages of handshake and dataencrypt are hardcoded. Also, the end user selects AltEmail, AltIPAddr, AltURI, and AltDomain.

Table 29. Certificate templates PKI Services provides (continued)

Certificate template	Description
Five-year PKI intermediate CA certificate	A template for requesting a server certificate that PKI Services generates. This is similar to the PKI SSL server template except that KeyUsage is hardcoded as certsign. Also, this certificate is auto-approved (because it runs under the user ID of the requestor, that is the person requesting this must be highly authorized). The user ID and password are required, and the units of work should run under the client's ID. In other words, the end user must be someone who can do this using RACDCERT alone, that is, must have CONTROL authority to IRR.DIGTCERT.GENCERT, and so forth. Given this requirement, the administrator need not approve this. The PassPhrase is required.

TEMPLATE sections

TEMPLATE sections define the fields that comprise a specific certificate request. They define the certificate templates referenced in the APPLICATION section. The pkiserv.tmpl certificate templates file contains eight TEMPLATE sections, for the eight certificates the preceding section describes.

Each template section begins with one or more template names.

<TEMPLATE NAME=tmpl-name>...</TEMPLATE NAME>

The pkiserv.tmpl certificate templates file that ships with PKI Services includes lines like the following:

Example:

```
<TEMPLATE NAME=1 Year PKI SSL Browser Certificate>
<TEMPLATE NAME=PKI Browser Certificate>
<NICKNAME=1YBSSL>
```

The true name of a certificate template is its actual complete name. This is the name in the first line, `1 Year PKI SSL Browser Certificate`, However, you can refer to a single template by more than one name by using an alias. The template name in the second line, `PKI Browser Certificate`, is an alias. An alias simply differentiates browser from server certificates. Finally, renewing a certificate requires recalling the template name, so the template name must be stored with the certificate. The NICKNAME (or short name) serves this purpose.

Notes:

1. You can have more than one alias. (Use an additional <TEMPLATE NAME=*alias*> line for each one.)
2. The value of a NICKNAME is an 8-character string.
3. SAF certificate templates do not include nicknames.

The following table shows the true name, alias, and nickname for each certificate template:

Table 30. Names of certificate templates

True name	Alias	Nickname
1 Year PKI SSL Browser Certificate	PKI Browser Certificate	1YBSSL
1 Year PKI S/MIME Browser Certificate	PKI Browser Certificate	1YBSM
2 Year PKI Browser Certificate For Authenticating To z/OS	PKI Browser Certificate	2YBZOS

Table 30. Names of certificate templates (continued)

True name	Alias	Nickname
5 Year PKI SSL Server Certificate	PKI Server Certificate	5YSSSL
5 Year PKI IPSEC Server (Firewall) Certificate	PKI Server Certificate	5YSIPS
5 Year PKI Intermediate CA Certificate	PKI Server Certificate	5YSCA
1 Year SAF Server Certificate	SAF Server Certificate	none
1 Year SAF Browser Certificate	SAF Browser Certificate	none

TEMPLATE sections can have the following subsections:

- CONTENT
- APPL
- CONSTANT
- ADMINAPPROVE
- SUCCESSCONTENT
- FAILURECONTENT
- RETRIEVECONTENT
- RETURNCERT

<CONTENT>...</CONTENT>

This subsection contains the HTML to display a Web page to the end user requesting a certificate of a specific type. (See Figure 6 on page 131 for a sample Web page.) Field names on the certificate request (such as a text box where the user enters a value for `Common Name`) match the names of INSERT sections. The following examples show the INSERT sections corresponding to the field names `%%CommonName%%` and `%%Requestor (optional)%%`:

Examples:

```
<INSERT NAME=CommonName>
<p> Common Name [optfield]
<BR>
<INPUT NAME="CommonName" TYPE="text" SIZE=64  maxlength="64">
<INSERT>


<INSERT NAME=Requestor>
<p> Your name for tracking this request [optfield] <BR>
<INPUT NAME="Requestor" TYPE="text" SIZE=32 maxlength="32">
<INSERT>
```

Named fields in this subsection are optional if the named field contains more that one word within the `%%` delimiters (as in `%%Requestor (optional)%%`). The user need not supply a value for Requestor.

<APPL>...</APPL>

This subsection identifies certificate fields for which the application itself should provide values. This subsection should contain only named fields, one per line. The only supported named fields allowed in this section are:

- UserId
- HostIdMap

Example:

```
<APPL>
 %%UserId%%
 %%HostIdMap=@www.ibm.com%%
<APPL>
```

<CONSTANT>...</CONSTANT>

This subsection identifies certificate fields that have a constant (hardcoded) value for everyone. This subsection should contain only named fields, one per line. The syntax for specifying the values is `%%field-name=field-value%%`:

Example:

```
%%KeyUsage=handshake%%
```

<ADMINAPPROVE>...</ADMINAPPROVE>

This optional subsection contains the named fields that the administrator can modify when approving certificate requests. (The named fields refer to INSERT sections.) When an end user requests a certificate, the certificate request may contain fields that the end user cannot see. When approving a request, the administrator can modify:

- Fields that are present and visible to the end user in the certificate request, for example `Common Name`
- Fields that are not visible to the end user but are hardcoded (in the CONSTANT subsection) in the template, for example `Organizational unit`
- Fields that are not visible to the end user and that the PKI Services administrator can add, for example, `HostIdMappings extension` or an empty `Organizational Unit` field (these are listed in the <ADMINAPPROVE> section, and either the end user did not fill them in or they are not present on the template request form).

The presence of this section (even if empty) indicates that an administrator must approve this request. The absence of this section indicates using auto-approval.

Note: In the pkiserv.tmpl certificate templates file, the only certificate templates that are auto-approved are the following:

- One-year SAF server certificate
- One-year SAF browser certificate
- Two-year PKI browser certificate for authenticating to z/OS
- Five-year PKI intermediate CA certificate

You can put the following fields in the ADMINAPPROVE section:

- AltDomain
- AltEmail
- AltIPAddr
- AltURI
- CommonName
- Country

- EndDate
- HostIdMap (can repeat)
- KeyUsage
- Locality
- Org
- OrgUnit (can repeat)
- StartDate
- StateProv
- Title

Note: The following fields are not modifiable and are ignored in the ADMINAPPROVE section:

- Label
- PublicKey
- Requestor
- SignWith
- UserId

(For information about fields, see Table 27 on page 68.)

Example:

```
<ADMINAPPROVE>
%%KeyUsage%%
%%CommonName%%
%%OrgUnit%%
%%Org%%
%%Country%%
%%HostIdMap%%
%%HostIdMap%%
%%HostIdMap%%
%%HostIdMap%%
<ADMINAPPROVE>
```

<SUCCESSCONTENT>...</SUCCESSCONTENT>

This subsection contains the HTML to display to the end user a Web page saying that the certificate request was submitted successfully. Any named fields in this subsection are interpreted as content inserts defined by INSERT sections. For PKISERV, the INSERT sections are included as part of the HTML to display a Web page to the end user.

In all of the templates included with PKI Services, <SUCCESSCONTENT> contains only the named field `%%-requestok%%`. (See “What are named fields?” on page 66 for an explanation of named fields.) This contains HTML for the Web page ″Request submitted successfully.″ (For a sample of this Web page, see Figure 7 on page 132.)

<FAILURECONTENT>...</FAILURECONTENT>

This subsection contains the HTML to display to the end user a Web page saying the certificate request was not submitted successfully. Any named fields in this subsection are interpreted as content inserts defined by INSERT sections. For PKISERV, the INSERT sections are included as part of the HTML to display a Web page to the end user.

In all of the templates included with PKI Services, <SUCCESSCONTENT> contains only the named field %%-requestbad%%. (See “What are named fields?” on page 66 for an explanation of named fields.) This contains HTML for the Web page that says, ″Request was not successful.″

<RETRIEVECONTENT>...</RETRIEVECONTENT>

This subsection contains the HTML to display to the end user a Web page to enable certificate retrieval. Any named fields in this subsection are interpreted as content inserts that the INSERT sections define. For PKISERV, the INSERT sections are included as part of the HTML presented to the end user.

For a sample of a Web page this section generates, see Figure 8 on page 133. You may want to look at this Web page while reading the following explanation:

In all of the templates included with PKI Services, <RETRIEVECONTENT> contains the following:

- The named field %%-copyright%%, which displays any copyright information. (See “What are named fields?” on page 66 for an explanation of named fields.)
- The title of the Web page (This appears in the banner of your browser. Figure 8 on page 133 does not include the banner header but shows only the frame containing the content and not the browser window displaying the content.)
- A JavaScript script for processing the fields the user enters the Web page
- A heading that says ″Retrieve Your (name of certificate).″ This uses the substitution variable [tmplname]. (See “What are substitution variables?” on page 66 for an explanation of substitution variables.
- Text: a heading and paragraph about bookmarking this Web page
- The named field %%TransactionId%% — A field where you enter your transaction ID if it is not already displayed
- A field where you enter the passphrase you entered on the certificate request form

<RETURNCERT>...</RETURNCERT>

This subsection contains the HTML to display to the end user a Web page upon successful certificate retrieval. For PKISERV, if the certificate being retrieved is a browser certificate, then this section must contain a single line containing a browser qualified INSERT name.

Example:

```
%%returnbrowsercert[browsertype]%%
```

Additionally, INSERTs for Netscape (returnbrowsercertNS) and Internet Explorer (returnbrowsercertIE) containing browser-specific HTML for returning certificates must be defined elsewhere in the pkiserv.tmpl certificates template file. If the certificate being retrieved is a server certificate, this section should contain the HTML necessary to present the certificate to the user as text.

Summary of subsections contained in certificate templates

The following table summarizes the subsections that are present in the various certificate templates in the pkiserv.tmpl file (as it is shipped):

Table 31. Summary of subsections in certificate templates

Subsection (in TEMPLATE section)	One-year SAF browser	One-year SAF server	One-year PKI SSL browser	One-year PKI SSL S/MIME browser	Two-year PKI browser cert. for authen. to z/OS	Five-year PKI SSL server	Five-year PKI IPSEC server (firewall)	Five-year PKI int. CA
CONTENT	X	X	X	X	X	X	X	X
APPL	X	X			X			X
CONSTANT	X	X	X	X	X	X	X	X
ADMINAPPROVE			X	X		X	X	
SUCCESSCONTENT	X	X	X	X	X	X	X	X
FAILURECONTENT	X	X	X	X	X	X	X	X
RETRIEVECONTENT	X	X	X	X	X	X	X	X
RETURNCERT	X	X	X	X	X	X	X	X

Summary of fields in certificate templates

Now that you are familiar with the certificate templates and the fields, the following table summarizes the fields that the various certificate templates contain:

Table 32. Summary of fields in certificate templates

Certificate Template	Required user input fields	Optional user input fields	Application provided fields	Constants (field and value)
One-year PKI SSL browser certificate	**Request Web page:** • CommonName • PassPhrase • PublicKey (The browser itself provides this.) **Retrieve Web page:** • TransactionID • ChallengePassPhrase	• Requestor (At Retrieve Web page) • ChallengePassPhrase		• NotBefore = 0 • NotAfter = 365 • KeyUsage = handshake • OrgUnit = Class 1 Internet Certificate CA • Org = The Firm • SignWith = PKI:
One-year PKI S/MIME browser certificate	**Request Web page:** • CommonName • AltEmail • PassPhrase • PublicKey (The browser provides this.) **Retrieve Web page** • TransactionID	• Requestor (At Retrieve Web page) • ChallengePassPhrase		• NotBefore = 0 • NotAfter = 365 • KeyUsage = handshake • OrgUnit = Class 1 Internet Certificate CA • Org = The Firm • SignWith = PKI:

Table 32. Summary of fields in certificate templates (continued)

Certificate Template	Required user input fields	Optional user input fields	Application provided fields	Constants (field and value)
One-year SAF server certificate	• OrgUnit • Org • Country • Label • PublicKey (Retrieve Web page) • TransactionId	• CommonName • OrgUnit2 • Locality • StateProv • AltEmail • AltDomain • AltURI • AltIPAddr	• UserId	• KeyUsage = handshake • NotAfter = 365 • SignWith = SAF:CERTAUTH/taca
One-year SAF browser certificate	**Request Web page:** • Label • PublicKey (The browser itself provides this.) **Note:** PublicKey is coded with the substitution variable browsertype. For Internet Explorer, this generates two fields: – CSP — the Cryptographic Service Provider. (Defaults to Microsoft Enhanced Cryptographic Provider) – KeyProt — Enable strong private key protection. Defaults to No. For Netscape, this generates one field: – PublicKeyNS — key size. (Defaults to high grade.) **Retrieve Web page:** • TransactionId		• UserId	• KeyUsage = handshake • NotAfter = 365 • OrgUnit = SAF template certificate • OrgUnit = Nuts and Bolts Division • Org = The Firm • Country = US • SignWith = SAF:CERTAUTH/taca • CommonName[1] =
Two-year PKI browser certificate for authenticating to z/OS	**Request Web page:** • PassPhrase • PublicKey (The browser provides this) **At Retrieve Web page:** • TransactionID	• Requestor (At Retrieve Web page) • ChallengePassPhrase	• UserId • HostIdMap **Note:** HostIdMap is formed by concatenating UserId with *@host-name.*	• NotBefore = 0 • NotAfter = 730 • KeyUsage = handshake • OrgUnit = Class 1 Internet Certificate CA • Org = The Firm • SignWith = PKI • CommonName[1] =

Table 32. Summary of fields in certificate templates (continued)

Certificate Template	Required user input fields	Optional user input fields	Application provided fields	Constants (field and value)
Five-year PKI SSL server certificate	**Request Web page:** • PassPhrase • PublicKey (The browser provides this.) **Retrieve Web page:** • TransactionID	• CommonName • OrgUnit • OrgUnit • Org • Locality • StateProv • Country • AltEmail • AltDomain • AltURI • AltIPAddr • Requestor (At Retrieve Web page) • ChallengePassPhrase		• NotBefore = 0 • NotAfter = 1825 • KeyUsage = handshake • SignWith = PKI:
Five-year PKI IPSEC server (firewall) certificate	**Request Web page:** • PassPhrase • PublicKey (This is the PKCS#10 request) **Retrieve Web page:** • TransactionId	• CommonName • OrgUnit • OrgUnit2 • Org • Locality • StateProv • Country • AltEmail • AltDomain • AltURI • AltIPAddr • Requestor (At Retrieve Web page) • ChallengePassPhrase		• KeyUsage = handshake • KeyUsage = dataencrypt • NotBefore = 0 • NotAfter = 1825 • SignWith = PKI: **Note:** You can have more than one kind of KeyUsage.
Five-year PKI intermediate CA certificate	**Request Web page:** • PassPhrase • PublicKey **Retrieve Web page:** • TransactionID	• CommonName • OrgUnit • OrgUnit2 • Org • Locality • StateProv • Country • AltEmail • AltDomain • AltURI • AltIPAddr • Requestor (At Retrieve Web page) • ChallengePassPhrase	• UserId	• NotBefore = 0 • NotAfter = 1825 • KeyUsage = certsign • SignWith = PKI:

Table 32. Summary of fields in certificate templates (continued)

Certificate Template	Required user input fields	Optional user input fields	Application provided fields	Constants (field and value)

Note:

1 Although CommonName is a constant, no value is assigned to it. This indicates that RACF must determine the value. The user authenticates by specifying a user ID and password. (If UserId is listed in the APPL section, this means the application provides the user ID and password.) Providing the user ID and password enables RACF to look up the CommonName value in the user's profile.

Examining the pkiserv.tmpl file

After the initial section of comments, the next section of the pkiserv.tmpl file is the APPLICATION section. The following example shows the APPLICATION section. (The vertical ellipses indicate omitted sections.)

```
<APPLICATION NAME=PKISERV> 1
<CONTENT> 2
<HTML><HEAD>
<TITLE> Web Based Certificate Generation Application </TITLE>
%%-copyright%%
</HEAD>
<BODY>
<H1>PKISERV Certificate Generation Application</H1>
<p>
<A HREF="/PKIServ/cacerts/cacert.der">Install 3
 our CA certificate into your browser </A>
<H2>Choose one of the following:</H2>
<ul>
<li><h3>Request a new certificate using a model</h3>
<FORM name=mainform METHOD=GET ACTION="/PKIServ/ssl-cgi/catmpl.rexx"> 4
<p> Select the certificate template to use as a model
<SELECT NAME="Template"> 5
 %%1 Year PKI SSL Browser Certificate%%
     <OPTION>1 Year PKI SSL Browser Certificate
 %%1 Year PKI S/MIME Browser Certificate%%
     <OPTION>1 Year PKI S/MIME Browser Certificate
 %%2 Year PKI Browser Certificate For Authenticating To z/OS%%
.
.
.
</HTML>
</CONTENT>
<RECONTENT> 6
<HTML><HEAD>
<TITLE> PKISERV Renew or Revoke a Browser Certificate </TITLE>
%%-copyright%%
</HEAD>
<BODY>
<H1>Renew or Revoke a Browser Certificate</H1>
.
.
.
</BODY>
</HTML>
</RECONTENT>
<RESUCCESSCONTENT> 7
 %%-renewrevokeok%%
</RESUCCESSCONTENT>
<REFAILURECONTENT> 8
 %%-renewrevokebad%%
</REFAILURECONTENT>
<ADMINHEADER> 9
<HTML><HEAD>
<TITLE> Web Based Certificate Generation Administration </TITLE> 10
```

```
%%-copyright%%
</HEAD>
<BODY>
</ADMINHEADER>
<ADMINFOOTER>
<p> %%-pagefooter%% 11
</BODY>
</HTML>
</ADMINFOOTER>
</APPLICATION>
```

The numbers in the following list refer to the highlighted items in the preceding example:

1. This is the beginning of the APPLICATION section. The name of the application is PKISERV.
2. This is the beginning of the CONTENT subsection. The CONTENT subsection contains HTML to display the Web page where the end user requests or retrieves a certificate. The <H1> indicates the main heading of that Web page, ″Web Based Certificate Generation Application.″ (See Figure 4 on page 126 for a sample of that Web page.)
3. The HREF tag is the link to install the certificate in the browser.
4. The ACTION tag indicates where to go when the user clicks the **Request certificate** button.
5. The SELECT tag produces a drop-down that lists the certificate templates the user can request. (The named fields, which are bracketed with %% symbols, are the names of the certificate templates.)
6. The RECONTENT section contains the HTML to display the Web page where the end user renews or revokes a certificate. The main heading on this Web page is ″Renew or Revoke a Browser Certificate.″ (See Figure 12 on page 138 for a sample of that Web page.)
7. The RESUCCESSCONTENT subsection references the %%-renewrevokeok%% named field, which is defined in the INSERT section. This contains HTML for the Web page displayed when the user's attempt to revoke a certificate is successful. The main heading on this Web page is ″Request submitted successfully.″ (See Figure 7 on page 132 for a sample of that Web page.)
8. The REFAILURECONTENT subsection references the %%-renewrevokebad%% named field, which is defined in the INSERT section. This contains HTML for the Web page displayed when the user's attempt to renew or revoke a certificate fails. The main heading on this Web page is ″Request was not successful.″
9. The ADMINHEADER subsection references the %%-copyright%% named field, which is defined in the INSERT section. This should contain the copyright statement for your company.
10. The title appears in the banner across the very top of the browser window.
11. The ADMINFOOTER subsection references the %%-pagefooter%% named field, which is defined in the INSERT section. This named field should specify the e-mail address of your PKI Services administrator.

The TEMPLATE sections follow the APPLICATION section. The following example shows a TEMPLATE section. (The vertical ellipses indicate omitted sections.)

```
# ====================================================================
#
# Template Name - 2 Year PKI Browser Certificate For Authenticating
#                 to z/OS 1
```

```
#
# Function - Creates a 2 year certificate good for authenticating to
#            z/OS....
:
:
#
# User input fields:
#  Requestor - optional
#  PassPhrase - required
#  PublicKey - required (Provided by the browser itself)
:
:
# ====================================================================
#
<TEMPLATE NAME=2 Year PKI Browser Certificate For Authenticating To z/OS> 2
<TEMPLATE NAME=PKI Browser Certificate>
<NICKNAME=2YBZOS>
<CONTENT> 3
<HTML><HEAD>
<TITLE> Web Based PKIX Certificate Generation Application Pg 2</TITLE> 4
%%-copyright%% 5
%%-AdditionalHead[browsertype]%%
<SCRIPT LANGUAGE="JavaScript"> 6
<!--
:
:
//-->
</SCRIPT>
</HEAD>

<BODY>
<H1>2 Year Browser Certificate For Authenticating To z/OS</H1> 7
<p>
<H2>Choose one of the following:</H2>
:
:
#<FORM NAME="CertReq" METHOD=POST ACTION=
#              "/PKIServ/ssl-cgi-bin/careq.rexx" onSubmit= 8
    "if(ValidateEntry()) return false; else return true;">

<INPUT NAME="Template" TYPE="hidden" VALUE="[tmplname]">
<p> Enter values for the following field(s) 9
 %%Requestor (optional)%%
 %%PassPhrase%%
 %%PublicKey2[browsertype]%%
<INPUT TYPE="reset" VALUE="Clear">
</FORM>
<p>
<H3><li>Pick Up a Previously Issued Certificate</H3>
<FORM METHOD=GET ACTION="/PKIServ/ssl-cgi/caretrieve.rexx">
<INPUT NAME="Template" TYPE="hidden" VALUE="[tmplname]">
<INPUT TYPE="submit" VALUE="Retrieve your certificate">
</FORM>
</ul>
<p>%%-pagefooter%% 10
</BODY>
</HTML>
</CONTENT>
<APPL> 11
 %%UserId%%
 %%HostIdMap=@host-name%%
</APPL>
<CONSTANT> 12
 %%NotBefore=0%%
 %%NotAfter=730%%
 %%KeyUsage=handshake%%
 %%OrgUnit=Class 1 Internet Certificate CA%%
 %%Org=The Firm%%
 %%SignWith=PKI:%%
 %%CommonName=%%
```

```
</CONSTANT>
<SUCCESSCONTENT>  13
 %%-requestok%%
</SUCCESSCONTENT>
<FAILURECONTENT>  14
 %%-requestbad%%
</FAILURECONTENT>

<RETRIEVECONTENT> 15
<HTML><HEAD>
%%-copyright%%
<TITLE> Web Based PKIX Certificate Generation Application Pg 3</TITLE>
<SCRIPT LANGUAGE="JavaScript">
<!--
:
//-->
</SCRIPT>
</HEAD>

<BODY>
<H1> Retrieve Your [tmplname]</H1>  16
<H3>Please bookmark this page</h3>
:
#<FORM NAME=retrieveform METHOD=POST ACTION=
#      "/PKIServ/ssl-cgi-bin/auth/cagetcert.rexx" onSubmit=  17
:
</FORM>
:
<p>%%-pagefooter%%
</BODY>
</HTML>
</RETRIEVECONTENT>
<RETURNCERT>  18
%%returnbrowsercert[browsertype]%%
</RETURNCERT>
</TEMPLATE>
```

The numbers in the following list refer to the highlighted items in the preceding example:

1. The template begins with a block comment identifying the template and explaining its use and fields.
2. There are three names for each certificate (except for SAF templates, which do not include nicknames). The first TEMPLATE NAME line defines the true (actual, complete) name of the certificate. The next TEMPLATE NAME line defines an alias. (This simply differentiates browser from server certificates.) The NICKNAME defines an 8-character string.
3. The CONTENT subsection contains the HTML to display a Web page to the end user requesting this type of certificate. (The CGI script catmpl.rexx displays this content.)
4. The title contains the heading that appears at the very top of the browser when the Web page is displayed.
5. The %%-copyright%% named field displays the copyright statement.
6. This JavaScript script provides the underlying logic for the text entry that the user must perform.
7. The heading is the main heading on the Web page for requesting the selected certificate.
8. The ACTION tag indicates that the CGI script that gets control when the user clicks the **Submit certificate request** button is careq.rexx.

9. Fields for which the user can supply input include %%Requestor%%, %%PassPhrase%%, and %%PublicKey2%%. (These fields are named fields that are defined in the INSERT section, which is shown later.) All fields not marked optional are required. %%PublicKey2%% contains the substitution variable, `[browsertype]`. This is replaced at run time with `IE` or `NS`, depending on the browser the user has. This is necessary because the browsers behave differently for key generation and certificates.
10. The %%-pagefooter%% named field is defined in the INSERT section (shown later). This contains the e-mail address of the PKI Services administrator.
11. The APPL subsection indicates the fields that careq.rexx itself provides, in this case, %%UserId%% and %%HostIdMap%%. (These are set from the z/OS HTTP Server environment variable REMOTE_USER.)
12. The CONSTANT subsection has hardcoded values to use, for example (for the non-SAF certificates), the signing certificate is `PKI:`.
13. The SUCCESSCONTENT subsection contains the HTML to display upon successfully requesting the certificate. It includes the %%-requestok%% named field. (This is defined in the INSERT section, shown later. See Item 1 on page 88.)
14. The FAILURECONTENT subsection contains the HTML to display when the certificate request is unsuccessful. This subsection contains the %%-requestbad%% named field. (This named field is defined in the INSERT section, shown later.)
15. The requestok INSERT (mentioned in Item 13) includes an ACTION that calls caretrieve.rexx, which displays the HTML in the RETRIEVECONTENT subsection. The first time the Web page is displayed, it includes the transaction ID associated with the certificate request. If the user leaves the Web page and then returns, the transaction ID field must be filled in. Entering the transaction ID and clicking the **Continue** button calls cagetcert.rexx.
16. The main heading on the Web page is ″Retrieve Your (Name of Certificate).″
17. The ACTION is to call cagetcert.rexx as Item 15 indicates.
18. The RETURNCERT subsection contains the %%returnpkcs10cert%% named field, which is defined in an INSERT. (See Item 4 on page 88.)

The final section of the pkiserv.tmpl certificate templates file includes sample INSERTS. The following example shows sample INSERTS. (The vertical ellipses indicate omitted sections.)

```
# ==================================================================
#
# Sample INSERTS
#
# ==================================================================
#
<INSERT NAME=-AdditionalHeadIE>
<OBJECT
  classid="clsid:43F8F289-7A20-11D0-8F06-00C04FC295E1"
  CODEBASE="xenroll.cab"
  id="certmgr"
>
</OBJECT>
</INSERT>

<INSERT NAME=-requestok>    1
<HTML><HEAD>
<TITLE> Web Based Certificate Generation Success</TITLE>
</HEAD>
<BODY>
<H1> Request submitted Successfully</H1>
```

```
[errorinfo]
<p> Here's your transaction ID. You will need it to retrieve your
certificate.  Press 'Continue' to retrieve the certificate.
<p> <TABLE BORDER><TR><TD>[transactionid]</TD></TR></TABLE>
<FORM METHOD=GET ACTION="/PKIServ/ssl-cgi/caretrieve.rexx"> 2
<INPUT NAME="Template" TYPE="hidden" VALUE="[tmplname]">
<INPUT NAME="TransactionId" TYPE="hidden" VALUE="[transactionid]">
<INPUT TYPE="submit" VALUE="Continue">
</FORM>
<p>%%-pagefooter%%
</BODY>
</HTML>
</INSERT>

<INSERT NAME=-requestbad> 3
<HTML><HEAD>
<TITLE> Web Based Certificate Generation Failure</TITLE>
</HEAD>
<BODY>
<H1> Request was not successful</H1>
<p> Please correct the problem or report the error to your Web admin
person<br>
<PRE>
[errorinfo]
</PRE>
<p>%%-pagefooter%%
</BODY>
</HTML>
</INSERT>
:
<INSERT NAME=-returnpkcs10cert> 4
<HTML><HEAD>
<TITLE> Web Based SAF Certificate Generation Application Pg 4</TITLE>
</HEAD>
<BODY>
<H1> Here's Your Certificate. Cut and Paste it to a File</H1>
<TABLE BORDER><TR><TD>
<PRE>
[base64cert] 5
</PRE>
</TD></TR></TABLE>
<p>%%-pagefooter%%
</BODY>
</HTML>
</INSERT>
:
:
</BODY>
</HTML>
</INSERT>
#
# ====================================================================
#
# X.509 fields (INSERTs) valid for certificate requests
#
# ====================================================================
#
<INSERT NAME=PublicKeyIE> 6
<SCRIPT LANGUAGE="VBScript">
<!--
:
:
// -->
<
:
:
// -->
</SCRIPT>
```

```
# ====================================================================
:
<INSERT NAME=PassPhrase> 7
<p> Pass phrase for securing this request. You will need to supply
this value when retrieving your certificate [optfield] <BR>
<INPUT NAME="PassPhrase" TYPE="password" SIZE=32 maxlength="32"> <BR>
<p> Reenter your pass phrase to confirm <BR>
<INPUT NAME="ConfirmPassPhrase" TYPE="password" SIZE=32
 maxlength="32">
</INSERT>
:
<INSERT NAME=-pagefooter>
<p>email: webmaster@your_company.com
</INSERT>
```

The numbers in the following list refer to the highlighted items in the preceding example:

1. The requestok INSERT has the logic to generate the certificate. If the certificate is successfully generated, a Web page (whose main heading is ″Request submitted successfully″) is displayed. This Web page includes the transaction ID.
2. The requestok INSERT includes an ACTION that calls caretrieve.rexx, which allows the user to retrieve the certificate.
3. Alternately, if the request is not successful, the requestbad INSERT gains control.
4. (The caretrieve.rexx CGI displays the RETRIEVECONTENT subsection (see Item 15 on page 86) HTML, which displays a Web page that prompts the user for the transaction ID associated with the certificate request. The user enteres the transaction ID (and any password) and clicks the **Continue** button, which calls cagetcert.rexx.) The cagetcert.rexx CGI calls R_PKIServ for EXPORT of the certificate. If the export is successful, cagetcert.rexx displays the HTML under the RETURNCERT subsection (see Item 18 on page 86).
5. The base64-encoded certificate is displayed on the Web page by using the [base64cert] substitution variable.
6. This is a browser-qualified PublicKey INSERT for Internet Explorer.
7. Additional INSERTs are certificate field name INSERTs. These describe the fields using the HTML dialogs that are displayed on the Web pages if the user is allowed to input these fields. For example, PassPhrase is a text field with a maximum length of 32 characters. The two-year PKI browser certificate for authenticating to z/OS allows the user to fill in this field. (%%PassPhrase%% is listed in the input fields; see Item 9 on page 86.)

Relationship between CGIs and the pkiserv.tmpl file

CGIs are REXX execs that gain control when the end user clicks an action button (for example, the **Request certificate** button on the PKI Services home page. The CGIs read the pkiserv.tmpl file to determine the action to perform. They resolve substitution variables in the pkiserv.tmpl file.

The following are the CGIs for the end-user Web pages (including their directories):

- /usr/lpp/pkiserv/PKIServ/public-cgi/camain.rexx
- /usr/lpp/pkiserv/PKIServ/ssl-cgi-bin/catmpl.rexx
- /usr/lpp/pkiserv/PKIServ/ssl-cgi-bin/auth/careq.rexx
- /usr/lpp/pkiserv/PKIServ/ssl-cgi-bin/caretrieve.rexx
- /usr/lpp/pkiserv/PKIServ/ssl-cgi-bin/auth/cagetcert.rexx

- /usr/lpp/pkiserv/PKIServ/clientauth-cgi-bin/cadisplay.rexx
- /usr/lpp/pkiserv/PKIServ/clientauth-cgi-bin/camodify.rexx

The following table summarizes the actions the CGIs perform:

Table 33. CGI actions for end-user Web pages

REXX exec	Action	Sample associated Web page
camain.rexx	• When user clicks the **Request certificate** button, this calls catmpl.rexx, passing it a parameter identifying the selected template. • The user can click the **Pick up certificate** button to go directly to caretrieve.rexx (if the certificate is already requested). • The user can click the **Renew or revoke certificate** button to go to cadisplay.rexx. • An administrator can click the **Go to administration page** button to go to admmain.rexx. (See Table 35 on page 101 for more information about admmain.rexx.)	See Figure 4 on page 126.
catmpl.rexx	• Displays Web page coded in the HTML under the CONTENT subsection (of a TEMPLATE section) • When the user clicks the **Submit certificate request** button, this passes template and field name parameters to careq.rexx. • When the user clicks the **Retrieve your certificate** button, this passes control to caretrieve.rexx.	See Figure 6 on page 131.
careq.rexx	• Processes field names under the APPL subsection (of a TEMPLATE section) **Note:** Depending on the template, this can be: – UserId only – UserId and HostIdMap • Processes hardcoded field names under the CONSTANT subsection (of a TEMPLATE section) • Depending on the results, displays Web page coded in the HTML under the SUCCESSCONTENT or FAILURECONTENT subsection (of a TEMPLATE section) – The SUCCESSCONTENT subsection includes a **Continue** button the user can click to continue to caretrieve.rexx	See Figure 7 on page 132.
caretrieve.rexx	• Displays Web page coded in the HTML under the RETRIEVECONTENT subsection (of a TEMPLATE section). This HTML prompts the user to enter the transaction ID and a password if the user entered one when requesting the certificate • When the user clicks the **Retrieve and install certificate** button, this passes the transaction ID parameter to cagetcert.rexx.	See Figure 8 on page 133.

Table 33. CGI actions for end-user Web pages (continued)

REXX exec	Action	Sample associated Web page
cagetcert.rexx	• Displays Web page coded in the HTML under RETURNCERT subsection (of a TEMPLATE section). This HTML determines which of the following forms to use when returning the certificate: – as a base64-encoded certificate (for server certificates) – as an ActiveX object (for Microsoft Internet Explorer browser certificates) – as an application/x-x509-user-certificate MIME type (for Netscape browser certificates)	See Figure 9 on page 134.
cadisplay.rexx	• Displays Web page coded in the HTML under the RECONTENT subsection (of the APPLICATION section) • For renewing a certificate, the user fills in the passphrase and clicks the **Renew** button. For revoking a certificate, the user clicks the **Revoke** button. Both actions call camodify.rexx.	See Figure 12 on page 138.
camodify.rexx	• Displays Web page coded in the HTML under the SUCCESSCONTENT subsection (of a TEMPLATE section) for a successful renewal. The SUCCESSCONTENT subsection includes a **Continue** button the user can click to call caretrieve.rexx. • Displays the Web page coded in HTML under the RESUCCESSCONTENT subsection (of the APPLICATION section) for a successful revocation.	See Figure 7 on page 132.

Steps for performing minimal customization

Before you begin: Review the certificate templates and decide if there are any that you want to remove from the pkiserv.tmpl certificates template file. If so, do this first. (To remove a certificate template, you can simply remove its name from the APPLICATION section.)

Perform the following steps to do the minimal updates on the remaining certificate templates:

Note: Fields such as %%Org%%, %%Country%%, and so forth are used to form the subject's distinguished name. Therefore, make sure that the name formed has a suffix that matches a suffix that the LDAP directory supports (that is, that it matches one of the suffix values in the slapd.conf file).

1. For the SAF templates, update the following fields as needed:
   a. If present, replace the OrgUnit values in the following lines with values more appropriate to your organization:

      ```
      %%OrgUnit=Nuts and Bolts Division%%
      %%OrgUnit=SAF template certificate%%
      ```

   b. Replace `taca` in the following line with the correct label of the CERTAUTH signing certificate:

      ```
      %%SignWith=SAF:CERTAUTH/taca%%
      ```

2. For the PKI templates, replace the OrgUnit value in the following line with a value more appropriate for your organization:

```
%%OrgUnit=Class 1 Internet Certificate CA%%
```

3. If present, replace `The Firm` with the name of your company in the following %%Org line:

   ```
   %%Org=The Firm%%
   ```

4. If your company location is not the United States, update the following line by specifying the correct two-letter country abbreviation:

   ```
   %%Country=US%%
   ```

5. If present, replace `host-name` with the domain name of this system in the following %%HostIdMap line:

   ```
   %%HostIdMap=@host-name%%
   ```

   You also need to follow the instructions in “Administering HostIdMappings extensions” on page 168.

6. Insert the copyright statement for your company in the -copyright named field in the INSERT section.

7. Insert the e-mail address of your company’s PKI Services administrator in the -pagefooter named field in the INSERT section.

Customizing the end-user Web pages

If you want to do more than the minimal customization of an end-user Web pages, you must know the location of the code for that Web page.

Table 34. Location of code for various Web pages

Main header (and sample Web page if any)	Location of code in pkiserv.tmpl certificate templates file
″1 Year S/MIME Browser Certificate″	TEMPLATE section, CONTENT subsection
″1 Year SSL Browser Certificate″ (See Figure 6 on page 131.)	TEMPLATE section, CONTENT subsection
″2 Year Browser Certificate For Authenticating To z/OS″	TEMPLATE section, CONTENT subsection
″5 Year PKI Intermediate CA Certificate″	TEMPLATE section, CONTENT subsection
″5 Year PKI IPSEC Server (Firewall) Certificate″	TEMPLATE section, CONTENT subsection
″5 Year PKI SSL Server Certificate″	TEMPLATE section, CONTENT subsection
″Here’s Your Certificate. Cut and Paste it to a File″	INSERT section, -returnpkcs10cert INSERT **Note:** This is referenced in the RETURNCERT subsection of the TEMPLATE section of each certificate template.
″Internet Explorer Certificate Install″ (See Figure 9 on page 134.)	INSERT section, returnbrowsercertIE INSERT

Table 34. Location of code for various Web pages (continued)

Main header (and sample Web page if any)	Location of code in pkiserv.tmpl certificate templates file
″PKISERV Certificate Generation Application″ (See Figure 4 on page 126.)	APPLICATION section, CONTENT subsection
″Renew or Revoke a Browser Certificate″ (See Figure 12 on page 138.)	APPLICATION section, RECONTENT subsection
″Request submitted successfully″ (For submitting a successful certificate request or renewal, see Figure 7 on page 132.)	• For a successful certificate request or renewal: INSERT section, -requestok INSERT **Note:** This is referenced in the SUCCESSCONTENT subsection of the TEMPLATE section of the appropriate certificate template. • For a successful certificate revocation: INSERT section, -renewrevokeok INSERT. **Note:** This is referenced in the RESUCCESSCONTENT subsection of the APPLICATION section.
″Request was not successful″	• For an unsuccessful certificate request: INSERT section, -requestbad INSERT **Note:** This is referenced in the FAILURECONTENT subsection of the TEMPLATE section of each certificate template. • For an unsuccessful certificate revocation request: INSERT section, -renewrevokebad INSERT **Note:** This is referenced in the REFAILURECONTENT subsection of the APPLICATION section.
″Retrieve Your 1 Year S/MIME Browser Certificate″	TEMPLATE section, RETRIEVECONTENT subsection
″Retrieve Your 1 Year SSL Browser Certificate″ (See Figure 8 on page 133.)	TEMPLATE section, RETRIEVECONTENT subsection
″Retrieve Your 2 Year Browser Certificate For Authenticating To z/OS″	TEMPLATE section, RETRIEVECONTENT subsection
″Retrieve Your 5 Year PKI Intermediate CA Certificate″	TEMPLATE section RETRIEVECONTENT subsection
″Retrieve Your 5 Year PKI IPSEC Server (Firewall) Certificate″	TEMPLATE section, RETRIEVECONTENT subsection
″Retrieve Your 5 Year PKI SSL Server Certificate″	TEMPLATE section, RETRIEVECONTENT subsection
″Retrieve Your SAF Browser Certificate 1 Year″	TEMPLATE section, RETRIEVECONTENT subsection
″Retrieve Your SAF Server Certificate 1 Year″	TEMPLATE section, RETRIEVECONTENT subsection
″SAF Browser Certificate 1 Year (Auto Approved)″	TEMPLATE section, CONTENT subsection
″SAF Server Certificate 1 Year (Auto Approved)″	TEMPLATE section, CONTENT subsection

Note: Fields (such as the Key Usage (KeyUsage) drop down or the Organizational Unit (OrgUnit) text field) are defined in the pkiserv.tmpl certificate templates file, in the INSERT section. (See Table 27 on page 68 for descriptions of the fields.)

Steps for customizing the certificate templates

Perform the following steps to customize the end-user Web pages:

1. Review the templates and decide which one(s) you need to update.

---

2. If necessary, change the true name, alias, or nickname, as in the following lines.

```
<TEMPLATE NAME=true_name>
<TEMPLATE NAME=alias>
<NICKNAME=nickname>
```

true_name
Is the whole and complete name of the certificate template.

alias
Differentiates browser from server certificates. An alias is not required. You can have more than one alias.

nickname
Is an 8-character name. SAF certificates do not have nicknames. If a nickname is not present, the certificate is not renewable.

Example:

```
<TEMPLATENAME=1 Year PKI SSL Browser Certificate>
<TEMPLATENAME= PKI Browser Certificate>
<NICKNAME=1YBSSL>
```

---

3. If necessary, in the CONTENT subsection, change the certificate fields listed. The following example is from the one-year PKI SSL browser certificate template.

Example:

```
<p> Enter values for the following field(s)
 %%CommonName%%
 %%Requestor (optional)%%
 %%PassPhrase%%
 %%PublicKey2[browsertype]%%
```

---

4. If you add required fields in the preceding step, update the JavaScript code that is part of the embedded HTML in the to check for required fields that are missing.

---

5. If necessary, in the APPL subsection, change the list of certificate fields that the application provides. (Currently, the only supported fields are UserId and HostIdMap.) The following example is from the two-year PKI browser certificate for authenticating to z/OS:

Example:

```
<APPL>
 %%UserId%%
 %%HostIdMap=@host-name%%
<APPL>
```

---

6. If necessary, in the CONSTANT subsection, update the list of certificate fields whose values are hardcoded. The following example is from the one-year PKI SSL browser certificate template:

   Example:

   ```
   <CONSTANT>
    %%NotBefore=0%%
    %%NotAfter=365%%
    %%KeyUsage=handshake%%
    %%OrgUnit=Class 1 Internet Certificate CA%%
    %%Org=The Firm%%
    %%SignWith=PKI:%%
   <CONSTANT>
   ```

   Note: If you update the CONSTANT subsection to create subject distinguished names, make sure that the names match the LDAP suffix defined for your LDAP server. Otherwise the certificates are not posted to LDAP. PKI Services constructs the subject distinguished name from the fields specified in the following order:

   - CommonName
   - Title
   - OrgUnits (in the order that they appear in the template file)
   - Org
   - Locality
   - StateProv
   - Country

---

7. If necessary, edit the ADMINAPPROVE subsection. (Certificates requiring an administrator's approval have an ADMINAPPROVE subsection. The absence of the ADMINAPPROVE subsection indicates auto-approval for requests.) Make sure the ADMINAPPROVE subsection, if present, correctly lists the minimum set of certificate fields that the administrator can change.

   Notes:

   a. There may be more fields in the ADMINAPPROVE subsection than fields that the user can complete in the certificate request (because the users do not necessarily see all fields).

   b. Do not include the Requestor, Label, UserId, PublicKey, or SignWith fields in the ADMINAPPROVE subsection; these fields cannot be changed and are ignored if present. (See page 76 for a list of fields that can be in the ADMINAPPROVE subsection.)

   The following example of the ADMINAPPROVE subsection is from the one-year PKI SSL browser certificate template:

   Example:

   ```
   <ADMINAPPROVE>
    %%CommonName (Optional)%%
    %%OrgUnit (Optional)%%
   ```

```
 %%OrgUnit (Optional)%%
 %%Org (Optional)%%
 %%NotBefore (optional)%%
 %%NotAfter (Optional)%%
 %%KeyUsage (Optional)%%
 %%HostIdMap (Optional)%%
 %%HostIdMap (Optional)%%
 %%HostIdMap (Optional)%%
 %%HostIdMap (Optional)%%
<ADMINAPPROVE>
```

Note: The four %%HostIdMap%% lines in the example indicate that the approver can provide up to four HostIdMap entries.

---

8. If necessary, update the following:
   - The SUCCESSCONTENT subsection contains only the %%-requestok%% named field, which contains the HTML for the Web page whose main heading is ″Request submitted successfully.″ To make changes to this Web page, update the requestok INSERT (in the INSERT section of pkiserv.tmpl):

     ```
     <INSERT NAME=-requestok>
     <HTML><HEAD>
     <TITLE> Web Based Certificate Generation Success</TITLE>
     </HEAD>
     <BODY>
     <H1> Request submitted Successfully</H1>
     [errorinfo]
     <p> Here's your transaction ID. You will need it to retrieve your
     certificate.  Press 'Continue' to retrieve the certificate.
     <p> <TABLE BORDER><TR><TD>[transactionid]</TD></TR></TABLE>
     <FORM METHOD=GET ACTION="/PKIServ/ssl-cgi/caretrieve.rexx">
     <INPUT NAME="Template" TYPE="hidden" VALUE="[tmplname]">
     <INPUT NAME="TransactionId" TYPE="hidden" VALUE="[transactionid]">
     <INPUT TYPE="submit" VALUE="Continue">
     </FORM>
     <p>%%-pagefooter%%
     </BODY>
     </HTML>
     </INSERT>
     ```

   - The FAILURECONTENT subsection contains only the %%-requestbad%% named field, which contains the HTML for the Web page whose main heading is ″Request was not successful.″ To make changes to this Web page, update the requestbad INSERT:

     ```
     <INSERT NAME=-requestbad>
     <HTML><HEAD>
     <TITLE> Web Based Certificate Generation Failure</TITLE>
     </HEAD>
     <BODY>
     <H1> Request was not successful</H1>
     <p> Please correct the problem or report the error to your Web admin
     person<br>
     <PRE>
     [errorinfo]
     </PRE>
     <p>%%-pagefooter%%
     </BODY>
     </HTML>
     </INSERT>
     ```

---

9. If necessary, update the RETRIEVECONTENT subsection.

Note: See “Steps for changing the runtime user ID for retrieving certificates” on page 98 for directions for changing the runtime user ID for retrieving a certificate.

a. The RETRIEVECONTENT subsection includes the %%-copyright%% named field. If you want to make any changes in the copyright statement, update the copyright INSERT. (The following is the copyright INSERT as it is originally provided in the pkiserv.tmpl file. You should have previously updated this INSERT by providing information tailored to your company, as described in “Steps for performing minimal customization” on page 90.)

```
<INSERT NAME=-copyright>
<!--
/*******************************************************************/
/*                                                                 */
/* LICENSED MATERIALS - PROPERTY OF IBM                            */
/* THIS SCRIPT IS "RESTRICTED MATERIALS OF IBM"                    */
/* 5647-A01 (C) COPYRIGHT IBM CORP. 2000,2001                      */
/*                                                                 */
/*******************************************************************/
-->
</INSERT>
```

b. If necessary, update any desired Web page content (such as headers, footers, titles, background colors, frames, links, and so on) for the Web page whose main heading is ″Retrieve Your (certificate template name).″

---

10. If you are updating the template for a server certificate, you can update the HTML in the RETURNCERT subsection to customize the returned Web page. (For a browser template, you cannot change the RETURNCERT subsection. It must contain the %%returnbrowsercert%% named field, which contains the [browsertype] substitution variable. The INSERT section contains browser-specific returnbrowsercert INSERTs.)

Steps for adding a new certificate template

Perform the following steps to add a new certificate template:

1. Review the contents of the eight certificate templates provided with PKI Services to determine the one that most closely approximates the certificate template you want to add.

---

2. After you have determined the certificate template to use as a model, copy this section in the certificate templates file.

---

3. Provide a new name, alias, and, if present, nickname for the certificate template.

---

4. Follow the remaining steps, starting at Step 3 on page 93 in the preceding section.

Changing the runtime user ID

When the PKI Services CGIs are called, they are assigned a runtime user ID. This is the identity that is associated with the unit of work (task). This identity must be authorized to call the function being requested. (See Chapter 15, “RACF administration for PKI Services” on page 167 for more information.) Most of the templates run under the surrogate user ID (PKISERV) for requesting a certificate and for subsequently retrieving it.

There are two exceptions:

- The two SAF templates run under PKISERV for requesting a certificate but run under the client’s user ID for certificate retrieval.
- The five-year PKI intermediate CA template runs under the client’s user ID for requesting a certificate and for certificate retrieval.

The advantage of having PKISERV as the runtime user ID is that this is the only user ID that needs to be authorized for requesting certificates. The advantage of using the client’s user ID is that you have greater control over who can request and retrieve certificates. For example, you can require the user to authenticate by entering user ID and password before requesting or retrieving a certificate.

You can control the user ID under which a certificate request or retrieval runs by selectively commenting and uncommenting FORM statements in the pkiserv.tmpl file. (For requesting a certificate, the FORM statements are in the appropriate TEMPLATE section, in the CONTENT subsection. For retrieving a certificate, the FORM statements are in the appropriate TEMPLATE section, in the RETRIEVECONTENT subsection.)

There are three levels of access control for requesting and retrieving certificates:

- Under the client’s ID with user ID and password authentication
- Under the surrogate user ID with user ID and password authentication
- Under the surrogate user ID without user ID and password authentication.

Protection directives in the z/OS HTTP Server’s configuration file (which defaults to /etc/httpd.conf) enforce these three levels of access control. The default configuration for PKI Services maps the three levels of access control to the following CGI directories respectively:

- /PKIServ/ssl-cgi-bin/auth
- /PKIServ/ssl-cgi-bin/surrogateauth
- /PKIServ/ssl-cgi-bin

Each of the request and retrieve CGIs reside in all three directories. Thus, when you run a CGI you get the protection established for the directory from which it is called.

Each certificate template contains several FORM statements (two commented out and one uncommented, which is active) that determines which of these applies. You can change the access control by uncommenting one of the FORM statements that is commented out and commenting out the one that is active.

Steps for changing the runtime user ID for requesting certificates

Perform the following steps to change the runtime user ID for requesting a certificate.

1. In the pkiserv.tmpl file, find the CONTENT subsection of the TEMPLATE section for the template whose user ID you want to change. Locate the lines containing the FORM statements, such as those in the following example:

   Example:

```
<h3><li>Request a New Certificate
# This ACTION forces userid/pw authentication and runs the task under
# the client's ID
#<FORM NAME="CertReq" METHOD=POST ACTION=
#                "/PKIServ/ssl-cgi-bin/auth/careq.rexx" onSubmit=

# This ACTION forces userid/pw authentication but runs the task under
# the surrogate ID
#<FORM NAME="CertReq" METHOD=POST ACTION=
#              "/PKIServ/ssl-cgi-bin/surrogateauth/careq.rexx" onSubmit=

# This ACTION is for non z/OS clients. The task runs under the
# surrogate ID
 <FORM NAME="CertReq" METHOD=POST ACTION=
               "/PKIServ/ssl-cgi-bin/careq.rexx" onSubmit=
```

   Notice that the preceding lines contain three FORM statements. The first two FORM statements are commented out, so they are not active. They are for:

   - Requesting the certificate under the client's ID and using user ID and password authentication
   - Requesting the certificate under the surrogate ID and using user ID and password authentication

   The third FORM statement is for requesting the certificate under the surrogate user ID without user ID and password authentication. This is active (it is not commented out).

2. To change the runtime user ID, remove the comment delimiter (#) from in front of the lines for the commented-out FORM statement you want to use and insert the comment delimiter in front of the lines for the bottom FORM statement.

Steps for changing the runtime user ID for retrieving certificates

Perform the following steps to change the runtime user ID for retrieving a certificate.

1. In the pkiserv.tmpl file, find the RETRIEVECONTENT subsection of the TEMPLATE section for the template whose user ID you want to change. Locate the lines containing the FORM statements, such as those in the following example:

   Example:

```
<H1> Retrieve Your [tmplname]
<H3>Please bookmark this page
<p>Since your certificate may not have been issued yet, we recommend
that you create a bookmark to this location so that when you return to
this bookmark, the browser will display your transaction ID.
This is the easiest way to check your status.

# This ACTION forces userid/pw authentication and runs the task
# under the client's ID
#<FORM NAME=retrieveform METHOD=POST ACTION=
#      "/PKIServ/ssl-cgi-bin/auth/cagetcert.rexx" onSubmit=
#
# This ACTION forces userid/pw authentication but runs the task
# under the surrogate ID
#<FORM NAME=retrieveform METHOD=POST ACTION=
```

```
#      "/PKIServ/ssl-cgi-bin/surrogateauth/cagetcert.rexx" onSubmit=
#
# This ACTION is for non z/OS clients. The task runs under surrogate ID
 <FORM NAME=retrieveform METHOD=POST ACTION=
       "/PKIServ/ssl-cgi-bin/cagetcert.rexx" onSubmit=
```

Notice that the preceding lines contain three FORM statements. The first two FORM statements are commented out (they are not active). These are for:

- Retrieving the certificate under the client's ID
- Retrieving it under the surrogate ID, but requiring user ID and password authentication.

The third FORM statement is for retrieving the certificate under the surrogate user ID without user ID and password authentication. This is active (it is not commented out).

---

2. To change the runtime user ID, remove the comment delimiter (#) from in front of the lines for the commented-out FORM statement you want to use and insert the comment delimiter in front of the lines for the bottom FORM statement.

Customizing the end-user Web pages

Chapter 11. Customizing the administration Web pages

CGIs for administration Web pages

CGIs are REXX execs that gain control when the user clicks an action button. The administrative CGIs are connector REXX execs that render Web pages dynamically.

All of the administrative CGIs are contained in the usr/lpp/pkiserv/PKIServ/ssl-cgi-bin/auth/directory.

The following table (which lists the REXX execs in logical order) summarizes the actions the CGIs perform:

Table 35. CGI actions for administrative Web pages

REXX exec	Action	Sample Web page
admmain.rexx	This displays the administration home page. The main heading is ″PKI Services Administration.″ This Web page lets the administrator work with a single certificate request or certificate or search for certificate requests or certificates.	See Figure 16 on page 147.
admpend.rexx	On the administration home page, the administrator can search for certificate requests. This displays a Web page whose main heading is one of the following: • ″Certificate Requests″ Web page — This lists certificate requests matching the criteria and allows the administrator to process the certificate request(s). • ″Processing was not successful″ Web page	For an example of the ″Certificate Requests″ Web page, see Figure 20 on page 153.
admpendtid.rexx	On the administration home page, the administrator can enter a transaction ID to work with a single certificate request. This displays a Web page whose main heading is one of the following: • ″Single Request″— This lists the certificate request that matches the transaction ID and allows the administrator to process that certificate request. • Processing was not successful″	For an example of the ″Single Request″ Web page, see Figure 17 on page 148.
admmodtid.rexx	This displays the ″Modify and Approve Request″ Web page that appears when the administrator decides to modify a request before approving it (on the ″Single Request″ Web page).	See Figure 19 on page 150.
admicl.rexx	On the administration home page, the administrator can search for certificates. This displays a Web page whose main heading is one of the following: • ″Issued Certificates″ — This lists the certificate(s) that match the search criteria and allows the administrator to revoke or delete selected certificate(s). • ″Processing was not successful″	For a sample of the ″Issued Certificates″ Web page, see Figure 20 on page 153.
admiclcert.rexx	On the administration home page, the administrator can enter a serial number to work with a single certificate. This displays a Web page whose main heading is one of the following: • ″Single Issued Certificate″ — This lists the certificate that matches the serial number ID and allows the administrator to revoke or delete that certificate. • ″Processing was not successful″	For a sample of the ″Single Issued Certificate″ Web page, see Figure 24 on page 158.

Table 35. CGI actions for administrative Web pages (continued)

REXX exec	Action	Sample Web page
admacttid.rexx	Displays a Web page after the administrator processes a single certificate request (approving it with or without modifications, rejecting, or deleting it). This Web page has one of the following as its main heading: • ″Processing successful″ • ″Processing was not successful″	For a sample of the Web page whose main heading is ″Processing successful″ see Figure 18 on page 149.
admacttid2.rexx	This displays a Web page after the administrator approves a certificate request with modifications. The Web page has one of the following main headings: • ″Processing successful″ • ″Processing was not successful″	For a sample of the Web page whose main heading is ″Processing successful″ see Figure 18 on page 149.
admpendall.rexx	After the administrator searches for certificate requests and admpend.rexx displays the results, the administrator clicks a button to approve, reject, or delete selected certificate requests. This calls admpendall.rexx, whose main heading is one of the following: • ″Processing successful″ if the action was successful • ″Processing was not successful″ if the action failed (for example, if the administrator tried to delete certificate requests that were already deleted″ • Processing partially successful″ if not all of the selected requests are processed successfully	• For an example of the ″Processing successful″ Web page, see Figure 21 on page 155. • For an example of the ″Processing was not successful″ Web page, see Figure 22 on page 155. • For an example of the ″Processing partially successful″ Web page, see Figure 23 on page 156.
admactcert.rexx	Displays a Web page after the administrator tries to revoke or delete one or more selected certificates. The Web page has one of the following main headings: • ″Processing successful″ • ″Processing was not successful″	None
admiclall.rexx	After the administrator searches for certificates and admicl.rexx displays the results, the administrator clicks a button to revoke or delete selected certificates. This calls admiclall.rexx, which displays a Web page whose main heading is one of the following: • ″Processing successful″ if the action was successful • ″Processing was not successful″ if the action failed • Processing partially successful″ if not all of the selected certificates are processed successfully	None

Customizing the administration Web pages

The administration Web pages are not as customizable as the end-user Web pages. You can customize page headers, footers, frames, links, colors, and so forth, but you cannot change internal Web page content. Except for identifying the fields that an administrator can change when approving certificate requests, the administration Web page logic is fixed.

However, you can make changes in the following two subsections in the APPLICATION section of the pkiserv.tmpl certificate template file:

ADMINHEADER
Contains the general installation-specific HTML content for the header of all the administration Web pages.

ADMINFOOTER
Contains the general installation-specific HTML content for the footer of all the administration pages.

Steps for customizing the administration Web pages

Perform the following steps to customize the administration Web pages:

1. Add any desired Web page header for the administration pages to the ADMINHEADER subsection of the PKISERV APPLICATION section. (The ADMINHEADER subsection is near the end of the APPLICATION section.)

   Example:

   ```
   <ADMINHEADER>
   <HTML>h<HEAD>
   <TITLE>Web-Based Certificate Generation Administration</TITLE></HEAD>
   <BODY>
   </ADMINHEADER>
   ```

2. Add any desired Web page footer for the administration pages to the ADMINFOOTER subsection of the APPLICATION section. (The ADMINFOOTER subsection is near the end of the APPLICATION section.)

   Example:

   ```
   <ADMINFOOTER>
   <p> email: webmaster@company.com
   </BODY>
   </HTML>
   </ADMINFOOTER>
   ```

Changing the runtime behavior for accessing administration pages

When the administrator tries to access the administration pages (by clicking the **Go to administration page** button on the PKI Services home page), access to the administration pages is controlled in one of the following ways:

- A popup window appears, requiring the administrator to enter a user name and password. (See Figure 15 on page 144 for a sample of the authentication popup window.)
- Alternately, the administrator may have to authenticate by using a previously issued browser certificate. In other words, the administrator would need to have a certificate before visiting the administration Web pages.

By default, the first method is used. However, you can change the runtime behavior so that the second method is used instead. If you decide to use the second method, anyone intending to become a PKI Services administrator needs to request and retrieve a one-year PKI browser certificate for authenticating to z/OS before trying to access the administration pages.

Note: The one-year PKI browser certificate for authenticating to z/OS contains a HostIdMappings extension. (For more information, see Chapter 15, “RACF administration for PKI Services” on page 167.)

Steps for changing control of access to administration pages

Perform the following steps to change the access control of the administration pages to require authenticating by using a certificate:

1. Edit the pkiserv.tmpl certificate templates file and find the following lines in the APPLICATION section:

   ```
   # The following action will force userid/pw authentication for administrators
   <FORM name=admform METHOD=GET ACTION="/PKIServ/ssl-cgi/auth/admmain.rexx">
   # The following action will force client certificate authentication
   # for administrators
   #<FORM name=admform METHOD=GET
   # ACTION="/PKIServ/clientauth-cgi/auth/admmain.rexx">
   <p>
   <INPUT TYPE="submit" VALUE="Go to Admin Pages">
   </FORM>
   ```

   The first FORM statement in these lines is active (it is not commented out with # characters in front of the lines). This requires authentication by entering the user name and password in a popup window. The second FORM statement is commented out (using # characters). This requires authentication by using a previously issued browser certificate.

   ---

2. Comment out the first FORM statement (add # characters in front of the FORM and ACTION lines) and uncomment the second FORM statement (removing the # characters in front of the FORM and ACTION lines).

(Optional) Steps for removing the administration page link from the PKI Services home page

Optionally remove the administration page link from the PKI Services home page and provide an alternative way for administrators to access the administration home page.

Recommendation: Do these steps before going into production mode for added security. It will prevent your general end-user population from trying to access the administration pages if they are not authorized to do so.

To remove the administration page link and provide an alternative way to access the administration pages, perform the following steps:

1. If you want to require your administrators to authenticate using a certificate, make sure you have performed the steps in “Steps for changing control of access to administration pages” and that you have tested to ensure your changes work.

   ---

2. Edit the pkiserv.tmpl file and remove the entire HTML `<FORM>...</FORM>` in “Steps for changing control of access to administration pages”. (These are the very same lines you edited in that section.)

   ---

3. Make the administration URL available to your administrators through an alternate means. (For example, you can use a link in another internal web page or in an e-mail message, and so on.)

In the following URLs, *webserver-fully-qualified-domain-name* is the common name (CN) portion of the public webserver's distinguished name; see Table 11 on page 25. Web server redirection ensures that an SSL connection is established.

- If you are allowing your administrators to authenticate by specifying a user ID and password, the URL is:

  `http://webserver-fully-qualified-domain-name/PKIServ/ssl-cgi/auth/admmain.rexx`

- If you want your administrators to authenticate using a client certificate, the URL is:

  `http://webserver-fully-qualified-domain-name/PKIServ/clientauth-cgi/auth/admmain.rexx`

Note: Your administrators still need to visit the PKI Services home page to install the CA certificate into their browsers. See "Steps for accessing the administration home page" on page 141 for more information.

Customizing the administration Web pages

Chapter 12. Advanced customization

This chapter describes advanced customization methods that PKI Services provides, including:

- Using certificate policies
- Updating the signature algorithm
- Using the PKI exit.

Using certificate policies

Certificates can contain a CertificatePolicies extension. This extension contains policy information, such as the way in which your CA operates and the intended purpose of the issued certificates. (For more information about this extension, see the Internet Engineering Task Force (IETF) Web site (www.ietf.org) for RFC2459.)

By default, PKI Services does not include this extension in the certificates it creates. However, you can define your own CertificatePolicies extension by modifying fields in the CertPolicy section of the pkiserv.conf configuration file. The CertificatePolicies extension contains one or more PolicyInformation sequences. (Typical usage has just one of these.) The PolicyInformation sequence has the following format:

- Your Policy OID as registered with the appropriate standards organization (ISO or ITU)
- Zero or more PolicyQualifiers sequences, each having the following information:
  - Either a Certificate Practices Statement (CPS) URI
  - Or a UserNotice sequence, which consists of one or both of the following:
    - A notice (text string) intended to be viewed by customers using the certificate such as copyright or other legal information
    - Your organization's legal name (text string) with one or more notice numbers defined elsewhere, perhaps in your CPS

Unlike other extensions, which you can define on a per certificate template basis, PKI Services supports the CertificatePolicies extension only on a global basis. Either all the certificates PKI Services creates have the same CertificatePolicies extension or none of them have it.

Steps for creating the CertificatePolicies extension

Perform the following steps to create your own CertificatePolicies extension:

1. Edit the pkiserv.conf configuration file and find the CertPolicy section.

   _____

2. Change the value of PolicyRequired to T (True) as in the following line:

   ```
   PolicyRequired=T
   ```

   _____

3. If you want to have the extension marked critical (this is not recommended), set the PolicyCritical equal to T (True) as in the following line:

   ```
   PolicyCritical=T
   ```

   _____

4. Go to the OIDs section of the pkiserv.conf configuration file. By default (as shown in the following example), the value of MyPolicy is 1.2.3.4. The value of

MyPolicy should be a customer-specific (registered) Object ID identifying your organization's certificate. Replace the value of MyPolicy in the following line with your Object ID. Make a note of the value (you need it for the next step).

Example:

```
[OIDs]
MyPolicy=1.2.3.4
```

---

5. Go back to the CertPolicy section and update the PolicyName1 line to change *MyPolicy* to the value of MyPolicy in the OIDs section:

   ```
   [CertPolicy]
   PolicyName1=MyPolicy
   ```

---

6. If you want to add qualifiers, perform the following steps:
   a. Update the Policy1Org and Policy1Notice*n* fields in the following example:

      ```
      Policy1Org=My Company,Inc
      Policy1Notice1=1
      ```

      Policy1Org Your organization's name, for example, `International Business Machines, Inc.`

      Policy1Notice1 through Policy1Notice*n*
      Your notice numbers. (You may need more than one Policy1Notice*n* line, depending on how many notice numbers you have. Repeat the line as needed, by incrementing the suffix number on the keyword, for example Policy1Notice1, Policy1Notice2, and so forth.)

   b. Change the value of the UserNoticeText1 line shown in the following. The *statement* should be your notice text string, for example, `Certificate for IBM internal use only`.

      ```
      UserNoticeText1=statement
      ```

   c. Change the value of the CPS1 line shown in the following. The value should be your CPS URI, for example, `http://www.ibm.com/cps.html`.

      ```
      CPS1=http://www.mycompany.com/cps.html
      ```

   If you do not want to add qualifiers, delete or comment out (by inserting a # character at the start of the line) the preceding lines.

---

7. If you need multiple qualifiers, repeat the following fields as needed, incrementing the suffix numbers, for example:

   ```
   PolicyName2=MyOtherPolicy
   Policy2Org=International Business Machines, Inc.
   Policy2Notice1=5
   Policy2Notice2=9
   UserNoticeText2=Certificate is intended for testing only
   CPS2=http://www.ibm.com/cps2.html
   ```

Updating the signature algorithm

By default, PKI Services uses the SHA-1 with RSA encryption signature algorithm for signing certificates and CRLs. If you need to use one of the older RSA algorithms, you can change the SigAlg1 value in the CertPolicy section of the pkiserv.conf configuration file. The signature algorithm must be one of the following:

- sha-1WithRSAEncryption=1.2.840.113549.1.1.5
- md-5WithRSAEncryption=1.2.840.113549.1.1.4
- md-2WithRSAEncryption=1.2.840.113549.1.1.2

Steps for changing the signature algorithm

Perform the following steps to change the signature algorithm:

1. Edit the pkiserv.conf configuration file and find the OIDs section.

   ---

2. If you want to change from SHA-1 encryption to MD-5, add the following line:

   ```
   md-5WithRSAEncryption=1.2.840.113549.1.1.4
   ```

   Otherwise, to change to MD-2, add the following line:

   ```
   md-2WithRSAEncryption=1.2.840.113549.1.1.2
   ```

   ---

3. Find the CertPolicy section.

   ---

4. If you want to change from SHA-1 encryption to MD-5, change `sha-1WithRSAEncryption` in the following line to `md-5WithRSAEncryption`. If you want to change to MD-2, change `sha-1WithRSAEncryption` to `md-2WithRSAEncryption`.

   ```
   SigAlg1=sha-1WithRSAEncryption
   ```

Using the PKI exit

Programming Interface information

For the end-user functions except VERIFY, the PKISERV Web application CGIs support calling an installation-provided exit routine. The exit routine can perform tasks such as the following:

- Provide additional authorization checking
- Validate and change parameters
- Capture certificates for further processing.

PKI Services provides the following files for the exit. Both files are, by default, located in: /usr/lpp/pkiserv/samples/.

Table 36. Summary of information about important files for the exit routine

File name	Description
pkiexit.c	Code sample for the exit (in the C programming language). You probably need to update the exit code before using it.
Makefile.pkiexit	Makefile for pkiexit.c.

If the exit exists, it must be a UNIX executable residing in the HFS, and it must have appropriate permission assigned. To specify the exit, the UNIX programmer sets the _PKISERV_EXIT environment variable (see page 267). On input it receives standard UNIX parameters (that is, *argc* and *argv*[]). It communicates back to PKISERV through the return code and by writing to STDOUT.

Steps for updating the exit code sample

To update the exit code sample, pkiexit.c, perform the following steps:

1. Copy the sample exit and makefile to the current directory by entering the following commands:

   ```
   cp /usr/lpp/pkiserv/samples/pkiexit.c pkiexit.c
   cp /usr/lpp/pkiserv/samples/Makefile.pkiexit Makefile
   ```

   ---

2. Compile and link to produce the executable, pkiexit, by entering the following command:

   ```
   make
   ```

   ---

3. Move the executable to its execution directory and set the permissions by entering the following commands:

   ```
   mv pkiexit /full-directory-name
   chmod 755 /full-directory-name/pkiexit
   ```

   ---

4. Edit the Web server's environment variables file by entering the following command:

   ```
   OEDIT /etc/httpd.envvars
   ```

   and add the environment variable _PKISERV_EXIT by adding the following line to the file:

   _PKISERV_EXIT=/*full-directory-name*/pkiexit

Using the exit for pre- and post-processing

The exit is called:

- For preprocessing before calling the IRRSPX00 SAF callable service
- For post-processing after returning from the callable service.

The following table summarizes the values of the first two arguments for pre- and post-processing. (Additional arguments vary, depending on the function to perform.)

Table 37. Values of arguments for pre- and post-processing

Time of processing	Argument 1	Argument 2
Preprocessing	0	The function number from the SAF callable service in EBCDIC: **1** GENCERT **2** EXPORT **9** REQCERT **11** REVOKE **12** GENRENEW **13** REQRENEW
Post-processing	1	

Return Codes

The sections that follow contain tables of expected return codes. If calling the exit produces an unexpected return code, that is, one that is not listed, PKI Services treats it as a failure. Processing for the request stops and an error message is issued.

GENCERT and GENRENEW - preprocessing

Purpose: Provide additional authorization checking and parameter validation and modification.

Arguments:

argument **3...***argument n*
The parameters as input to the CGI plus values resolved by the CGI in *name*=*value* form, for example, `"CommonName=Sam Smith"`.

Return Codes:

Return Code	Meaning
0	Continue with the request with possible modifications.
4	Continue with the request with possible modifications, but change it to require administrator approval.
>=8 <50	Deny the request and return to the caller immediately.

STDOUT: Zero or more additional *CertPlist* parameters to add to the request in *name*=*value* form, one per line. For those fields defined as non-repeating (according to the documentation for the IRRSPX00 callable service, for example, *CommonName*), specifying the parameters here in effect replaces the CGI input values.

GENCERT and GENRENEW - post-processing

Purpose: Capture the TransactionId or failing return codes for further processing.

Arguments:

argument **3...***argument n***-3**
The final set of parameters as determined by the preprocessing exit in *name*=*value* form.

*argument n***-2**
The RACF return code from the callable service.

*argument n***-1**
The RACF reason code from the callable service.

argument n
The *TransactionId*. This is a string of undetermined value if the request was unsuccessful.

Return Codes:

Return Code	Meaning
0	Normal

STDOUT: Optional replacement *TransactionId*.

REQCERT and REQRENEW - preprocessing

Purpose: Provide additional authorization checking and parameter validation and modification.

Arguments:

argument **3...***argument n*
The parameters as input to the CGI plus values resolved by the CGI in *name*=*value* form, for example, `"CommonName=Sam Smith"`.

Return Codes:

Return Code	Meaning
0	Continue with the request with possible modifications.
4	Continue with the request with possible modifications, but change it to not require administrator approval.
>=8 <50	Deny the request and return to the caller immediately.

STDOUT: Zero or more additional *CertPlist* parameters to add to the request in *name*=*value* form, one per line. For those fields defined as non-repeating (according to the documentation for the IRRSPX00 callable service, for example, *CommonName*), specifying the parameters here in effect replaces the CGI input values.

REQCERT and REQRENEW - post-processing

Purpose: Capture the *TransactionId* or failing return codes for further processing.

Arguments:

argument **3...***argument n***-3**
The final set of parameters as determined by the preprocessing exit in *name*=*value* form.

*argument n***-2**
The RACF return code from the callable service.

*argument n***-1**
The RACF reason code from the callable service.

argument n
The *TransactionId*. This is a string of undetermined value if the request was unsuccessful.

Return Codes:

Return Code	Meaning
0	Normal

STDOUT: Optional replacement *TransactionId*.

EXPORT - preprocessing

Purpose: Provide additional authorization checking and parameter validation and modification.

Arguments:

argument **3...***argument n*
The parameters as input to the CGI in *name*=*value* form, for example, ″TransactionId=12345″.

Return Codes:

Return Code	Meaning
0	Continue with the export.
>=8 <50	Deny the request and return to the caller immediately.

STDOUT: Optional replacement *TransactionId* and *ChallengePassPhrase* parameters in *name*=*value* form, one per line. If these values are provided, they replace the user-provided values on the call to the SAF callable service. If TransactionId is specified without *ChallengePassPhrase*, the user-provided *ChallengePassPhrase* is used. If *ChallengePassPhrase* is specified without *TransactionId*, the user-provided *TransactionId* is used.

EXPORT - post-processing

Purpose: Capture the certificate or failing return codes for further processing.

Arguments:

argument **3...***argument n***-3**
The parameters as input to the CGI in *name*=*value* form, followed by any modified value provided by the preprocessing exit, also in *name*=*value* form.

*argument n***-2**
The RACF return code from the callable service.

*argument n***-1**
The RACF reason code from the callable service.

argument n
The base64–encoded certificate with header and footer. This is a string of undetermined value if the request was unsuccessful.

Return Codes:

Return Code	Meaning
0	Normal

STDOUT: Non-applicable.

REVOKE - preprocessing

Purpose: Provide additional authorization checking and parameter validation.

Arguments:

argument **3...***argument n*
The parameters as input to the CGI in *name*=*value* form, for example, ″reason=1″.

Return Codes:

Return Code	Meaning
0	Continue with the request.
>=8 <50	Deny the request and return to the caller immediately.

STDOUT: Non-applicable.

REVOKE - post-processing

Purpose: Capture the certificate or failing return codes or both for further processing.

Arguments:

argument **3...***argument n-2*
The parameters as input to the CGI in *name*=*value* form, for example, ″reason=1″.

*argument n***-1**
The RACF return code from the callable service.

argument n
The RACF reason code from the callable service.

Return Codes:

Return Code	Meaning
0	Normal

STDOUT: Non-applicable.

Scenarios for using the PKI exit

The sample PKI exit supplied with PKI Services, pkiexit.c, written in the C language. It is intended to demonstrate the power of the exit and to provide a guide for you to write your own exit. The main routine of the program determines which subroutine to call, based on the R_PKIServ function being called and whether this is a pre- or post-processing call. The individual subroutines in the program handle the following scenarios:

Scenario 1: Allow only selected users to request PKI browser certificates for authenticating to z/OS

This scenario is for allowing only selected local z/OS users to request PKI browser certificates for authenticating to z/OS. Additionally, this is for providing a customized TITLE value for the subject's distinguished name based on the user's role in the organization. Permission and the user's role in the organization is indicated by the user's level of access to RACF FACILITY Class resources called PROJ.MEMBER and PROJ.PARTNER. The access values are as follows:

NONE No access for either resource. The user is not permitted to request this type of certificate. The certificate request is denied.

READ to PROJ.MEMBER
The user is a team member and is permitted to request the certificate. The TITLE value is set to ″Team Member.″ Certificate requests for team members are automatically approved. (No administrator approval is required.)

UPDATE to PROJ.MEMBER
The user is the team's leader and is permitted to request the certificate. The TITLE value is set to ″Team Leader.″ A certificate request by the team leader is automatically approved. (No administrator approval is required.)

READ to PROJ.PARTNER
The user is considered to be a general partner of the team, not an active team member. The user is allowed to request certificates, but the requests require administrator approval before being issued. The TITLE value is set to ″Team Partner.″

UPDATE to PROJ.PARTNER
The user is considered to be a trusted partner of the team, not an active team member. The user is allowed to request certificates, and unlike requests of the general partner, the certificate request are automatically approved. The TITLE value is set to ″Team Trusted Partner.″

The preprocessing exit call for the GENCERT and REQCERT functions (subroutine preProcessGenReqCertExit) handles the logic described in the preceding. Here are the steps:

- The request values are passed into the exit through *argv* in *field-name=field-value* pairs, and the subroutine looks for the `Template=` and `UserId=` in the input parameters.
- When the exit code finds a `Template=` value containing ″PKI Browser Certificate For Authenticating To z/OS″, the _check_resource_auth_np() system function examines the user ID. This determines the user's access to the preceding profiles.
  - If the user has no access to either of these resources, return code 8 is set. This causes the request to be denied.
  - Otherwise the user's TITLE is set by writing the `TITLE=`*`title-value`* string to STDOUT.

By default, administrator approval is not required for the PKI browser certificate for authenticating to z/OS.

- When the use has only READ access to PROJ.PARTNER, the function must be changed to require administrator approval. This is done by setting return code 4.
- For all other accesses the function does not need to be changed.

Scenario 2: Maintain a customized certificate repository (database) independent of PKI Services

This scenario is for maintaining a customized certificate repository (database) that is independent of PKI Services. After a successful submission of a certificate request, PKI Services returns the transaction ID. This is saved in a new customer-provided database entry. An alias for this database entry is then returned to the end user as the transaction ID. Later, when the user wishes to pick up the certificate, the user-entered alias name is used to retrieve the actual PKI Services transaction ID. The retrieved certificate is saved in the database entry before being returned to the user.

Three different exit calls handle the preceding logic.

- Post-processing for the GENCERT or REQCERT functions (subroutine postProcessGenReqCertExit) returns a pretend alias entry name by suffixing the actual transaction ID with either ″SAF″ or ″PKI″. This is where the database entry should be created. (Note that the exit performs no actual database calls because this would be too customer-specific.)
- Preprocessing for the EXPORT function (subroutine preProcessExportExit) reverts the transaction ID to its original value. This emulates retrieval from the database entry.
- Post-processing for the EXPORT function (subroutine postProcessExportExit) saves the returned certificate to a database entry. This is emulated by writing it to a file.

Scenario 3: Mandate a policy for certificate renewal only within 30 days of expiration

This scenario is for mandating a policy that allows users to renew their certificates only when certificates are within 30 days of expiring. When the condition is met, you can change the expiration date for the renew request so that the new certificate's validity period is extended by the number of days specified by the NotAfter parameter. In other words, the new certificate should expire *n* days from the current date, where *n = number of days left in the old certificate's validity period + number of days specified by NotAfter*.

The preprocessing exit call for GENRENEW and REQRENEW functions (subroutine preProcessGenReqRenewExit) handles the preceding logic. Here are the steps:

- The user's certificate is extracted from the environment variable HTTPS_CLIENT_CERT.
- The *NotAfter* value is extracted from the input parameters (*argv*), converted to a number, and saved in the variable *requestPeriod*.
- Subroutine determineExpiration is called to extract the expiration date from the user's certificate. This subroutine calls several lower subroutines to base64 decode the certificate, DER decode the binary certificate, and convert the expiration date to a seconds value.
- Upon return from determineExpiration, the variable *timeBeforeExp* is the number of seconds from now that the certificate expires. This is compared against the number of seconds in 30 days (86400 * 30) to see if it is greater than 30 days.

  - If it is greater than 30, the request is rejected by setting return code 8.
  - If it is not greater than 30, the new *NotAfter* value is computed as *timeBeforeExp*/86400 + *requestPeriod*.
- This new NotAfter value is set by writing it to STDOUT.

End of Programming Interface information

Part 4. Using PKI Services

This part of the book explains how to use the PKI Services Web pages.

- Chapter 13, “Using the end-user Web pages” on page 125 shows the Web pages for the end user and explains how to perform tasks such as requesting a certificate, obtaining the certificate, and renewing or revoking a certificate.
- Chapter 14, “Using the administration Web pages” on page 141 shows the administration Web pages and explains how to process certificate requests and certificates.

Chapter 13. Using the end-user Web pages

This chapter describes how the end user can use the PKI Services Web pages.

Note: The PKI Services Web pages in this chapter may differ slightly from those on the Web. If you need to see the exact content, view the pages on the Web. Additionally, the pages may contain differences depending on the browser you are using. This chapter assumes you are using Internet Explorer.

By default, the end user can:

- Install a CA certificate into the browser
- Request a new certificate
- Pick up a previously requested certificate
- Renew or revoke a previously issued browser certificate

The following table lists the types of certificates you can request:

Table 38. Types of certificates you can request

Type of certificate	Use
One-year PKI SSL browser certificate	End-user client authentication using SSL
One-year PKI S/MIME browser certificate	Browser-based e-mail encryption
Two-year PKI browser certificate for authenticating to z/OS	End-user client authorization using SSL when logging onto z/OS
Five-year PKI SSL server certificate	SSL Web server certification
Five-year PKI IPSEC server (firewall) certificate	Firewall server identification and key exchange
Five-year PKI intermediate CA certificate	Subordinate (non-self-signed) Certificate Authority certification
One-year SAF browser certificate	End-user client authentication where RACF (not PKI Services) is the certificate provider
One-year SAF server certificate	Web server SSL certification where RACF (not PKI Services) is the certificate provider

Note: If your installation has not customized the certificate templates, the PKI Services Web pages in this chapter may still differ slightly from those on the Web; if your installation customized the templates, the Web pages in this chapter may differ greatly from those you view on the Web.

Steps for accessing the end-user Web pages

Perform the following preliminary steps to access the PKI Services Web pages:

1. Get your organization's URL for accessing the PKI Services Web pages. Enter this URL in your browser. This takes you to the PKISERV certificate generation application Web page (shown in the following figure):

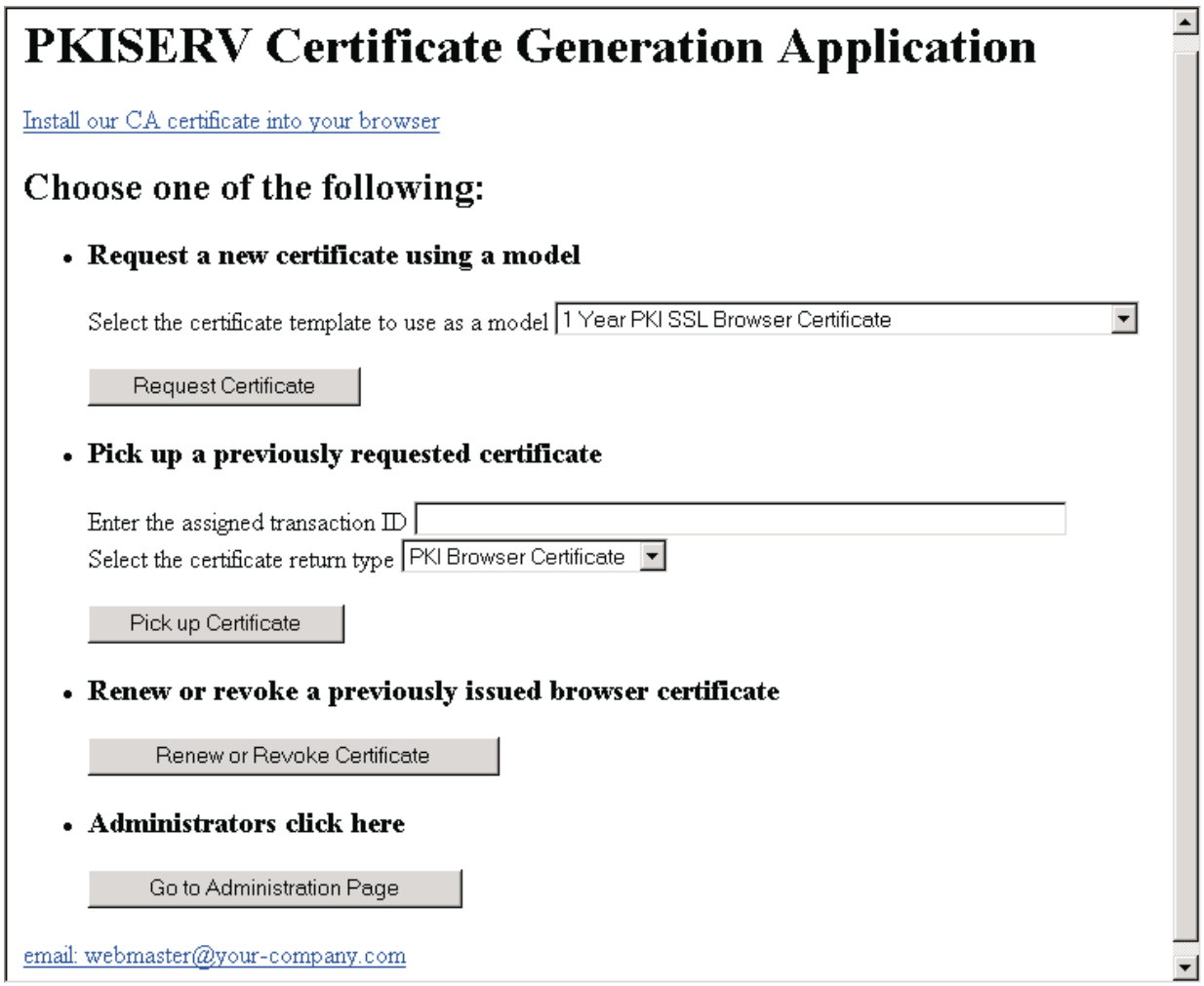


Figure 4. PKISERV certificate generation application Web page

2. If this is the first time you have accessed the forms on these Web pages, to install the CA certificate into your browser. Click the **Install our CA certificate into your browser** link and follow the directions.

   The following is a sample of the directions to follow for installing the CA certificate on Internet Explorer:

   a. After you click the **Install our CA certificate into your browser** link, a popup window called ″File download″ appears. Make sure the ″Open this file from its current location″ radio button is selected (rather than ″Save this file to disk″). Then click the **OK** button. This displays the following popup window:

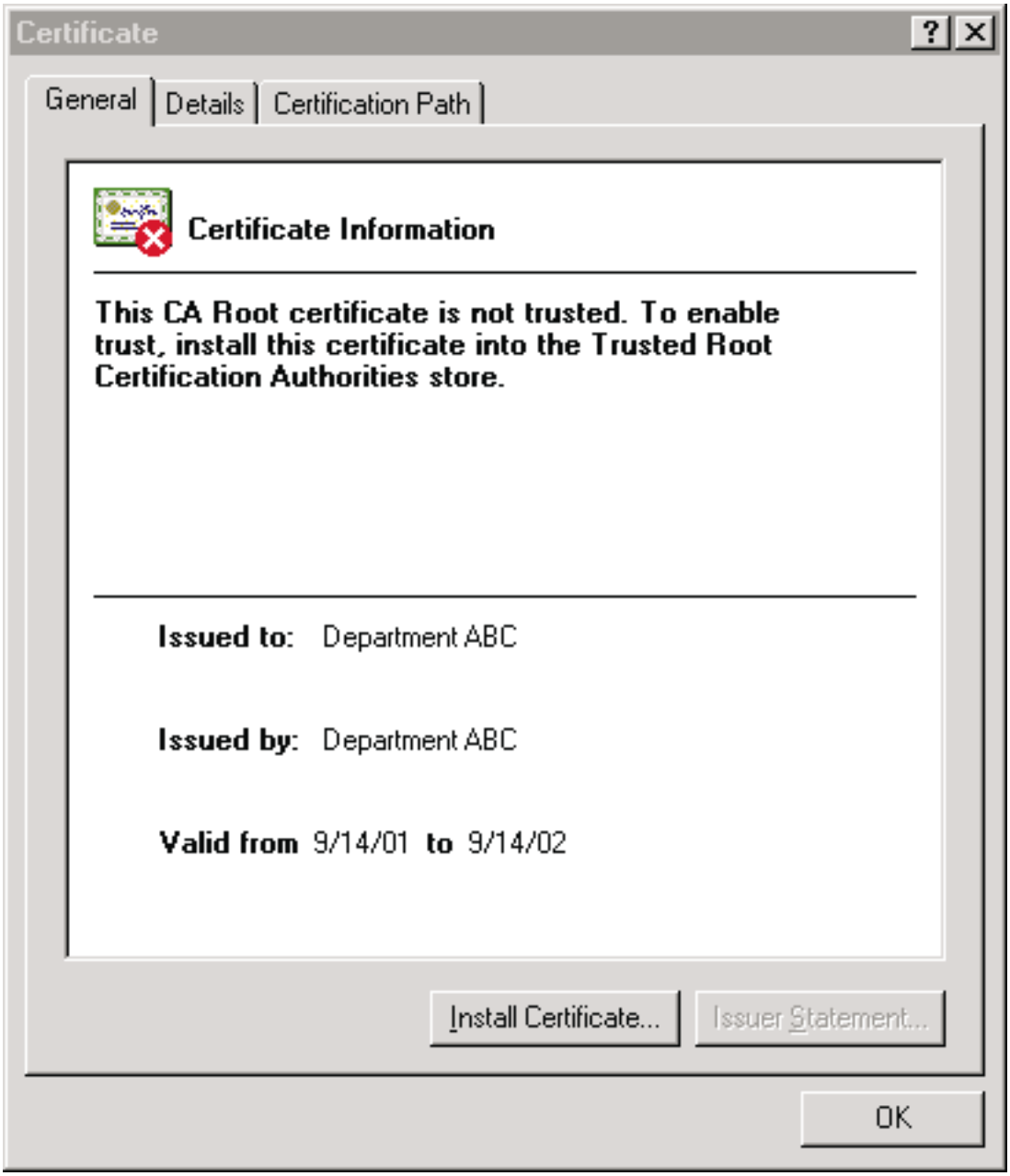


Figure 5. The Certificate popup window for installing the CA certificate

b. Click the **Install certificate** button. (This initiates a sequence of pop-ups in which you need to click **Next** buttons and finally the **Finish** button, culminating in a popup window that says, ″The import was successful.″)

You are now ready to perform tasks, such as:

- Requesting a new certificate
- Picking up a previously requested certificate
- Renewing or revoking a previously issued browser certificate

Summary of fields

When you request certificates, you provide information for the fields in certificate request forms. The following table describes the fields in the end-user Web pages:

Table 39. Summary of fields in end-user Web pages

Field	Description
Base64-encoded PKCS #10 certificate request	(This is for server or device enrollment only.) You create a certificate request on behalf of another server (which could be a z/OS server or other type of server) or device for which you are requesting a certificate. You use software specific to that server to generate the PKCS#10 request before going to the PKI Services Web site. Save the request in a file. Then open the file in a text editor such as Windows Notepad and copy and paste the contents into the text box on the enrollment form. A text area of 70 columns and 12 rows is allocated for this certificate request. Here is an example of the certificate request: `MIIBiDCB8gIBADAZMRcwFQYDVQQDEw5Kb2huIFEuIFB1YmxpYzCBnzANBgkqhkiG 9w0BAQEFAAOBjQAwgYkCgYEAsCT1cJHAGPqi6OjAyL+xNbt8z5ngmvqO2V0O3oYu /mEnQtRM96e+2jbmDCRo5tWVklG4OYf9ZVB5biURMJFLztfa4AVdEVtun8DH2pwc wiNIZZcC1Zym5adurUmyDk64PgiiIPMQS/t0ttG4c5U8uWSK0b1J4V4f7ps+t1aG t+cCAwEAAaAwMC4GCSqGSIb3DQEJDjEhMB8wHQYDVR0OBBYEFAlKTovBBvnFqDAO 1oIhtRinwRC9MA0GCSqGSIb3DQEBBQUAA4GBAIbCVpwYvppIX3HHmpKZPNY8Snsz AJrDsgAEH51WOIRGywhqKcLLxa9htoQai6cdc8RpFVTwk6UfdCOGxMn4aFb34Tk3 5WYdzOiHXg8MhHiB3EruwdWs+S7Fv3JhU3FLwU6lFLfAjbVi+35iEWQymOR6mE5W CathprmGfKRsDE5E`</br>
Challenge passphrase	This is the passphrase you entered when requesting a certificate. You type the same passphrase, exactly as you typed it on the request form. This is a case-sensitive text field of up to 32 characters.
Common name	Your name, such as John Smith. (You can use your first and last name, in that order.) This is a text field of up to 64 characters. **Note:** For SSL servers, the common name is the server's fully qualified domain name, for example, `www.ibm.com`.
Country	The country where your organization is located. This is a 2-character text field.
Cryptographic service provider	(This is for the Internet Explorer browser only.) The Cryptographic Service Provider to generate your public/private key pair. You select a value from the drop-down list. The default selection is ″Microsoft Enhanced Cryptographic Provider″; this provides 1024-bit key encryption. The other choice is the ″Microsoft Base Cryptographic Provider″; this provides 512-bit key encryption. Larger keys are more secure, but they also increase the time that is needed for connecting to a secure session.
Domain name	The host name of the machine where a certificate will be installed. This is a text field of up to 100 characters. **Note:** The value is one of the list of subject's alternate names that is saved in the subject alternate name extension in the certificate.
E-mail address	Your e-mail address, including the @ character and any periods (.). This is a text field of up to 100 characters. **Note:** The value is one of the list of subject's alternate names that is saved in the subject alternate name extension in the certificate.
HostIdMappings extension	This is the user ID for authorization purposes in the following format: `subject-id@host-name` for example, DSmith@ibm.com. This is a text field of up to 100 characters.
IP address	The unique IP version 4 address that specifies the location of each device or workstation on the Internet, for example, 9.67.97.103. (PKI Services supports only IP version 4 addresses.) The IP address is in dotted decimal format and is a text field of up to 15 characters. **Note:** The value is one of the list of subject's alternate names that is saved in the subject alternate name extension in the certificate.
Key protection	(This is for the Internet Explorer browser only.) This asks if you want to enable private key protection. (The dropdown choices are Yes and No.)

Table 39. Summary of fields in end-user Web pages (continued)

Field	Description
Key size	(This is for the Netscape browser only.) This is the key size for your public/private key pair. Select a value from the drop-down list. Larger keys are more secure, but they also increase the time needed for connecting to a secure session.
Key usage	This indicates the intended purpose of the certificate. Possible values are: **handshake** Protocol handshaking (for example, SSL) **dataencrypt** Data encryption **certsign** Certificate signing **docsign** Document signing
Label	The label assigned to the requested certificate. This is a text field of up to 32 characters.
Locality	The city or municipality where your organization is located, such as Pittsburgh or Paris. This is a text field of up to 64 characters.
Not before (date)	A number of days, added to the current date (by default, you can select either 0 or 30), before which the certificate is not valid.
Not after (date)	A number of days, added to the current date, after which the certificate expires. By default, you can select either 1 year or two years for the time at which the certificate expires.
Organization	The legally registered name (or trademark name, for example, IBM) of your organization. This is a text field of up to 64 characters.
Organizational unit	The name of your division or department. (There can be more than one organizational unit field on a request form. For example, one could be for your department and another for your division.) This is a text field of up to 64 characters.
Pass phrase	You decide this value when requesting a certificate (and must later supply this value when retrieving the certificate). You enter and then reenter this when requesting a certificate. This is a case-sensitive text field of up to 32 characters. (There is no minimum number of characters, and you can use any characters, but alphanumeric characters (A–Z, a–z, and 0–9) are recommended.
State or Province	The state or province where your organization is located. Your registration policies determine whether you spell out the full name of the state or province or use an abbreviation. This is a text field of up to 64 characters.
Title	Your job title. This is a text field of up to 64 characters.
Transaction ID	PKISERV Web pages assign this after you request your certificate. When it is displayed, you need to record this number. This is a text field of up to 56 characters.
Uniform resource identifier (URI)	A name or address referring to an internet resource; a URL is one kind of uniform resource identifier. This is a text field of up to 100 characters. **Note:** The value is one of the list of subject's alternate names that is saved in the subject alternate name extension in the certificate.
Your name	Your name (for tracking purposes). This can be in any format, for example, John Smith or John. J. Smith. This is a text field of up to 32 characters.

Steps for requesting a new certificate

To request a new certificate, first go to the PKI Services home page (see Figure 4 on page 126).

Perform the following steps to request a new certificate:

1. Click the down arrow to the right of the field beside ″Select the certificate template to use as a model.″ This displays a list of certificate templates from which you can select.

   Note: The following list shows the certificate templates that PKI Services provides by default. This list may differ from the certificate templates your

installation provides because your installation can customize the certificate templates and Web pages.

- 1-year PKI SSL browser certificate
- 1-year PKI S/MIME browser certificate
- 2-year PKI browser certificate for authenticating to z/OS
- 5-year PKI SSL server certificate
- 5-year PKI IPSEC server (Firewall) certificate
- 5-year PKI intermediate CA certificate
- 1-year SAF browser certificate
- 1-year SAF server certificate

---

2. Click one of the items in the list. The drop-down list then collapses so that only the certificate you selected appears in the field and is highlighted.

---

3. Click the **Request certificate** button. A form where you fill in information is displayed.

   Note: You may need to click through some additional panels specific to your browser (for example, clicking **Next** on Netscape or answering ″Do you want to proceed″ on Internet Explorer) before the certificate request form appears.

---

4. Fill in the necessary information in the certificate request form. For example, if you are requesting a one-year SSL browser certificate, the following form appears:

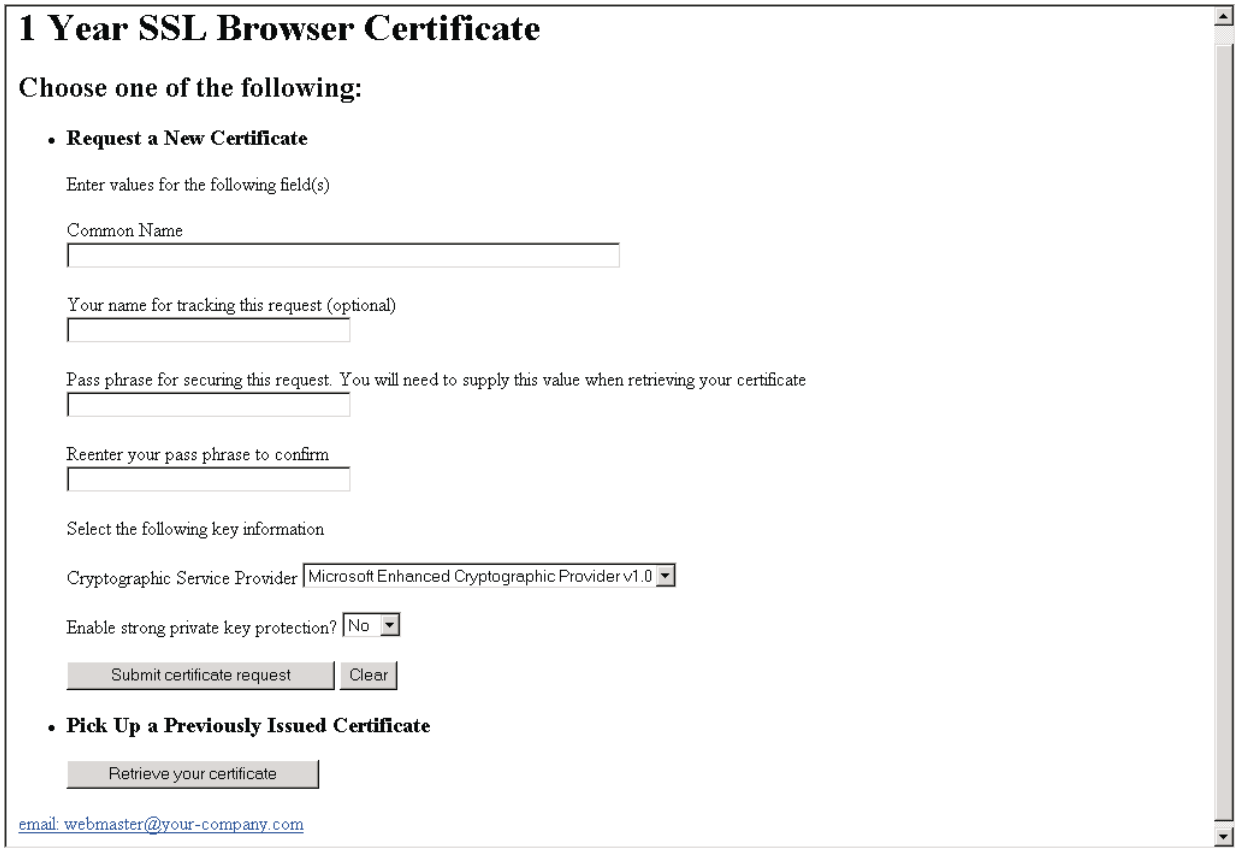


Figure 6. One-year SSL browser certificate request form

Note: The form that appears depends on the certificate you are requesting and, in some instances, the fields that appear on the form depend on the browser you are using.

a. In the case of the one-year SSL browser certificate, fill in your common name (see Table 39 on page 128 for descriptions of fields) and passphrase (twice). If you are using Netscape, select a key size from a drop-down list. Alternately, if you are using Internet Explorer, click the drop-down lists to select your cryptographic service provider and to specify whether to use strong private key protection.

b. When you are satisfied with the information you have entered, click the **Submit certificate request** button.

---

5. If the request is successful, you see a page like the following, which tells you your transaction ID.

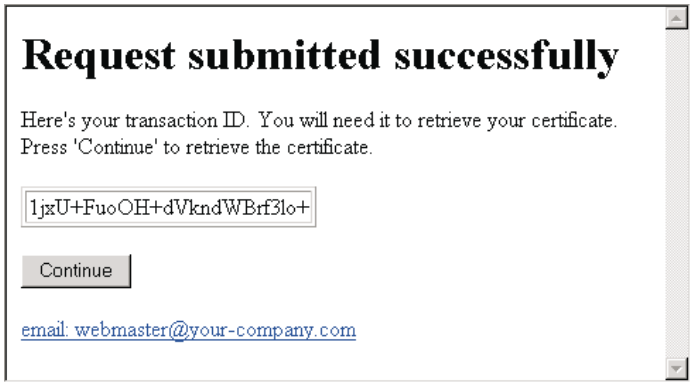


Figure 7. Successful request displays transaction ID

a. Make a note of the Transaction ID. (You can copy and paste the Transaction ID to a file so that you have it for future reference, or you can write it in the box below. The reason for keeping a record of the Transaction ID is that, depending on how you go to the Web page to retrieve your certificate (see Figure 8 on page 133), you may have to fill in the transaction ID on that Web page.)

Transaction ID:	

b. Click the **Continue** button. This displays the following Web page:

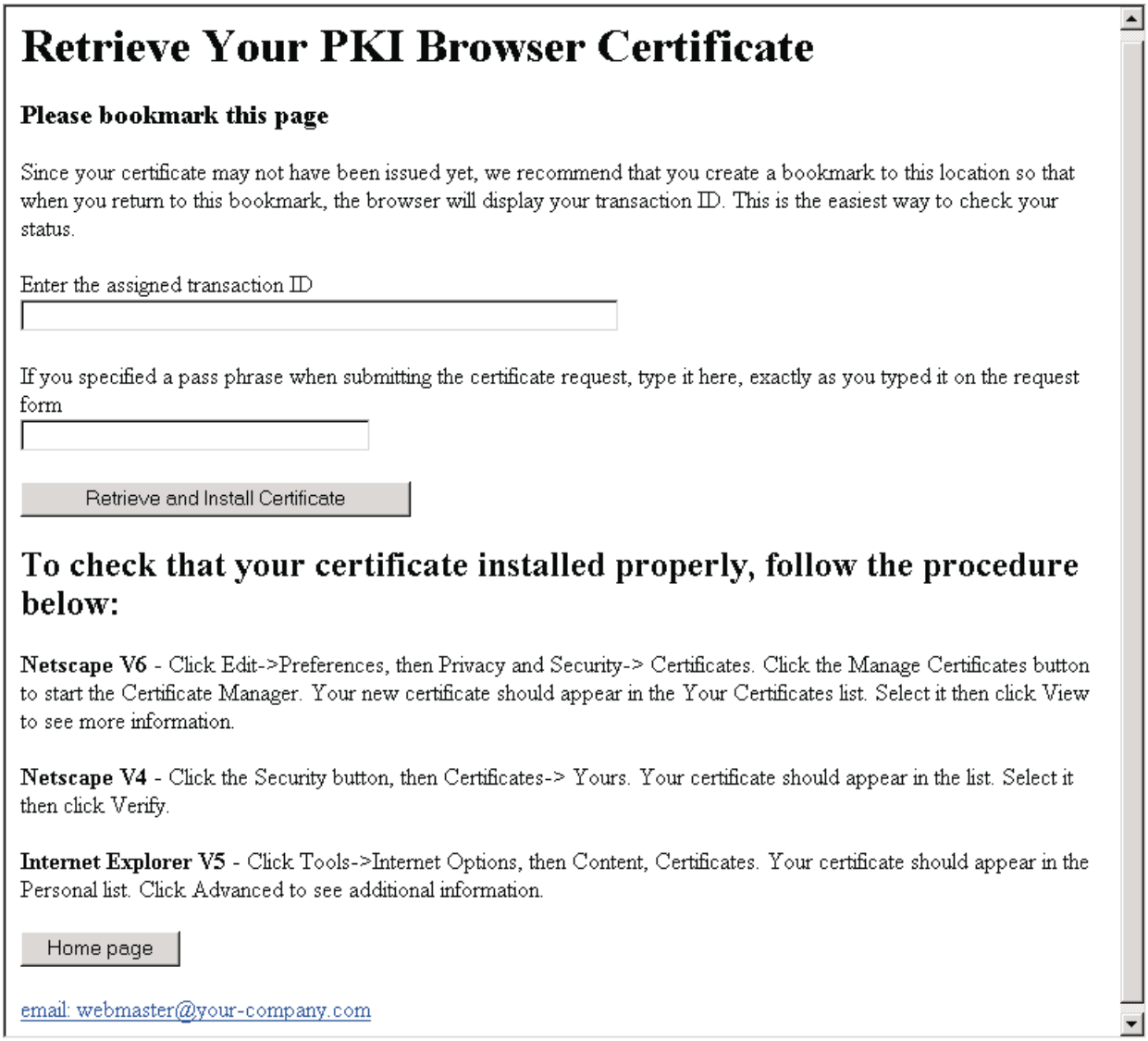


Figure 8. Web page to retrieve your certificate

c. Bookmark this Web page.

   Note: After you submit the request for a certificate, your PKI Services administrator may need to approve the request before you can pick up your certificate. The amount of time that this takes can vary from a few minutes to a few days, depending on your installation. You bookmark this Web page so that you can return to it at a later time.

d. From this Web page, you can start the steps to retrieve your certificate (see "Steps for retrieving your certificate from the bookmarked Web page") or you can return to the PKI Services home page (by clicking the **Home** button).

Steps for retrieving your certificate from the bookmarked Web page

You can retrieve your certificate:

- From Web page you bookmarked in Step 5c on page 133. (This Web page contains your transaction ID, so you do not have to enter it.) The steps that follow are for retrieving your certificate from the bookmarked Web page.
- From the PKI Services home page (see Figure 4 on page 126 and “Steps for retrieving your certificate from the PKI Services home page” on page 136).

Perform the following steps to retrieve your certificate from the bookmarked Web page:

1. Go to the bookmarked Web page. (See Figure 8 on page 133.)

   _____

2. If you entered a passphrase when requesting your certificate, enter the passphrase.

   _____

3. Click the **Retrieve and install certificate** button. If you are using Netscape, go to Step 5 on page 135. If you are using Internet Explorer and the retrieval of a certificate is successful, this displays the Web page shown in Figure 9. (This is for a browser certificate. For a server certificate, Figure 10 on page 135 shows an example of the Web page.)

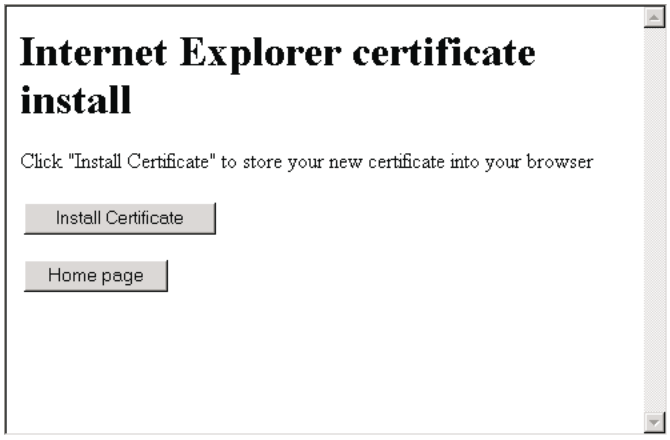


Figure 9. Browser certificate installation Web page

Figure 10. Server certificate installation Web page

4. Click the **Install certificate** button. If the certificate installs successfully, you get a popup window that says, ″Your new certificate installed successfully.″

5. Check that your certificate installed correctly:
   - For Netscape, click the **Security** button, then Certificates -> Yours. Your certificate should appear in the list. Select it and click Verify.
   - For Internet Explorer, Click Tools -> Internet Options, then Content, **Certificates**. Your certificate should appear in the Personal list. Click Advanced to see additional information.

Steps for retrieving your certificate from the PKI Services home page

Before you begin:

To retrieve your certificate from the PKI Services home page, you must first know your transaction ID. You should have recorded this when your certificate request was successful. (See Figure 7 on page 132.)

Perform the following steps to retrieve your certificate from the PKI Services home page:

1. Enter your transaction ID and select the certificate type using the drop-down. Then click the **Pick up certificate** button on the PKI Services home page. (See Figure 4 on page 126.) This displays the Web page that Figure 8 on page 133 displays.

   ---

2. Enter your passphrase (this is the challenge passphrase) if you specified one when requesting your certificate.

   ---

3. Click the **Retrieve and install certificate** button. If you are using Netscape, go to Step 5. If you are using Internet Explorer and the retrieval of the certificate is successful, this displays the Web page that Figure 9 on page 134 shows. (This is for a browser certificate. For a server certificate, Figure 10 on page 135 shows an example of the Web page.)

   ---

4. Click the **Install certificate** button. If the certificate installs successfully, you get a popup window that says, ″Your new certificate installed successfully.″

   ---

5. Check that your certificate installed correctly:
   - For Netscape, click the **Security** button, then Certificates -> Yours. Your certificate should appear in the list. Select it and click Verify.
   - For Internet Explorer, Click Tools -> Internet Options, then Content, Certificates. Your certificate should appear in the Personal list. Click Advanced to see additional information.

Steps for renewing a certificate

Perform the following steps to renew a certificate:

1. On the PKI Services home page (see Figure 4 on page 126), click the **Renew or revoke certificate** button. This displays a popup window with a list of certificates, such as the following figure shows:

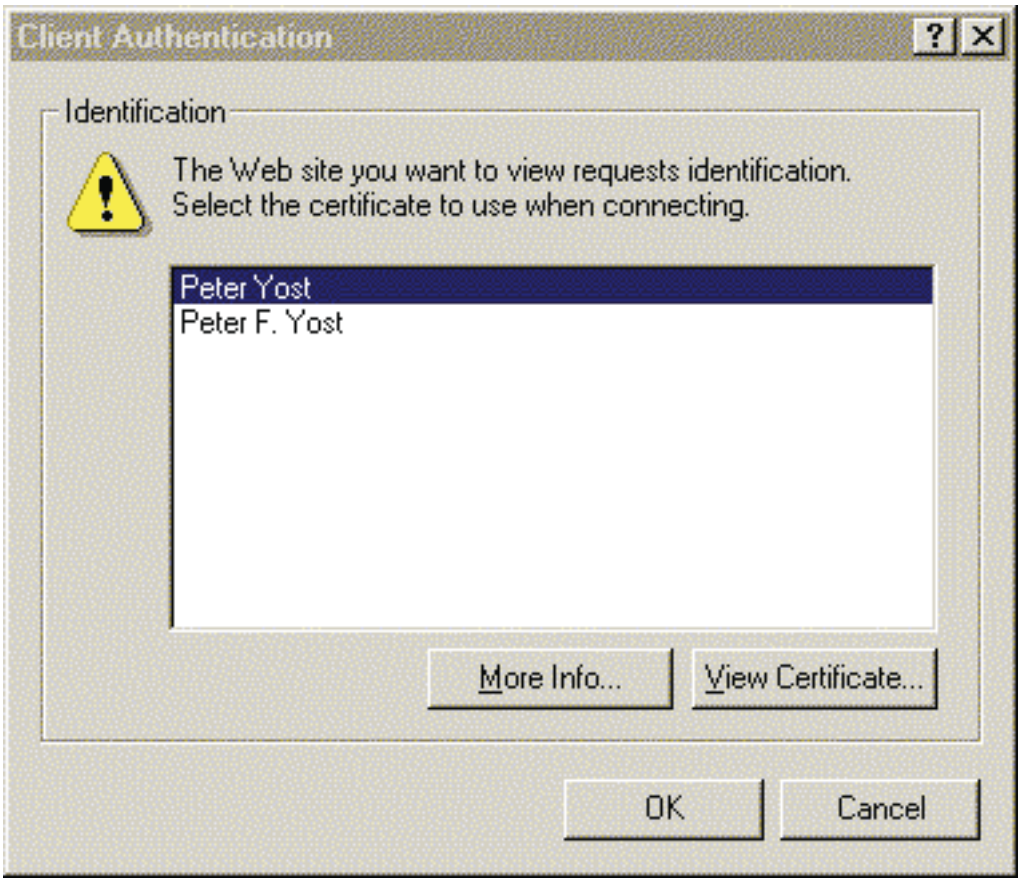


Figure 11. Popup window listing certificates

2. The popup window may list more than one certificate. It lists certificates by nicknames of how they are installed in the browser. Therefore, you may not be able to identify the PKI Services certificate you want to renew. Highlight the entry you think is the right one and click the **OK** button. If the certificate you selected is one that PKI Services issued and it is not expired or revoked, this displays the following Web page:

Renew or Revoke a Browser Certificate

Here is the certificate you selected:

Requestor	Certificate ID / Certificate Names/ Validity	Usage	Status	Dates
Peter Yost	**Serial #:**7 **Template:**1 Year PKI SSL Browser Certificate **Subject:**CN=Peter Yost,OU=Class 1 Internet Certificate CA,O=The Firm **Issuer:**OU=Department ABC,O=Company Inc.,L=Anywhere City,C=zz **Validity:**2001/09/14 00:00:00 - 2002/09/13 23:59:59	handshake	Active	**Created:** 2001/09/14 **Modified:**2001/09/14

If this is the correct certificate, choose one of the following:

(otherwise you need to restart your browser to pick another certificate)

- **Renew the above certificate**

  Pass phrase for securing this request. You will need to supply this value when retrieving your certificate

  Reenter your pass phrase to confirm

  Renew

- **Revoke the above certificate**

  Revoke | No Reason

Home Page

email: webmaster@your-company.com

Figure 12. Renew or revoke a certificate Web page

Note: If this is not the PKI Services certificate you want to renew, you need to close your browser (because the browser caches information) before again clicking the **Renew or revoke certificate** button as in Step 1 on page 136.

3. Under the ″Renew the above certificate″ section, enter your passphrase in the two fields requesting it.

4. Click the **Renew** button.

5. If the renewal request is successful, this displays a Web page that says ″Request submitted successfully″ and displays the transaction ID. Click the **Continue** button on this Web page.

---

6. This takes you the Web page from which you retrieve your certificate (see Figure 8 on page 133 for an example of this Web page and “Steps for retrieving your certificate from the bookmarked Web page” on page 133 for the directions to follow).

Steps for revoking a certificate

Revoking a certificate means that you cannot continue to use the certificate. You might want to revoke your certificate if you suspect your private key has been compromised.

Perform the following steps to revoke a certificate:

1. On the PKI Services home page (see Figure 4 on page 126), click the **Renew or revoke certificate** button. This displays a popup window with a list of certificates, as in Figure 11 on page 137.

---

2. The popup window may list more than one certificate. The way it lists certificates by nicknames of how they are installed in the browser. You may not be able to identify the PKI Services certificate you want to revoke. Highlight the entry you think is the right one and click the **OK** button. If the certificate you selected is one that PKI Services issued and it is not expired or revoked, this displays the ″Renew or revoke a browser certificate″ Web page (see “Steps for renewing a certificate” on page 136).

   Note: If this is not the PKI Services certificate you want to revoke, you need to close your browser before again clicking the **Renew or revoke certificate** button as in Step 1 on page 136.

---

3. Make sure the certificate you want to revoke is the one described at the top of the Web page. You can click the drop-down list (of reasons) to select a reason if you wish. Click the **Revoke** button.

---

4. This displays a Web page that says ″Request submitted successfully″ You can click the **Home page** button to return to the PKI Services Home page.

Chapter 14. Using the administration Web pages

This chapter presents background information about certificate requests and certificates and explains how the administrator can use the administration Web pages to perform the following tasks:

- Process a certificate request
  - Approve a request without making changes
  - Approve a request with changes
  - Reject a request
  - Delete a request
- Process a certificate
  - Revoke a certificate
  - Delete a certificate
- Perform searches for certificate requests and certificates

Note: The PKI Services Web pages in this chapter may differ slightly from those on the Web. If you need to see the exact content, view the pages on the Web. Additionally, the pages may contain differences depending on the browser you are using. This chapter assumes you are using Internet Explorer.

Steps for accessing the administration home page

Perform the following preliminary steps to access the administration home page:

1. Get your organization's URL for accessing the PKI Services Web pages. Enter this URL in your browser. This takes you to the PKI Services home page (as shown in the following figure):

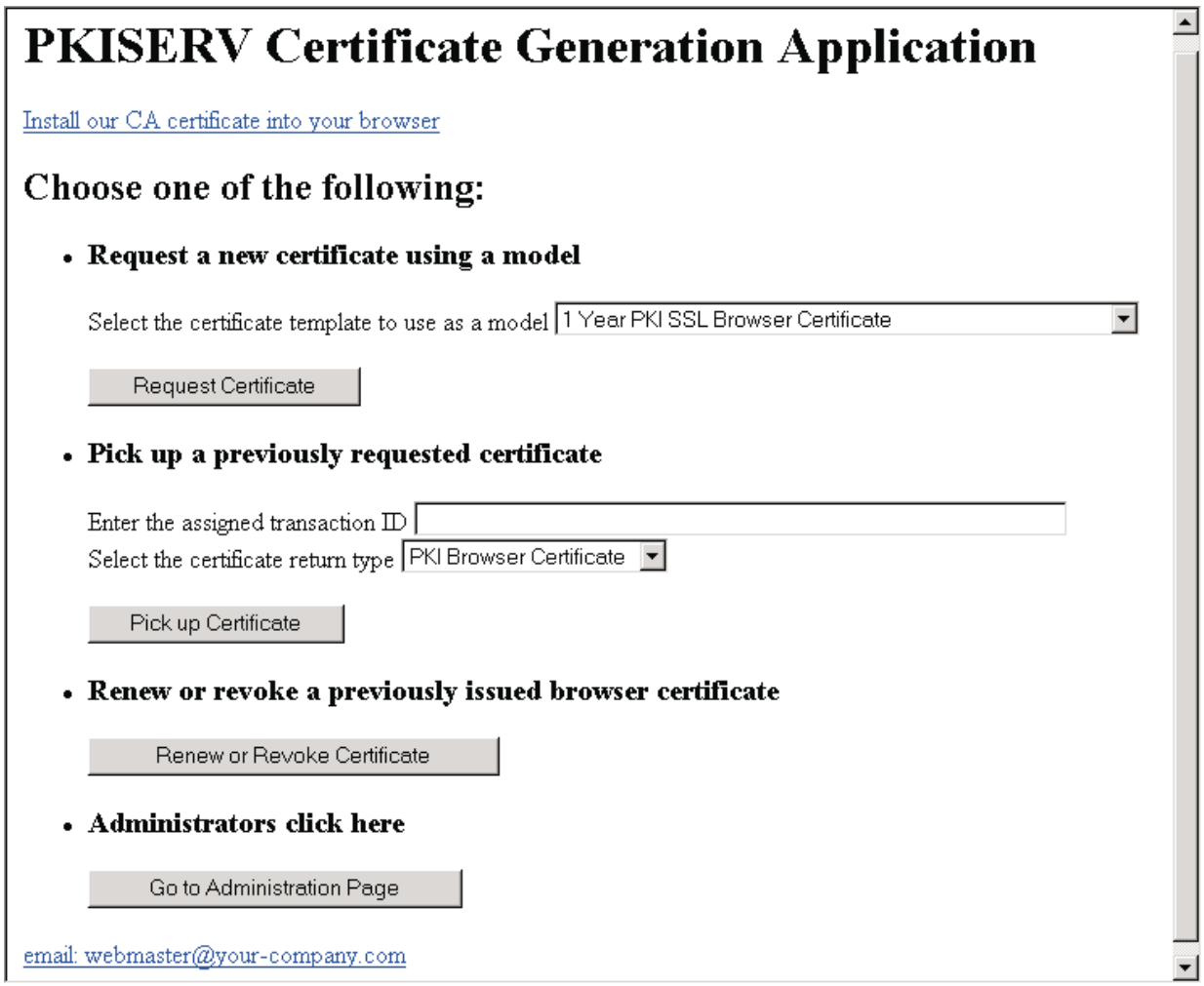


Figure 13. PKI Services home page

2. If this is the first time you have accessed the forms on these Web pages, to install the CA certificate into your browser. Click the **Install our CA certificate into your browser** link and follow the directions.

   The following is a sample of the directions to follow for installing the CA certificate on Internet Explorer:

   a. After you click the **Install our CA certificate into your browser** link, a popup window called ″File download″ appears. Make sure the ″Open this file from its current location″ radio button is selected (rather than ″Save this file to disk″). Then click the **OK** button. This displays the following popup window:

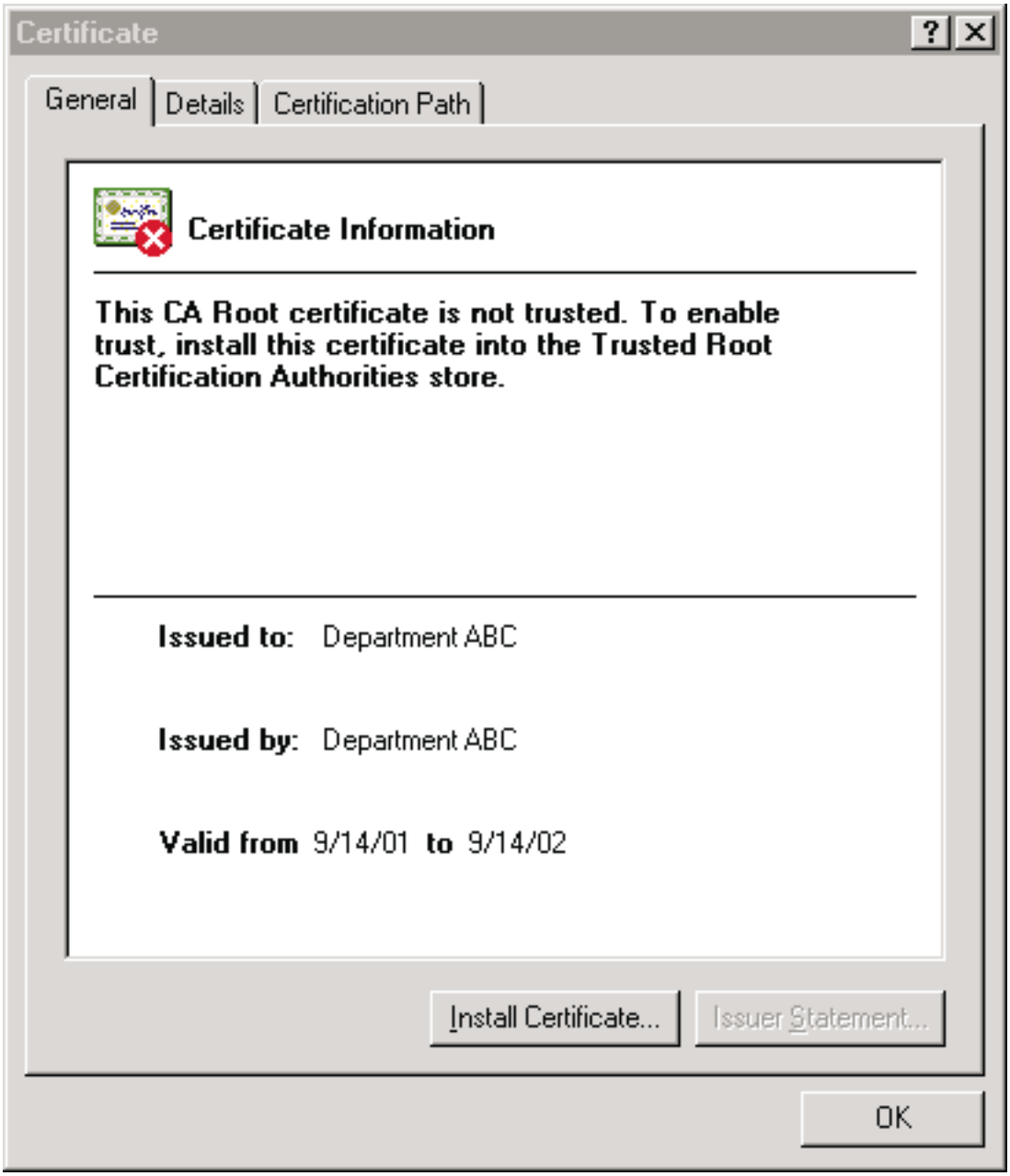


Figure 14. The Certificate popup window for installing the CA certificate

b. Click the **Install certificate** button. (This initiates a sequence of pop-ups in which you need to click **Next** buttons and finally the **Finish** button, culminating in a popup window that says, ″The import was successful.″)

3. Click the **Go to administration page** button.

4. You will be prompted to authenticate, as shown in the following figure. Provide the necessary information:

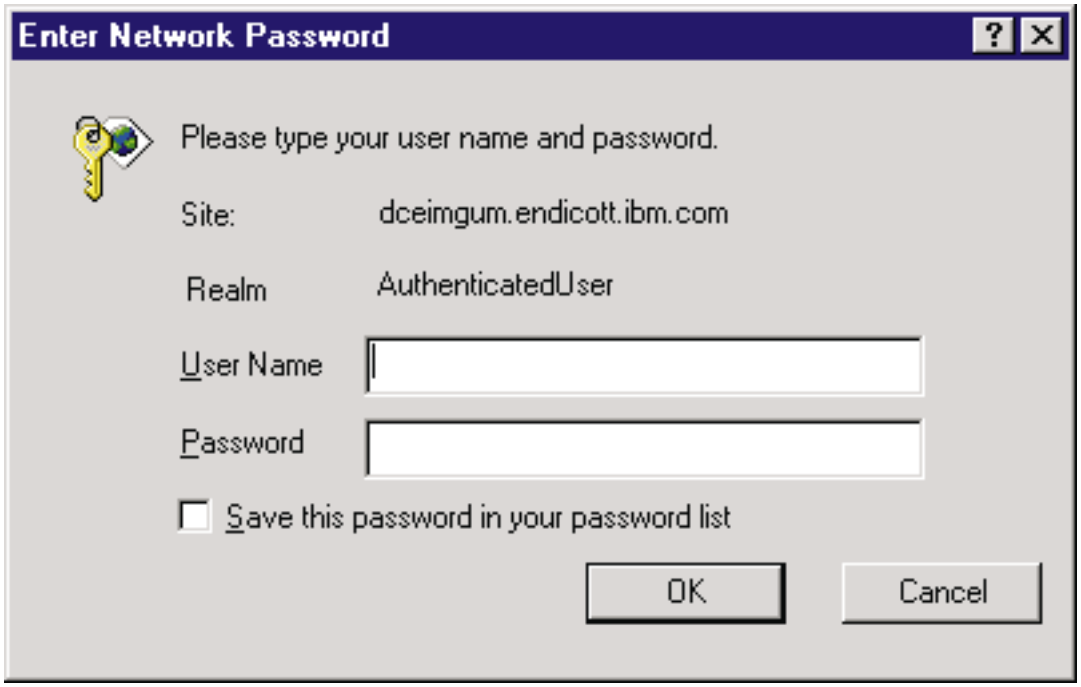


Figure 15. Entering your user ID and password

a. Fill in your z/OS user ID and password
b. If you want to eliminate having to reenter your user ID and password each time you access the administration pages, check the check box.
c. Click **OK**.

   This calls up the ″PKI Services administration″ Web page. (See Figure 16 on page 147.)

Notes:

a. Your Web server programmer may provide you with an alternate URL for accessing the administration home page. You may also have to authenticate using a certificate instead of a user ID and password.
b. Your browser caches the authentication information that you provide. Therefore, if you need to change this information, you first must close all instances of your browser. Then open the browser, and, when the panel shown in Figure 15 appears, enter the correct information.

Fields in the administration Web pages

When you process certificates requests and certificates, you provide information for various fields in the Web pages. The following table describes the fields in the administration Web pages:

Table 40. Summary of fields in the administration pages

Field	Description
Recent activity	This specifies a time range for searches. Possible values include: • Not selected • Within the past day • Within the past week • Within the past month • Within the past six months
Requestor name	The name of the person requesting the certificate, as it appears in the common name field of the certificate request form.
Serial number	PKI Services assigns this number to a certificate when you approve it.
Transaction ID	PKI Services assigns this number to a request when a user requests it. This is a text field of up to 56 characters.

Processing certificate requests

Before you can use the Web page to process certificate requests, you need to understand the statuses of certificate requests and the actions you can perform on these certificate requests.

Status of certificate requests

Requests for certificates are kept in a request database while they are active. This is from the moment they are created until an event occurs that causes them to be deleted. The following table summarizes possible statuses. During the time period when a certificate request is active, it can have only one of the following statuses at a time:

Table 41. Statuses of certificate requests

Status	Meaning
Pending Approval	The request requires administrative approval. No action has been taken on the request yet.
Approved	The administrator explicitly approved the request or it was submitted as an auto-approved certificate request. The actual certificate may or may not have been created at this point.
Completed	The certificate has been issued and the requestor has retrieved it. This is a final state.
Rejected	The administrator rejected the request, and the requestor has **not** been informed of this action (because the user has not tried to retrieve the certificate).
Rejected, User notified	The administrator rejected the request and the requestor has been informed of this action when attempting to retrieve the certificate. This is a final state.

A request is deleted from the request database when the administrator explicitly deletes it or when the request expires. This expiration time period is configurable and varies depending on whether the request was finalized or not.

Actions on certificate requests

The following table summarizes actions on certificate requests and the required status for each of these actions:

Table 42. Summary of actions to perform on requests and required status

Action	Required status of request
Approve	″Pending Approval″
Approve with modifications	
Reject	
Delete	All statuses (″Pending Approval,″ ″Approved,″ ″Completed,″ ″Rejected,″ or ″Rejected, Notified″)

Using the PKI Services administration home page

The following figure shows the PKI Services administration home page:

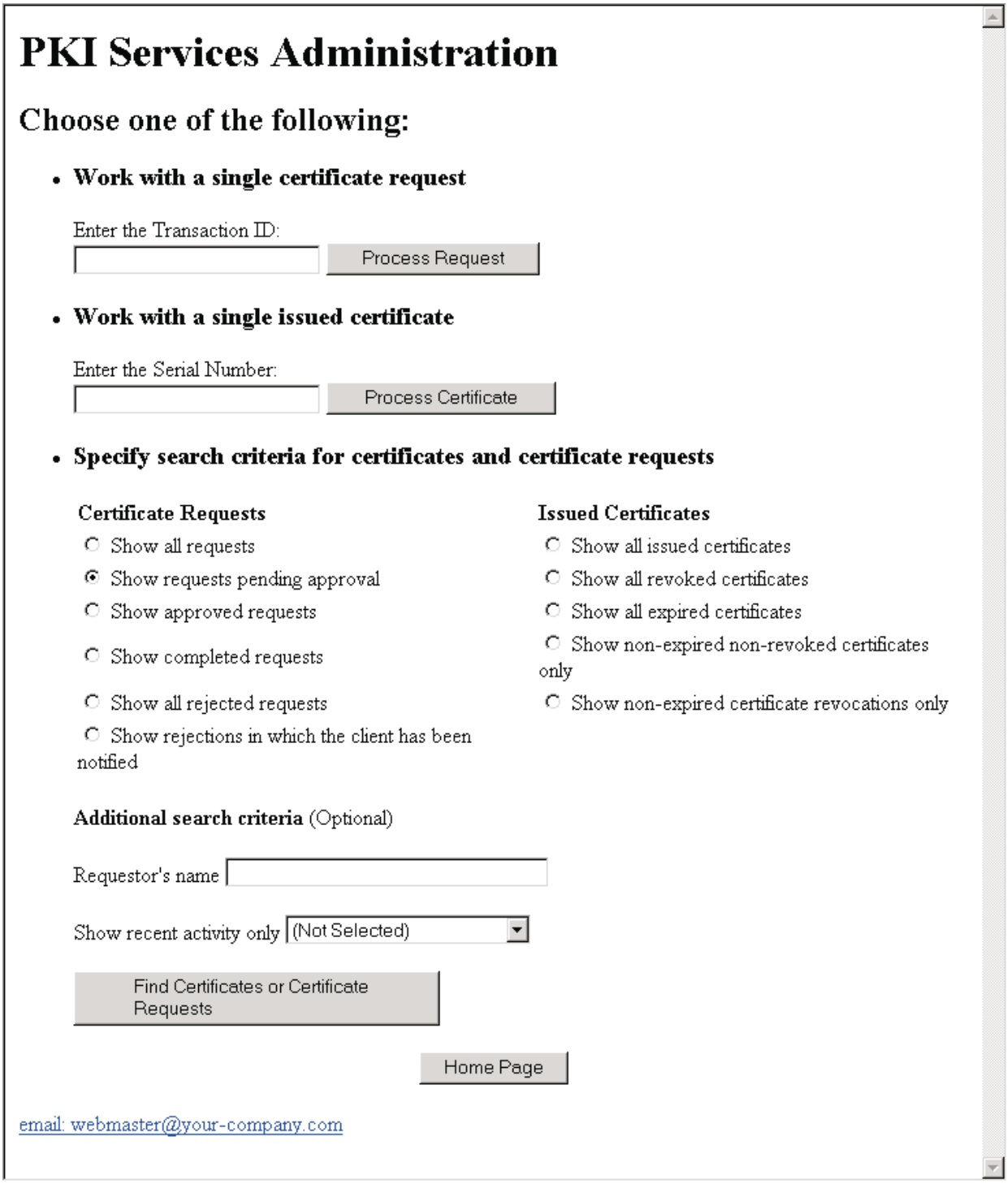


Figure 16. PKI Services administration home page

This Web page allows you to:

- Process a single certificate request (by specifying its transaction ID)
- Process a single certificate (by specifying its serial number)
- Search for groups of certificate requests or certificates by status and additional search criteria so that you can process them

You can process a single certificate request if you know its transaction ID. Otherwise, you can perform a search to display all certificate requests of a particular status.

Steps for processing a single request

To process a single request, perform the following steps:

1. On the PKI Services administration home page (see Figure 16 on page 147), enter the transaction ID in the field provided for it, and click the **Process request** button. This displays the single request approval Web page as shown in the following figure:

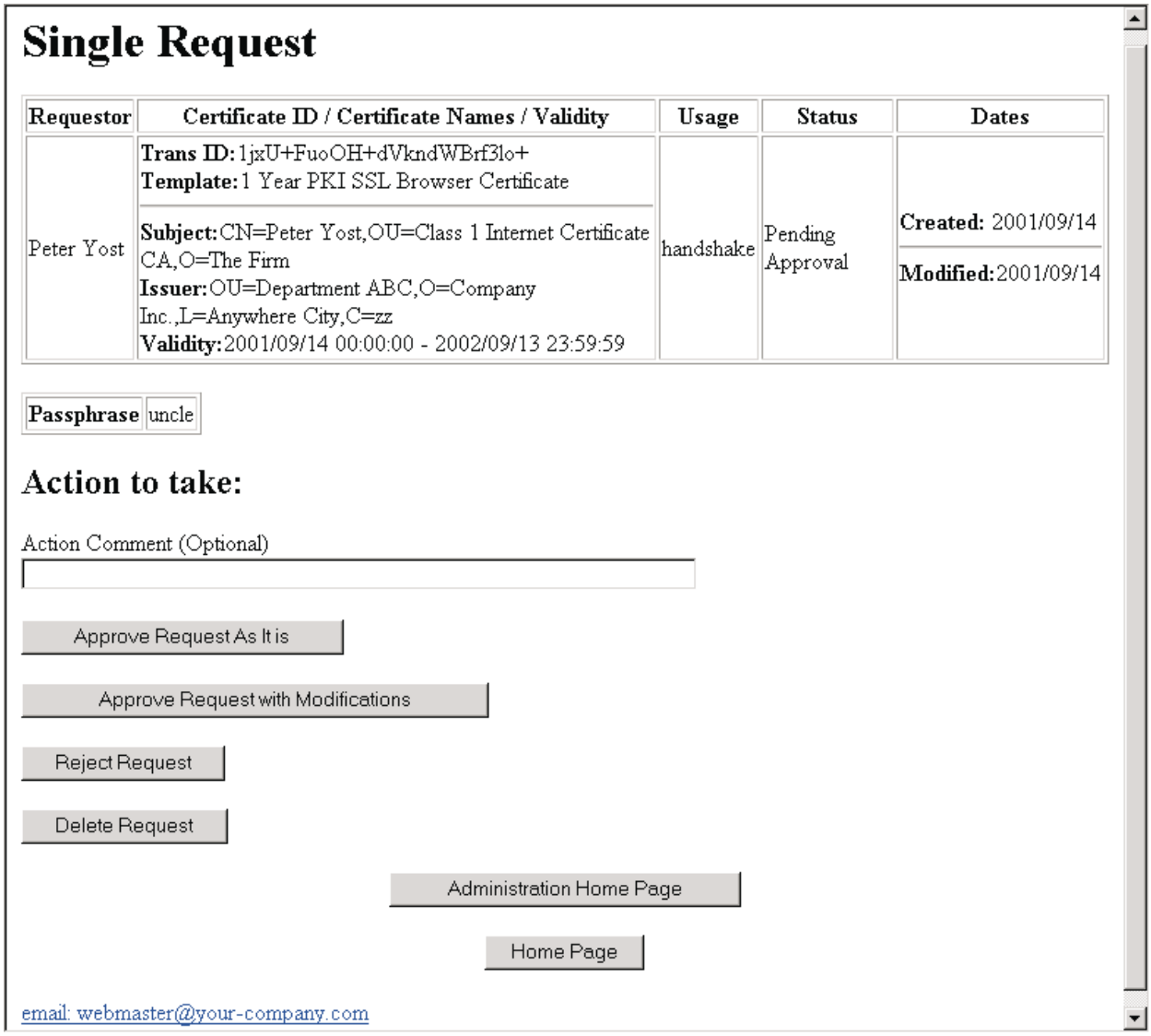


Figure 17. Single request approval Web page

2. Make sure the request is the correct one by reviewing the information in the top part of the Web page.

3. Optionally insert a comment.

4. Click one of the buttons to process the request:
   - **Approve the request as is** button
   - **Approve the request with modifications** button
   - **Reject request** button
   - **Delete request** button

   Note: The buttons that appear on the Web page depend on the status of the request. For example, the top three buttons in the preceding list appear only if the status of a request is ″Pending Approval.″ If the administrator has already processed the request, the only button that appears is the **Delete request** button.

   a. If you click the **Approve the request as is** button and processing is successful, this displays a Web page that says you that ″Processing is successful,″ such as the following:

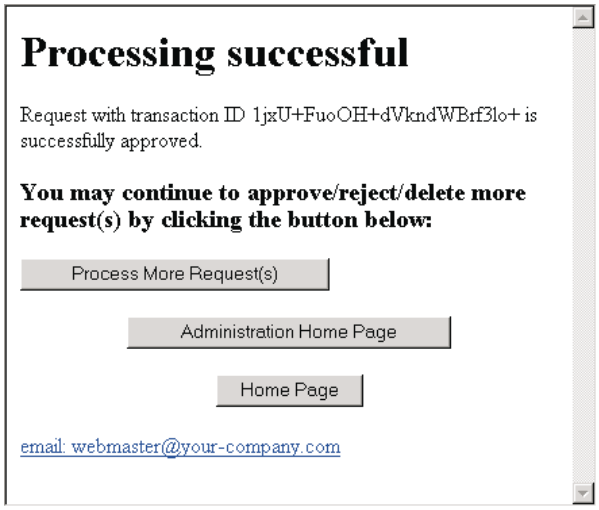


Figure 18. Processing successful Web page

(Otherwise, the Web page says ″Processing is not successful.″) From these Web pages, you can then click the **Process more request(s)** button to return to the PKI Services administration home page (Figure 16 on page 147).

   b. If you click the **Approve the request with modifications** button, this displays the following Web page:

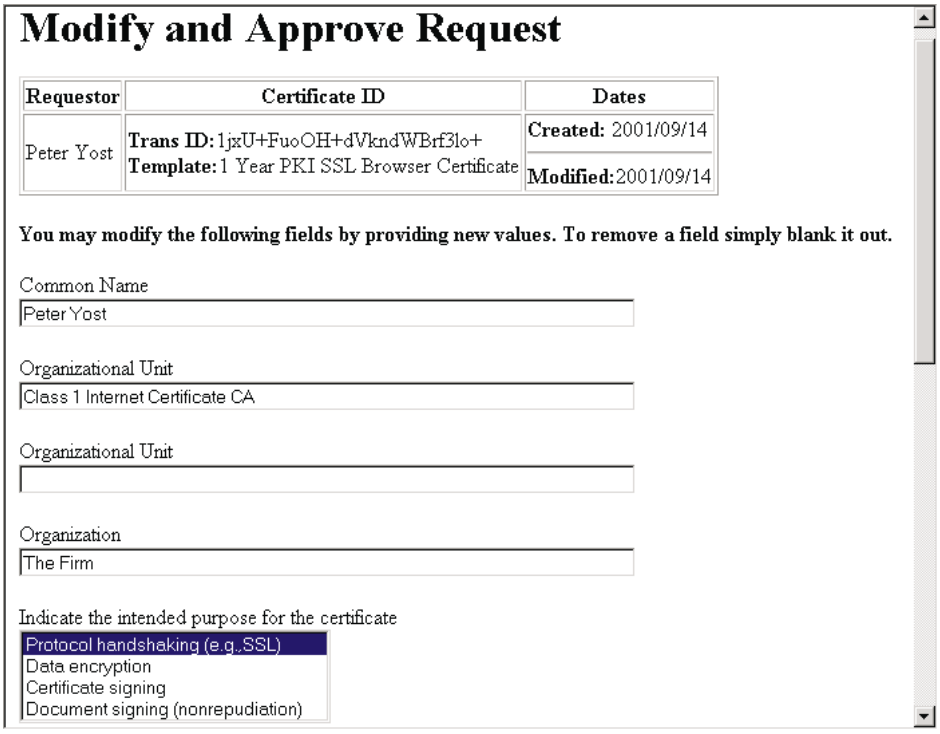


Figure 19. Modifying the request Web page (Part 1 of 2)

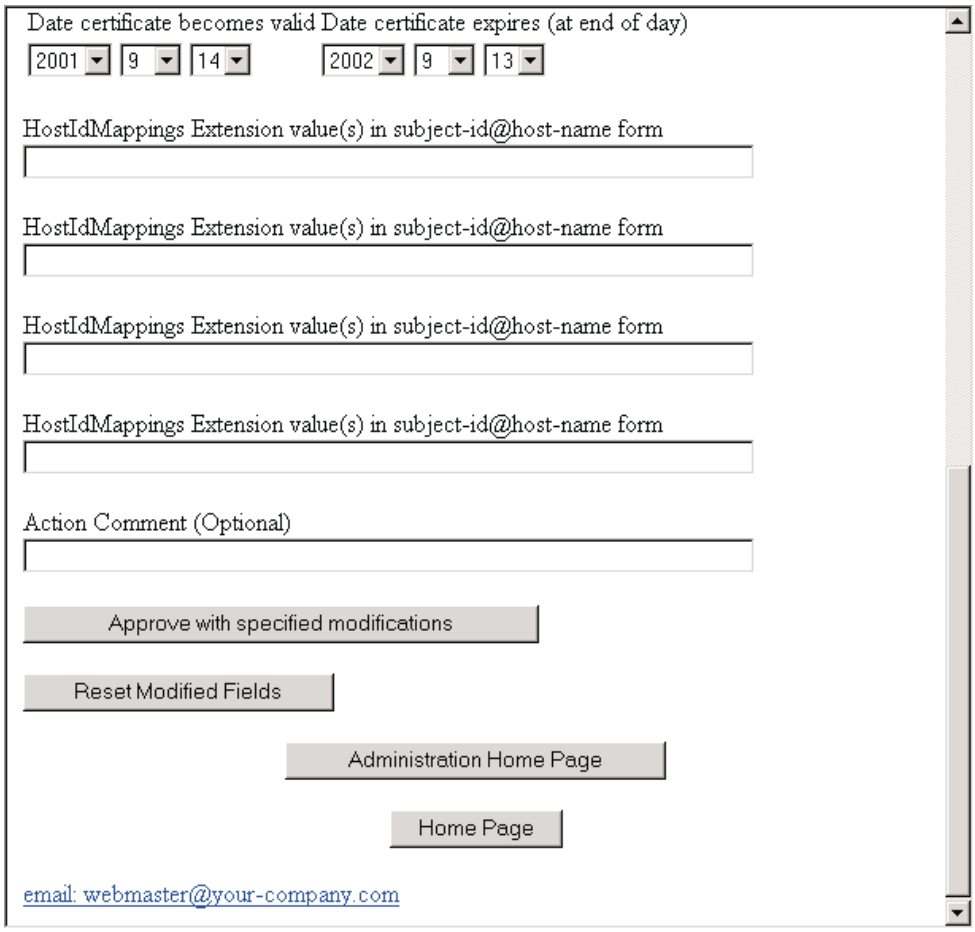


Figure 19. Modifying the request Web page (Part 2 of 2)

On this Web page, you can change the following fields:

- Common name
- Organizational unit(s) (This can be multiple fields)
- Organization
- Certificate purpose
- Date certificate becomes valid
- Date certificate expires
- HostIdMappings extensions (This can be multiple fields)
- Optional comment about action you perform on the certificate.

When you are satisfied with the changes you have made, click the **Approve with specified modifications** button; or, if you change your mind, you can click **Reset modified fields**. Alternately, you can click **Home page** to go to the PKI Services home page (see Figure 13 on page 142).

c. If you click the **Reject request** button, this displays a Web page that informs you that ″Processing is successful″ or that ″Processing is not successful.″ From these Web pages, click the **Process more request(s)** button to return to the PKI Services administration home page (Figure 16 on page 147).

d. If you click the **Delete request** button, this displays a Web page that informs you that ″Processing is successful″ or that ″Processing is not successful.″ On these Web pages, click the **Process more request(s)** button to return to the PKI Services administration home page (see Figure 16 on page 147).

Steps for processing requests by performing searches

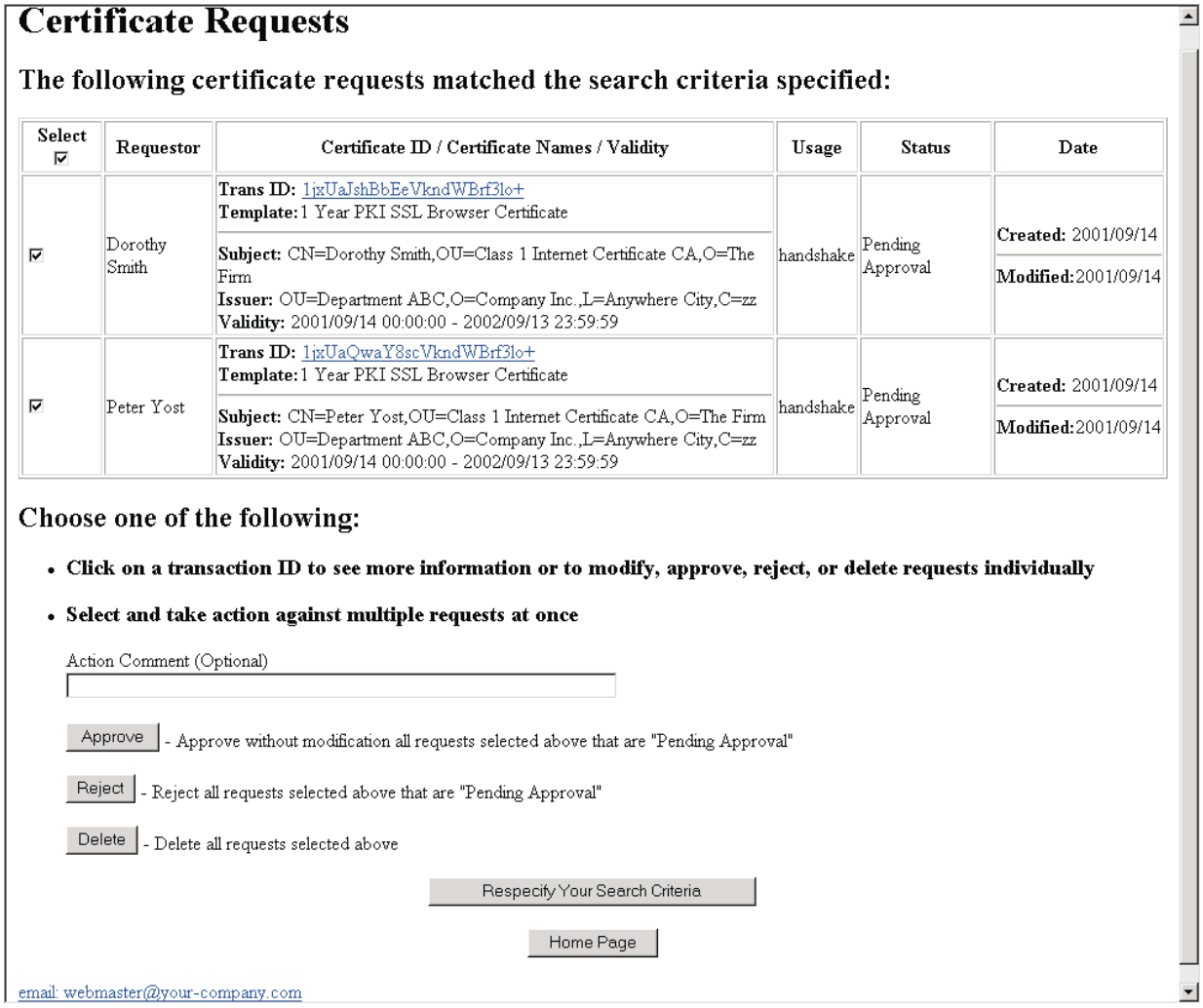
The administrator can use the Web page to search for certificate requests of various statuses. The following table summarizes the searches listed on the Web page and the certificate requests that are displayed as a result:

Table 43. Searches to display certificate requests

Search criteria	Results
Show all requests	Displays all certificate requests (all statuses: ″Pending Approval,″ ″Approved,″ ″Completed,″ ″Rejected,″ and ″Rejected, User Notified″).
Show requests pending approval	Displays only certificate requests whose status is ″Pending Approval.″
Show approved requests	Displays certificate requests whose status is ″Approved″ or ″Completed.″
Show completed requests	Displays certificate requests whose status is ″Completed.″
Show all rejected requests	Displays certificate requests whose status is ″Rejected″ or ″Rejected, User Notified.″
Show Rejections in which the client has been notified	Displays certificate requests whose status is ″Rejected, User Notified.″

To process requests by performing a search for requests of a particular status, perform the following steps:

1. On the PKI Services administration home page (see Figure 16 on page 147), select one of the searches by clicking the appropriate radio button under ″Certificate Requests.″ (The preceding table describes these searches.) You can optionally fill in additional search criteria (″Requestor's name″ and ″Show recent activity only″).
2. Click **Find certificates or certificate requests** button. This displays the following Web page:



Certificate Requests

The following certificate requests matched the search criteria specified:

Select	Requestor	Certificate ID / Certificate Names / Validity	Usage	Status	Date
☑	Dorothy Smith	**Trans ID:** 1jxUaJshBbEeVkndWBrf3lo+ **Template:** 1 Year PKI SSL Browser Certificate **Subject:** CN=Dorothy Smith,OU=Class 1 Internet Certificate CA,O=The Firm **Issuer:** OU=Department ABC,O=Company Inc.,L=Anywhere City,C=zz **Validity:** 2001/09/14 00:00:00 - 2002/09/13 23:59:59	handshake	Pending Approval	**Created:** 2001/09/14 **Modified:** 2001/09/14
☑	Peter Yost	**Trans ID:** 1jxUaQwaY8scVkndWBrf3lo+ **Template:** 1 Year PKI SSL Browser Certificate **Subject:** CN=Peter Yost,OU=Class 1 Internet Certificate CA,O=The Firm **Issuer:** OU=Department ABC,O=Company Inc.,L=Anywhere City,C=zz **Validity:** 2001/09/14 00:00:00 - 2002/09/13 23:59:59	handshake	Pending Approval	**Created:** 2001/09/14 **Modified:** 2001/09/14

Choose one of the following:

- **Click on a transaction ID to see more information or to modify, approve, reject, or delete requests individually**
- **Select and take action against multiple requests at once**

Action Comment (Optional)

Approve - Approve without modification all requests selected above that are "Pending Approval"

Reject - Reject all requests selected above that are "Pending Approval"

Delete - Delete all requests selected above

Respecify Your Search Criteria

Home Page

email: webmaster@your-company.com

Figure 20. Processing requests after searching

Note: The table at the top of the Web page shows the certificate requests that match your search criteria. (If multiple certificates requests match the search criteria, up to ten appear on a Web page, and a button at the bottom of the Web age allows you to view the next set.)

---

3. You can use this Web page:
   - To process a single certificate request
   - To perform the same action on all of the certificate requests that are listed
   - To process selected requests

   To process a single certificate request:

   a. Click on its transaction ID in the table at the top of the Web page. This transfers you to the single request Web page; see Figure 17 on page 148.
   b. From the single request Web page, you can perform the steps in the preceding section, starting with Step 2 on page 148).

   To perform the same action on all the certificate requests that are listed:

   a. Optionally enter a comment.

b. Click one of the action buttons below the comment field to perform that action on all listed requests:

Approve Approves without modification all requests that are pending approval.
Reject Rejects all requests that are pending approval.
Delete Deletes all requests.

Note: The **Approve** and **Reject** buttons appear only if certificate requests are pending approval. Otherwise, only the **Delete** button appears.

To process selected certificate requests:

a. Uncheck the check box beside the **Select** column header. (When the check box beside **Select** is checked, all the individual check boxes in the body of the table are checked. This means all these certificate requests are selected. Unchecking the box in the header unchecks all the boxes in the body of the table.)

b. Check the check boxes of all the certificate requests for which you want to perform a particular action.

c. Optionally enter a comment.

d. Click one of the action buttons below the comment field to perform that action on all listed requests. The action buttons include the following:

Approve Approves without modification all requests that are pending approval.
Reject Rejects all requests that are pending approval.
Delete Deletes all requests.

Note: The **Approve** and **Reject** buttons appear only if certificate requests are pending approval. Otherwise, only the **Delete** button appears.

Tip: If you select the **Show all requests** radio button (see Figure 16 on page 147) and click the **Approve** button on this Web page, only the certificate requests whose status is ″Pending Approval″ are approved.

Instead of processing one or more certificate requests, you can click the **Respecify your search criteria Web page** button to return to the PKI Services administration home page (see Figure 16 on page 147) or the **Home page** button to return to the PKI Services home page (see Figure 13 on page 142).

---

4. After you click an action button, the next Web page is one of the following:
   - Processing successful (see Figure 21 on page 155)
   - Processing was not successful (see Figure 22 on page 155)
   - Processing partially successful (see Figure 23 on page 156)

   If ″Processing was not successful,″ you can click on the transaction ID to display the 'Single Request' Web page; see Figure 17 on page 148. Processing can be unsuccessful because requests do not have the status required for the action you selected; see Table 42 on page 146.

   If you get ″Processing partially successful,″ you can click on the transaction ID to display the 'Single Request' Web page; see Figure 17 on page 148. This message can occur when your organization has more than one administrator and it involves the following sequence:

   – One administrator performs a search

- Another administrator performs a search before the first administrator has approved requests displayed in the search results
- One of the administrators approves only some of the requests
- The other administrator tries to approve requests including at least one the preceding administrator has already approved and one that the preceding administrator has not already approved.

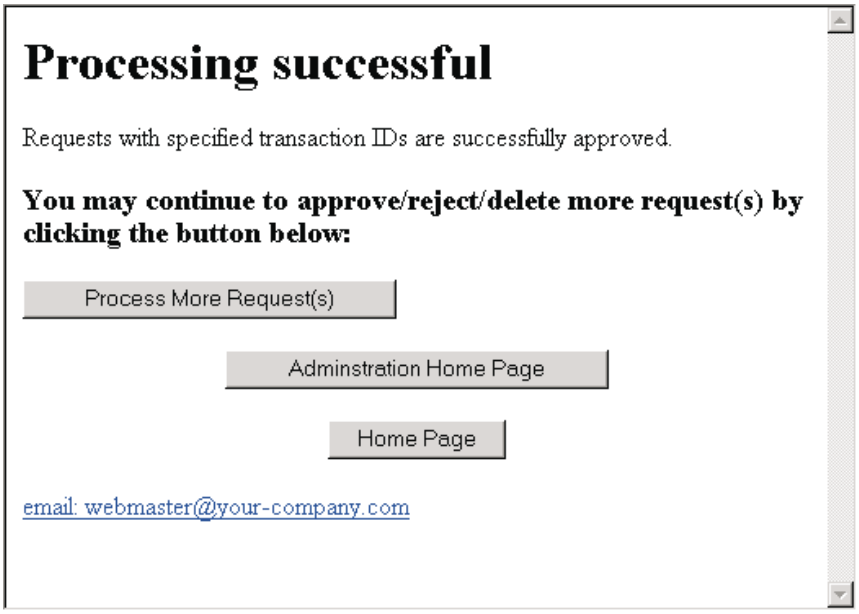


Figure 21. Request processing was successful Web page

Figure 22. Request processing was not successful Web page

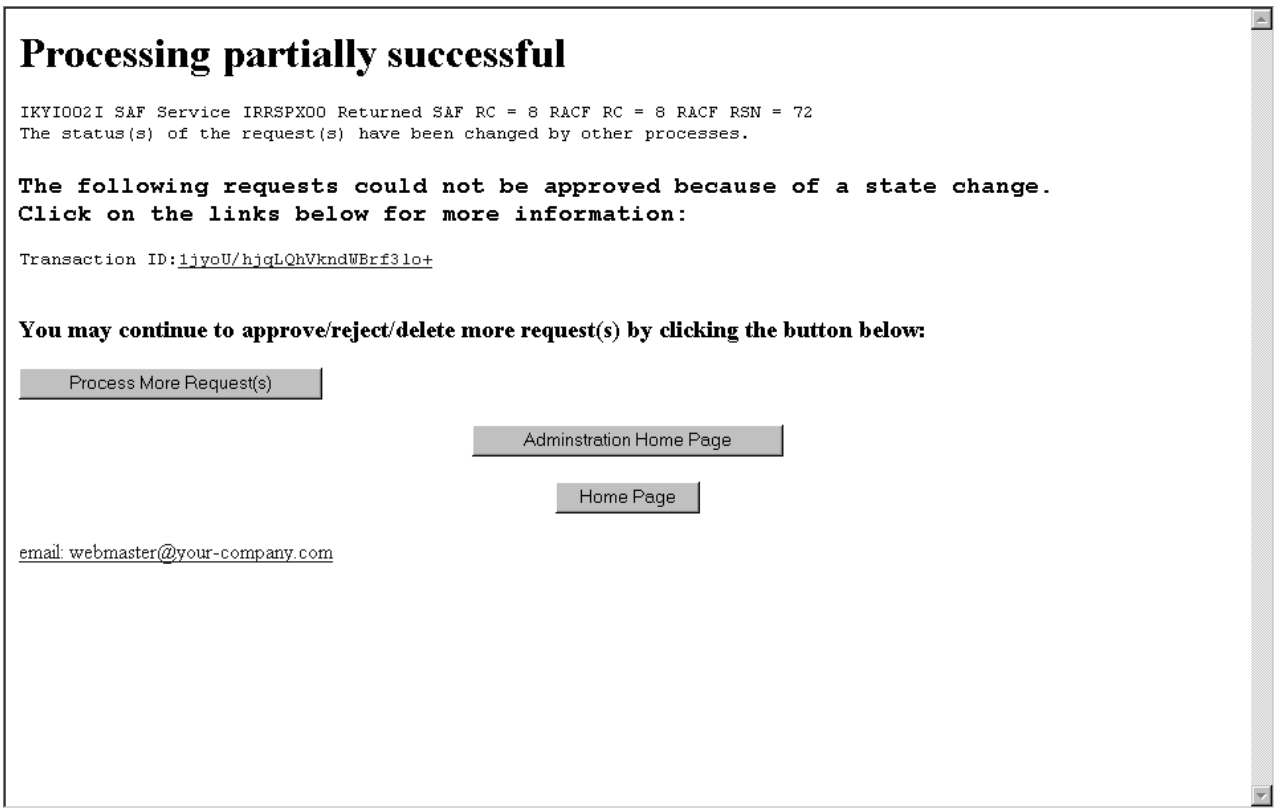


Figure 23. Request processing was partially successful Web page

5. After approving requests as appropriate, you can:
   - Click **Process more request(s)** to return to Figure 20 on page 153
   - Click **Administration home page** to return to Figure 13 on page 142
   - Click **Home page** to return to Figure 16 on page 147.

Processing certificates

Before you can use the Web page to process certificates, you need to understand the statuses of certificates and actions you can perform on certificates.

Status of certificates

Certificates that have been created from requests are maintained permanently in an issued certificate database. Another name for this is the issued certificate list (ICL). Issued certificates are also published in an LDAP directory.

A certificate can have only one of the following states (statuses) at a time:

Table 44. Status of certificates

Active	The certificate has not yet expired and has not been revoked.
Expired	The certificate has not been revoked but has expired.
Revoked	The certificate has not expired but it has been revoked. Such certificates are published on the next certificate revocation list (CRL).

Table 44. Status of certificates (continued)

Revoked, Expired	The certificate was revoked and time has elapsed such that it has expired too. Such certificates would not be published on the next CRL.

The administrator must approve a request for the certificate to have a status (as enumerated in the preceding list) or for the administrator to delete the certificate from the ICL. (An administrator can delete a certificate from the ICL, but this would not be a normal situation.) Alternately, the administrator can reject a request or delete the request from the request database (RDB). If the administrator does not approve the request, it is never listed in the ICL.

Actions for certificates

The following table summarizes actions on certificates and the required status to perform these actions:

Table 45. Summary of actions to perform and required status to do so

Action	Required status of certificate	Who performs action
Renew	″Active″	End user
Revoke	″Active″	End user or administrator
Delete	All (″Active,″ ″Expired,″ ″Revoked,″ or ″Revoked, Expired″)	Administrator

Steps for processing a single certificate

To process a single certificate, perform the following steps:

1. On the PKI Services administration home page (see Figure 16 on page 147), enter the serial number of the certificate you want to process in the field provided for it. This displays the following Web page:

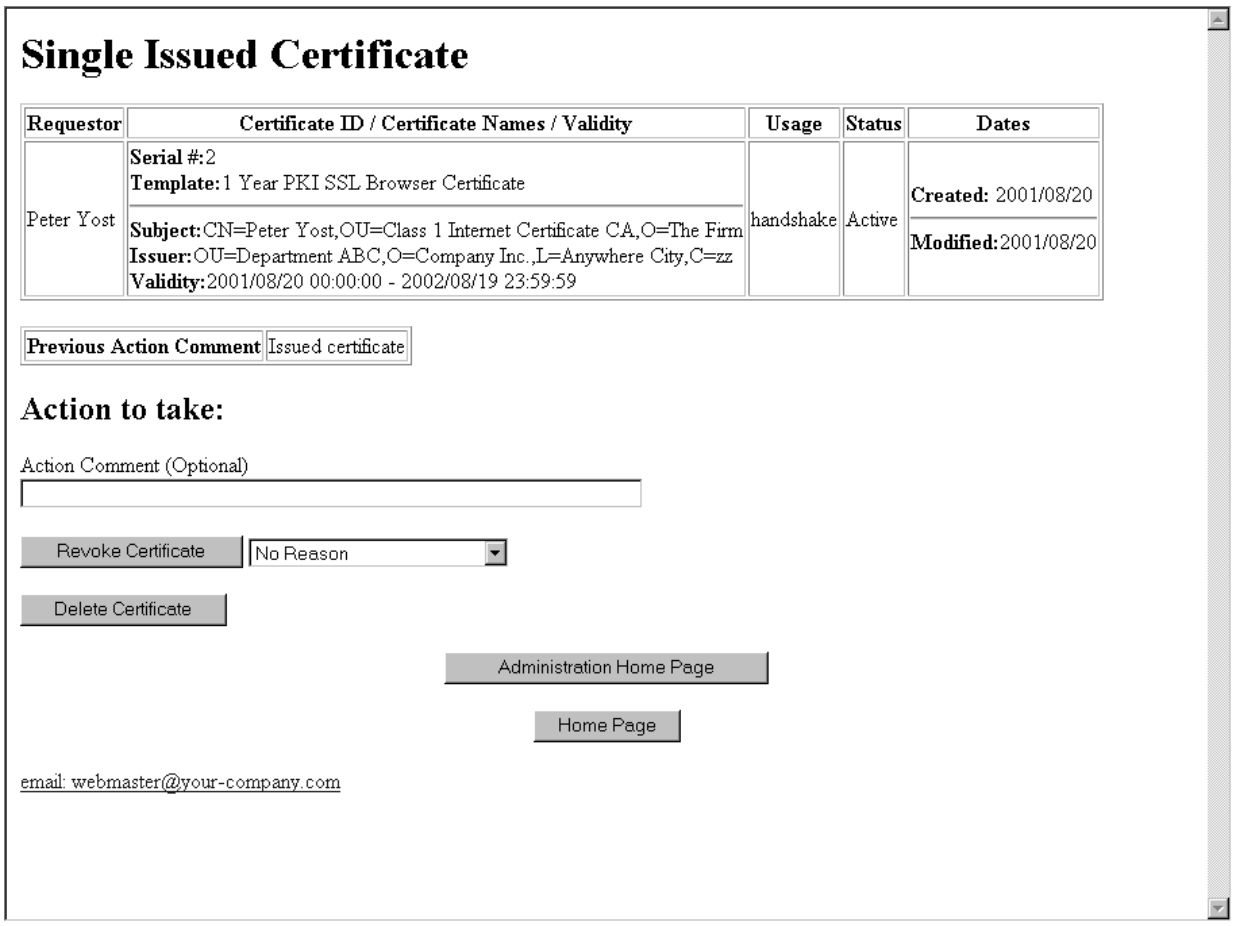


Figure 24. Processing a certificate from the single certificate Web page

2. Make sure the certificate is the correct one by reviewing the information in the top part of the Web page.

3. If you are going to process a certificate from this Web page, you can optionally insert a comment.

4. Click one of the following buttons to process the certificate:

Revoke certificate	Revokes the certificate.
Delete certificate	Deletes the certificate. (This is for cleanup purposes.)

   Note: The **Revoke** button appears only if the status of the certificate is Active.

Steps for processing certificates by performing searches

The administrator can use the Web page to search for certificates of various statuses. The following table summarizes the searches listed on the Web page and the certificates that are displayed as a result:

Table 46. Searches to display certificates

Searches	Results
Show all issued certificates	Displays all certificates (can be any status — ″Active,″ ″Expired,″ ″Revoked,″ or ″Revoked, Expired″).
Show all revoked certificates	Displays certificate requests whose status is ″Revoked″ or ″Revoked, Expired.″
Show all expired certificates	Displays certificates whose status is ″Expired″ or ″Revoked, Expired.″
Show non-expired, non-revoked certificates only	Displays certificates whose status is ″Active.″
Show non-expired certificate revocations only	Displays certificate requests whose status is ″Revoked.″

To process certificates by performing a search for certificates of a particular status, perform the following steps:

1. On the PKI Services administration home page (see Figure 16 on page 147), select one of the searches by clicking the appropriate radio button under ″Issued Certificates″. (The preceding table describes these searches.) You can optionally fill in additional search criteria (″Requestor's name″ and ″Show recent activity only″).

   _____

2. Click the **Find certificates or certificate requests** button. This displays the following Web page.

Issued Certificates

The following issued certificates matched the search criteria specified:

Select ☑	Requestor	Certificate Names / Validity	Usage	Status	Date
☑	Peter Yost	**Serial #:** 3 **Template:** 1 Year PKI SSL Browser Certificate **Subject:** CN=Peter Yost,OU=Class 1 Internet Certificate CA,O=The Firm **Issuer:** OU=Department ABC,O=Company Inc.,L=Anywhere City,C=zz **Validity:** 2001/12/06 00:00:00 - 2002/12/05 23:59:59	handshake	Active	**Created:** 2001/12/06 **Modified:** 2001/12/06
☑	Dorothy Smith	**Serial #:** 4 **Template:** 1 Year PKI SSL Browser Certificate **Subject:** CN=Dorothy Smith,OU=Class 1 Internet Certificate CA,O=The Firm **Issuer:** OU=Department ABC,O=Company Inc.,L=Anywhere City,C=zz **Validity:** 2001/12/06 00:00:00 - 2002/12/05 23:59:59	handshake	Active	**Created:** 2001/12/06 **Modified:** 2001/12/06

Choose one of the following:

- **Click on a serial number to see more information or to revoke or delete certificates individually**
- **Select and take action against multiple requests at once**

Action Comment (Optional)

Revoke All selected Active Certificates | No Reason

Delete All selected Certificates

Respecify Your Search Criteria

Home Page

email: webmaster@your-company.com

Figure 25. Processing certificates using searches

Note: The table at the top of the Web page shows the certificates that match your search criteria. (If multiple certificates match the search criteria, up to ten appear on a Web page, and a button at the bottom of the Web page allows you to view the next set.)

---

3. You can use this Web page:
   - To process a single certificate
   - To perform the same action on all of the certificates that are listed
   - To process selected certificates

   To process a single certificates:

   a. Click on its serial number in the table at the top of the Web page. This transfers you to the single certificate Web page; see Figure 24 on page 158.
   b. From the single certificate Web page, you can perform the steps in the preceding section, starting with Step 2 on page 158).

   To perform the same action on all the certificates that are listed:

   a. Optionally enter a comment.
   b. Click one of the action buttons below the comment field to perform that action on all listed certificates:

Revoke all selected active certificates	Revokes the certificates.
Delete all selected certificates	Deletes the certificates.

Note: For the **Revoke** button to appear, your search must match at least one certificate whose status is active.

To process selected certificates:

a. Uncheck the check box beside the **Select** column header. (When the check box beside **Select** is checked, all the individual check boxes in the body of the table are checked. This means all these certificates are selected. Unchecking the box in the header unchecks all the boxes in the body of the table.)

b. Check the check boxes of all the certificates for which you want to perform a particular action.

c. Optionally enter a comment.

d. Click one of the action buttons below the comment field to perform that action on all listed requests. The action buttons include:

Revoke all selected active certificates	Revokes the certificates.
Delete all selected certificates	Deletes the certificates.

Note: For the **Revoke** button to appear, your search must match at least one certificate whose status is active.

Instead of processing one or more certificates, you can click the **Respecify your search criteria Web page** button to return to the PKI Services administration home page (see Figure 16 on page 147) or the **Home page** button to return to the PKI Services home page (see Figure 13 on page 142).

---

4. After you click an action button, the next Web page tells you:
   - ″Processing was successful″ (see Figure 26 on page 162)
   - ″Processing was not successful″ (see Figure 27 on page 162)
   - ″Processing partially successful″ (see Figure 28 on page 163)

   If ″Processing was not successful,″ you can click on a serial number to display the ″Single Certificate″ Web page; see Figure 24 on page 158. Processing can be unsuccessful because certificates do not have the status required for the action you selected; see Table 45 on page 157.

   If you get ″Processing partially successful,″ you can click on the serial number to display the ″Single Certificate″ Web page; see Figure 24 on page 158. The ″Processing partially successful″ message can occur when your organization has more than one administrator and it involves the following sequence:
   - One administrator performs a search
   - Another administrator performs a search before the first administrator has revoked or deleted certificates displayed in the search results
   - One of the administrators revokes or deletes some of the certificates
   - The other administrator tries to revoke or delete certificates including at least one the preceding administrator has already revoked or deleted and at least one the preceding administrator has not already revoked or deleted.

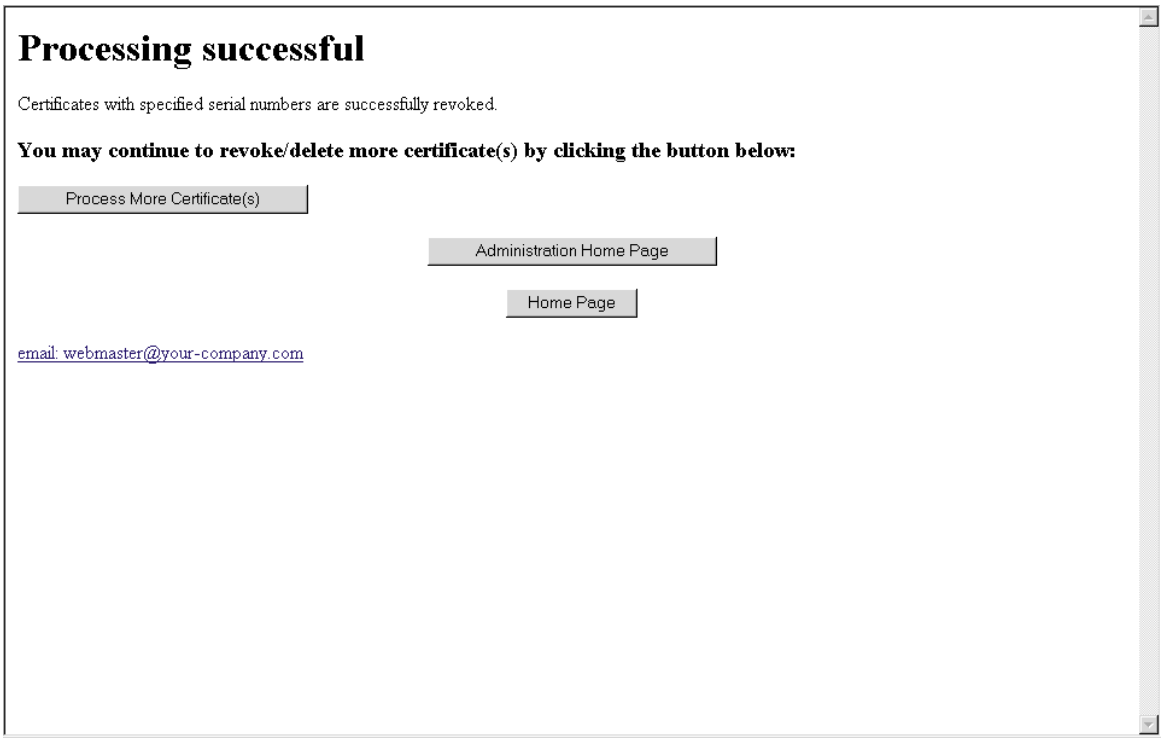


Figure 26. Processing of certificate was successful Web page

Figure 27. Request processing was not successful Web page

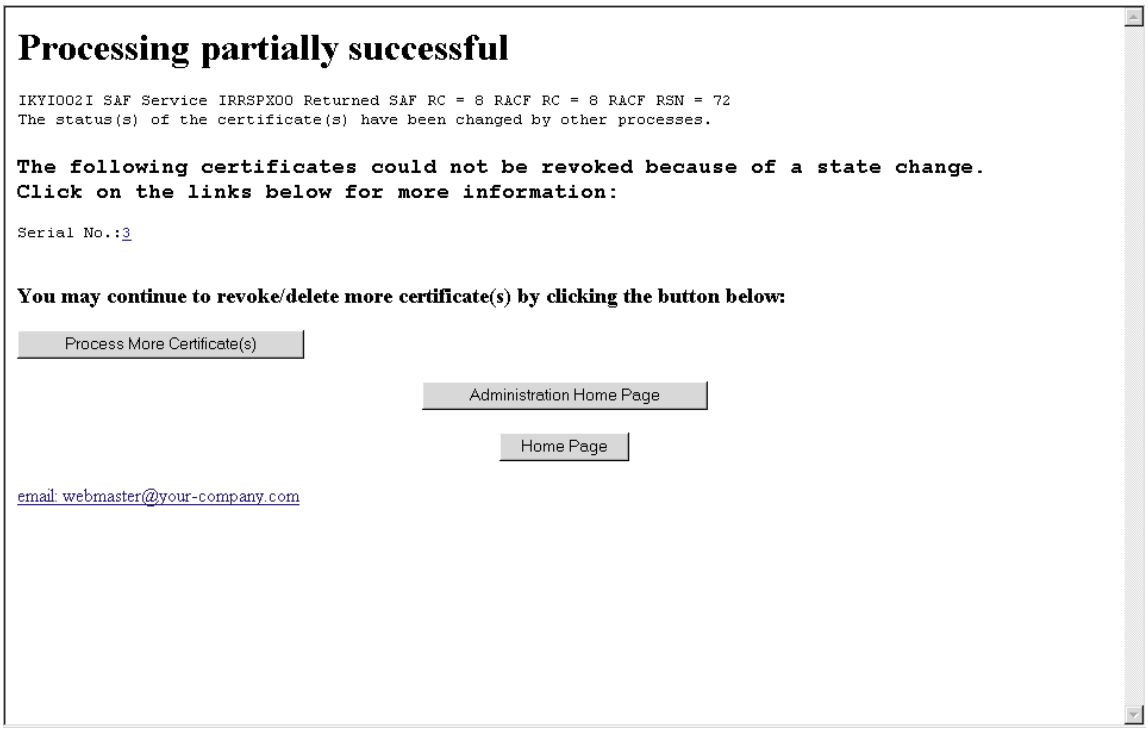


Figure 28. Request processing was partially successful Web page

You can click the **Home page** button to return you to the PKI Services home page (see Figure 13 on page 142).

Relationship between certificate requests and matching certificates

PKI Services maintains two databases:

- The request database (RDB) also called the ObjectStore
- The Issued Certificate List (ICL)

RDB records are temporary in nature. They exist only to track active requests. PKI Services automatically removes these records when they are complete or go inactive. ICL records are permanent. Requests for certificates (both new and renewal) are stored in the RDB. Once approved, a matching certificate is created from the request and stored in the ICL. (Note, the creation of the certificate may not be instantaneous.) At this point, the two database records, though related, exist independently of each other.

- After a request is approved, there is no way for you to unapprove a request. If you mistakenly approve a request that you meant to reject, you should immediately delete the RDB entry. This prevents the user from retrieving the certificate. You should then search the issued certificates to see if the certificate has been issued. If it has, you should revoke it in case the user has already picked it up.
- Revoking a certificate (an ICL action) has no effect on its matching RDB entry. If you revoke a certificate, you should also delete its matching RDB entry if it exists. This prevents the user from retrieving the certificate, if the user has not already done so.
- You can delete RDB entries any time after they have been completed to save space in the database if desired.

- Under normal circumstances, ICL entries should not be deleted. If you delete an ICL entry, you will no longer be able to revoke or renew the certificate.
- You can delete entries in any state in either database to clean up error conditions.

Part 5. Administering RACF for PKI Services

This part of the book explains how to perform the following tasks:

- Authorizing users for the PKI Services administration group (connecting and deleting members)
- Authorizing users for inquiry access
- Administering HostIdMappings extensions
- Locating your PKI Services certificate and key ring
- Establishing PKI Services as an intermediate certificate authority
- Renewing your PKI Services certificate authority certificate
- Recovering a CA certificate profile
- Controlling applications that call R_PKIServ

Chapter 15. RACF administration for PKI Services

This chapter describes the tasks that the RACF administrator performs after PKI Services has been set up and customized.

The following topics are covered:

For more information about the RACF commands shown in this chapter, see *z/OS Security Server RACF Command Language Reference*.

Authorizing users for the PKI Services administration group

You need to know how to add and delete members from the PKI Services administration group (by default, PKIGRP).

Connecting members to the group

The PKI Services administration group is a RACF group containing the list of user IDs that are authorized to use PKI Services administration functions. To connect a member to the group, enter the following command, replacing *pkigroup_mem* with the member's user ID and *pkigroup* with the name of the PKI Services administration group (**PKIGRP** by default). (See Table 17 on page 32 for more information.)

```
CONNECT pkigroup_mem GROUP(pkigroup)
```

Note: You need to enter this command for each user ID in turn.

Deleting members from groups

To remove a user from a group, enter the following command, replacing *pkigroup_mem* with the user ID of the member you want to delete and *pkigroup* with the name of the PKI Services administration group (**PKIGRP** by default).

```
REMOVE pkigroup_mem GROUP(pkigroup)
```

Authorizing users for inquiry access

You can add groups of users who do not need the full administrative authority of users in the PKIGRP group. The following procedure can be used to authorize a new group for inquiry abilities, such as a help desk might require. The commands shown include variables whose names are appropriate for this scenario.

Steps for authorizing users for inquiry access

Before you begin: You need to know the high-level VSAM data set qualifier used for the IKYSETUP variable *vsamhlq* value, in case your installation did not use the `PKISRVD` default. (See Table 17 on page 32.)

RACF administration for PKI Services

Perform the following steps to add and administer a group that needs authority to query PKI Services information.

1. Add the new group.

   Example:

   ```
   ADDGROUP HELPDESK OMVS(GID(197312))
   ```

   ---

2. Connect each member to the new group. Repeat for each user ID you need to connect.

   Example:

   ```
   CONNECT OPER17 GROUP(HELPDESK)
   ```

   ---

3. Authorize the new group for READ access to the resources of PKI Services. Replace your installation's value for the data set's high-level qualifier if your installation did not use the `PKISRVD` default.

   Example:

   ```
   PERMIT 'PKISRVD.**' ID(HELPDESK) ACCESS(READ)
   PERMIT IRR.RPKISERV.PKIADMIN CLASS(FACILITY) ID(HELPDESK) ACCESS(READ)
   SETROPTS GENERIC(DATASET) REFRESH
   SETROPTS RACLIST(FACILITY) REFRESH
   ```

   The SETROPTS commands activate the profiles that authorize READ access.

   ---

4. If necessary, you can remove a user from the group. The following example removes the user you connected in Step 2.

   Example:

   ```
   REMOVE OPER17 GROUP(HELPDESK)
   ```

   ---

5. If necessary, you can delete the group. The following example deletes the group you created in Step 1.

   Example:

   ```
   DELGROUP(HELPDESK)
   ```

Administering HostIdMappings extensions

You can add a `HostIdMappings` extension to certificates you create for certain users, allowing you to specify the user IDs that each user will be able to use for login to particular servers (or hosts). Controlling an identity used for login purposes is a very important security objective. Therefore, you must exercise administrative control in the following areas by authorizing:

- PKI Services as a highly trusted certificate authority whose certificates will be honored when they contain `HostIdMappings` extensions
- Particular servers to accept logins from clients whose certificates contain `HostIdMappings` extensions

Steps for administering HostIdMappings extensions

Perform the following steps to allow the Web server to accept logins from clients who have been issued PKI Services certificates with HostIdMappings extensions:

1. Determine if PKI Services is defined as a highly trusted certificate authority on your system by listing its certificate authority definition by using the RACDCERT CERTAUTH LIST command.

   Example:

   ```
   RACDCERT CERTAUTH LIST(LABEL('Local PKI CA'))
   ```

   Check the `Status` information near the top of the output listing for the `HIGHTRUST` attribute.

   ---

2. If not already defined, add the `HIGHTRUST` attribute to the certificate authority definition for PKI Services.

   Example:

   ```
   RACDCERT CERTAUTH ALTER(LABEL('Local PKI CA')) HIGHTRUST
   ```

   ---

3. Define a resource in the SERVAUTH class for each server (host) name you want your Web server to honor when accepting logins for certificates containing HostIdMappings extensions. The resource name follows the format: `IRR.HOST.`*`hostname`*. The *hostname* is the value of the HostIdMappings extension entry pertaining to the z/OS host system you are administering (without the subject ID portion). This is usually a domain name, such as plpsc.pok.ibm.com. The following example shows defining a resource.

   Example:

   ```
   RDEFINE SERVAUTH IRR.HOST.PLPSC.POK.IBM.COM UACC(NONE)
   ```

   ---

4. Permit your Web server to access this resource with READ authority. Be sure the Web server is defined as a RACF user.

   Example:

   ```
   PERMIT IRR.HOST.PLPSC.POK.IBM.COM CLASS(SERVAUTH) ID(WEBSRV) ACCESS(READ)
   ```

   ---

5. Activate the SERVAUTH class, if not already active.

   Example:

   ```
   SETROPTS CLASSACT(SERVAUTH)
   ```

   If already active, refresh the SERVAUTH class.

   Example:

   ```
   SETROPTS CLASSACT(SERVAUTH) REFRESH
   ```

Note: On a z/OS system, a HostIdMapping is not honored if the target user ID was created after the start of the validity period for the certificate containing the HostIdMappings extension. Therefore, if you are creating user IDs specifically for certificates with HostIdMappings extensions, make sure that you create the user IDs before the certificate requests are submitted.

Alternately, when approving the certificate, you can modify the date the certificate becomes valid so that it is not earlier than the date the user ID was created . For renewed certificates, all the original information is replicated in the new certificate, including the date the certificate becomes valid and any HostIdMappings. If you wish to change a HostIdMapping when approving the renewed certificate, you must also modify the date the certificate becomes valid so that it is not earlier than the date the user ID was created.

See *z/OS Security Server RACF Command Language Reference* for details about syntax and authorization required for using the RACDCERT command.

Locating your PKI Services certificate and key ring

The IKYSETUP exec sets up the RACF environment for PKI Services. After the set up is complete, you may need to go back and locate the PKI Services certificate or key ring, possibly to diagnose error conditions. You can do this by using various RACF TSO commands.

Before you begin: You need to determine the following setup information:

Table 47. Information you need for locating your PKI Services certificate and key ring

Information needed	Where to find this information	Record your value here
ca_label - Label of your CA certificate in RACF	See Table 11 on page 25.	
ca_ring - PKI Services SAF key ring	See Table 17 on page 32.	
daemon - User ID for the PKI daemon	See Table 17 on page 32.	
log_dsn - Data set name of the IKYSETUP log	See Table 17 on page 32.	
export_dsn - Data set name of your CA certificate as exported from RACF	See Table 17 on page 32.	

Steps for locating the PKI Services certificate and key ring

Perform the following steps to locate the PKI Services certificate and key ring:

1. Locate the certificate by using one of the following two commands. To locate the certificate in RACF, using the export data set containing the certificate as saved by IKYSETUP, enter the following RACF command from a TSO command prompt:

   ```
   RACDCERT CHECKCERT(export_dsn)
   ```

   The output should be something like the following:

   ```
   Digital certificate information for CERTAUTH:
     Label: Local PKI CA
     Certificate ID: 2QiJmZmDhZmjgdOWg4GTQNfSyUDDwUBA
     Status: HIGHTRUST
     Start Date: 2001/06/04 23:00:00
     End Date:   2020/01/01 22:59:59
     Serial Number:
          >00<
     Issuer's Name:
          >OU=Human Resources Certificate Authority.O=IBM.C=US<
     Subject's Name:
   ```

```
     >OU=Human Resources Certificate Authority.O=IBM.C=US<
  Key Usage: CERTSIGN
  Private Key Type: Non-ICSF
  Private Key Size: 1024
```

It is important to note the following if diagnosing errors:

- The first line must indicate that this is a CERTAUTH certificate.
- The Label must match your ca_label value (as in the preceding table).
- The Subject's Name must match the original value recorded for the PKI Services SUBJECTSDN in the IKYSETUP log.
- The Private Key Type and Size must be present.
- If the Issuer's Name differs from the Subject's Name, this indicates that the certificate was issued by another certificate authority.
- If the Serial Number is not equal to 00, this indicates that the certificate has been renewed or was issued by another certificate authority.

Alternately, you can locate the certificate directly by entering the following RACF command from a TSO command prompt:

```
RACDCERT CERTAUTH LIST(LABEL('ca_label'))
```

This should produce the same information as the preceding. In addition, any ring associations are also displayed:

```
Ring Associations:
  Ring Owner: PKISRVD
  Ring:
     >CAring<
```

In this information, you should ensure that one of the associations listed has the daemon user ID as the owner and that the ring name matches your ca_ring value (as listed in the preceding table).

---

2. Examine the CA key ring. To do so, from a TSO command prompt, enter the following RACF command:

   ```
   RACDCERT ID(daemon) LISTRING(ca_ring)
   ```

   This command should produce information such as the following:

   ```
   Digital ring information for user PKISRVD:

     Ring:
          >CAring<
     Certificate Label Name             Cert Owner     USAGE      DEFAULT
     --------------------------------   ------------   --------   -------
     Local PKI CA                       CERTAUTH       PERSONAL     YES
   ```

   The entry for the PKI Services CA certificate must have USAGE PERSONAL and DEFAULT YES.

Establishing PKI Services as an intermediate certificate authority

The default setup for PKI Services establishes the PKI Services certificate authority as a root CA, also known as a self-signed CA. Because there is no established trust hierarchy leading to a self-signed certificate, it is impossible to verify that a self-signed certificate is genuine. Accordingly, any person or application that wishes to process certificates issued by a root authority must explicitly trust the authenticity of the self-signed CA certificate.

Alternately, you can establish the PKI Services certificate authority as a intermediate (subordinate) certificate authority. An intermediate certificate authority is one whose certificate is signed by another higher certificate authority. This higher certificate authority may be a root CA or another intermediate CA. If the root CA certificate has previously been trusted, any lower intermediate CA certificate can be verified using the higher certificate.

Steps for establishing PKI Services as an intermediate CA

Before you begin: The commands in the steps that follow include several variables. The following table describes these variables. Determine the values for these variables and record the information on the blank lines:

Table 48. Information you need for establishing PKI Services as an intermediate CA

Information needed	Where to find this information	Information values
ca_label - This is the label of your CA certificate in RACF	See Table 11 on page 25.	_____
cert_dsn - This is name of the data set to contain your new certificate request and returned certificate.	You decide this based on local data set naming conventions.	_____
export_dsn - This is the data set name of your CA certificate as exported from RACF.	See Table 17 on page 32.	_____

Perform the following steps to establish PKI Services as an intermediate certificate authority:

1. If you have not yet configured your system for PKI Services, then perform all required steps in Chapter 4, “Running IKYSETUP to perform RACF administration” on page 23.

   _____

2. Determine what certificate authority will be acting as a higher authority for PKI Services. (This could be a public certificate authority, such as VeriSign, or a local, internal certificate authority, perhaps even another instance of PKI Services.)

   _____

3. Create a new certificate request from your self-signed CA certificate by entering the following RACF command from a TSO command prompt:

   ```
   RACDCERT CERTAUTH GENREQ(LABEL('ca_label')) DSN(cert_dsn)
   ```

   _____

4. Send the certificate request to the higher certificate authority, following the procedures that higher authority requires.

   _____

5. After the certificate has been issued, receive the certificate back into the certificate data set (*cert_dsn*).

   Note: The procedure for doing this can vary greatly depending on how the higher certificate authority delivers the new certificate:

   - If the certificate is delivered as base64 encoded text, the easiest way to deposit the certificate into the data set is to edit the certificate data set:
     a. Delete all existing lines in *cert_dsn*.
     b. Copy the base64 encoded text.
     c. Paste this into the ISPF edit window.
     d. Save.
   - If the certificate is delivered as binary data (also called DER encoded), the easiest way to deposit the certificate into the data set is to use binary FTP.

   _____

6. Receive the certificate back into the RACF data base by entering the following RACF command from a TSO command prompt:

   ```
   RACDCERT CERTAUTH ADD(cert_dsn)
   ```

   _____

7. Export the certificate in DER format to the export data set by entering the following RACF command from a TSO command prompt:

   ```
   RACDCERT CERTAUTH EXPORT(LABEL('ca_label')) DSN(export_dsn) FORMAT(CERTDER)
   ```

   _____

8. To make your new certificate available to your clients, set up the /var/pkiserv/directory by performing step 2 on page 48 through step 4 on page 48 in “Steps for setting up the /var/pkiserv directory” on page 47.

   _____

Renewing your PKI Services certificate authority certificate

Eventually, your PKI Services CA certificate will expire. To avoid complications related to an expired CA certificate, you should renew the certificate before it actually expires.

Note: You will receive MVS console message IKYP026E as the expiration date approaches.

Steps for renewing your PKI Services certificate authority certificate

Before you begin: The commands in the steps that follow include several variables. The following table describes these variables. Determine the values for these variables and record the information on the blank lines:

Table 49. Information you need for renewing your PKI Services certificate authority certificate

Information needed	Where to find this information	Information values
ca_label - This is the label of your CA certificate in RACF	See Table 11 on page 25.	_____

Table 49. Information you need for renewing your PKI Services certificate authority certificate (continued)

Information needed	Where to find this information	Information values
cert_dsn - This is name of the data set to contain your new certificate request and returned certificate.	You decide this based on local data set naming conventions.	_____
export_dsn - This is the data set name of your CA certificate as exported from RACF.	See Table 17 on page 32.	_____

Perform the following steps to renew your PKI Services CA certificate:

1. Create a new certificate request from your self-signed CA certificate by entering the following RACF command from a TSO command prompt:

   ```
   RACDCERT CERTAUTH GENREQ(LABEL('ca_label')) DSN(cert_dsn)
   ```

   _____

2. If your PKI Services certificate authority is a root CA (that is, it has a self-signed certificate, which is the default), then generate the self-signed renewal certificate by entering the following RACF command from a TSO command prompt:

   ```
   RACDCERT CERTAUTH GENCERT(cert_dsn) SIGNWITH(CERTAUTH LABEL('ca_label'))
   ```

   _____

3. Alternately, if your PKI Services certificate authority is an intermediate certificate authority, perform the following steps:
   a. Send the certificate request to the higher certificate authority, following the procedures that higher authority requires.
   b. After the certificate has been issued, receive the certificate back into the certificate data set (*cert_dsn*).

      Note: The procedure for doing this can vary greatly depending on how the higher certificate authority delivers the new certificate:

      - If the certificate is delivered as base64 encoded text, the easiest way to deposit the certificate into the data set is to edit the certificate data set:
        1) Delete all existing lines in *cert_dsn*.
        2) Copy the base64 encoded text.
        3) Paste this into the ISPF edit window.
        4) Save.
      - If the certificate is delivered as binary data (also called DER encoded), the easiest way to deposit the certificate into the data set is to use binary FTP.

   c. Receive the certificate back into the RACF data base by entering the following RACF command from a TSO command prompt:

      ```
      RACDCERT CERTAUTH ADD(cert_dsn)
      ```

   _____

4. Export the certificate in DER format to the export data set by entering the following RACF command from a TSO command prompt:

   ```
   RACDCERT CERTAUTH EXPORT(LABEL('ca_label')) DSN(export_dsn) FORMAT(CERTDER)
   ```

   _____

5. To make your new certificate available to your clients, set up the /var/pkiserv/directory by performing steps 2 on page 48 through 4 on page 48 in "Steps for setting up the /var/pkiserv directory" on page 47.

   _____

Recovering a CA certificate profile

Unless you change the IKYSETUP REXX exec to disable the function, IKYSETUP automatically backs up the PKI Services CA certificate and private key to a passphrase-encrypted data set that has PKCS#12 format. If the CA certificate profile in RACF is accidentally deleted, you can recover it from the backup data set.

Steps for recovering a CA certificate profile

Before you begin: The commands in the steps that follow include several variables. The following table describes these variables. Determine the values for these variables and record the information on the blank lines:

Table 50. Information you need for recovering a CA certificate profile

Information needed	Where to find this information	Information values
backup_dsn - The name of the data set containing the backup copy of your private key.	See Table 17 on page 32.	_____
ca_label - This is the label of your CA certificate in RACF	See Table 11 on page 25.	_____
ca_ring - The PKI Services SAF key ring.	See Table 17 on page 32.	_____
cert_dsn - This is name of the data set to contain your new certificate request and returned certificate.	You decide this based on local data set naming conventions.	_____
daemon - The user ID for the PKI daemon.	See Table 17 on page 32.	_____
export_dsn - This is the data set name of your CA certificate as exported from RACF.	See Table 17 on page 32.	_____
your-passphase - The pass phrase you used when backing up the private key.	You specified this when running IKYSETUP.	_____

Perform the following steps to recover a CA certificate profile:

1. Enter the following TSO commands:

```
RACDCERT CERTAUTH ADD(backup_dsn) PASSWORD(your-passphrase)
     WITHLABEL('ca_label') ICSF
RACDCERT CERTAUTH ADD(export_dsn)
RACDCERT ID(daemon) CONNECT(CERTAUTH LABEL('ca_label')
     RING(ca_ring) USAGE(PERSONAL) DEFAULT)
```

   Note: If you are not using ICSF, omit the ICSF keyword on the ADD.

   _____

2. Perform the following steps to update the RACF profile with the serial number of the last certificate PKI Services issued. (You need to restore the certificate serial number incrementer value that is stored in the profile because otherwise, PKI Services resumes issuing certificates starting from serial number 1.)

a. Make sure PKI Services is stopped. (See “Stopping the PKI Services daemon” on page 62 for details on how to do this.)

b. Enter the following command from the UNIX command line to run the iclview utility:

```
iclview \'pkisrvd.vsam.icl\'
```

Record the serial number displayed (in hex) of the last certificate listed:

Serial number (in hex) of last certificate:	

c. To determine your CA certificate’s profile name, issue the following command to perform an *unsuccessful* ADD:

```
RACDCERT CERTAUTH ADD(export_dsn) WITHLABEL('*** Bad Label ***')
```

The unsuccessful ADD displays an error message including the profile name. Record the profile name:

Profile name:	

d. Create the following ICHEINTY ALTER job in your JCL data set, replacing the highlighted values based on the information you recorded in the previous steps:

```
//SAMPIC JOB 'xxxxxxxx',NOTIFY=xxxxxx,
// CLASS=A,MSGCLASS=H,MSGLEVEL=(1,1),
// REGION=4M
//*************************************************************
//ASM EXEC ASMHCL,PARM.C='OBJECT,DECK,TEST',
// PARM.L='MAP,LET,LIST,NCAL,AC(1)'
//C.SYSIN DD *
SAMPICHE CSECT
SAMPICHE AMODE 31
SAMPICHE RMODE ANY
        STM R14,R12,12(R13) Save registers
        BALR R12,0
        USING *,R12 R12 = base register
        B DOIT
*********************************************************************
* Update the following declares with your certificate info          *
*********************************************************************
ENTRY EQU * Your CA certificate profile
ENTBLEN DC H'54' Length of cert profile name
ENTALEN DC H'54' Length of cert profile name
        DC CL43'00.OU=Human¢Resources¢Certificate¢Authority'
        DC CL11'.O=IBM.C=US'
LSER EQU * CERTLSER is an 8 byte field
LSERHIGH DC X'00000000' High word - set to zero
LSERLOW DC X'000000FF' Set to your last serial # (hex)
*********************************************************************
* Establish standard linkage                                        *
*********************************************************************
DOIT ST R13,SAVE+4 Save caller's save area address
     LA R15,SAVE Get the next save area address
     ST R15,8(R13) Link the save areas
     LR R13,R15 R13 points to next save area
     ICHEINTY ALTER,TYPE='GEN',ENTRYX=ENTRY,RELEASE=1.9,               X
           SEGMENT='CERTDATA',CLASS=DIGTCLAS,                          X
           ACTIONS=(A_LSER)
CLOSEUP EQU *
        L R13,SAVE+4 Get caller's save area address
        ST R15,16(R13) Save ICHEINTY RC
        LM R14,R12,12(R13) Restore registers except R13
        BR R14 Back to invoker
*********************************************************************
* CONSTANTS, SAVE AREAS, ETC                                        *
*********************************************************************
SAVE DS 18F
DIGTCLAS DC CL8'DIGTCERT'
         ORG
A_LSER ICHEACTN FIELD=CERTLSER,FLDATA=(8,LSER),RELEASE=1.9,MF=L
*********************************************************************
* General Equates                                                   *
*********************************************************************
R0 EQU 0
R1 EQU 1
R2 EQU 2
R3 EQU 3
R4 EQU 4
R5 EQU 5
R6 EQU 6
R7 EQU 7
R8 EQU 8
```

Figure 29. Sample JCL data set for restoring the certificate serial number incrementer value (Part 1 of 2)

```
R10 EQU 10
R11 EQU 11
R12 EQU 12
R13 EQU 13
R14 EQU 14
R15 EQU 15
    END SAMPICHE
//C.SYSLIB   DD DSN=SYS1.MACLIB,DISP=SHR
//           DD DSN=SYS1.MODGEN,DISP=SHR
//C.SYSPRINT DD SYSOUT=*
//L.SYSLMOD  DD DSN=SYS1.LINKLIB(SAMPICHE),DISP=SHR
//L.SYSPRINT DD SYSOUT=*
//L.SYSIN DD *
 NAME SAMPICHE(R)
/*
//RUNIT EXEC PGM=SAMPICHE
//*
```

Figure 29. Sample JCL data set for restoring the certificate serial number incrementer value (Part 2 of 2)

e. Submit the job and check its return code.

Controlling applications that invoke R_PKIServ

Authorized applications, such as servers, that invoke the `R_PKIServ` callable service (IRRSPX00) can request the generation, retrieval, and administration of PKIX-compliant X.509 Version 3 certificates and certificate requests. Applications can request end-user functions or administrative functions related to these requests. See *z/OS Security Server RACF Callable Services* for details of invoking IRRSPX00.

You authorize these applications by administering RACF profiles in the FACILITY class, based on whether the application requests end-user functions or administrative functions.

R_PKIServ end-user functions

The end-user functions are:

EXPORT Retrieves (exports) a previously requested certificate.

GENCERT Generates an auto-approved certificate.

GENRENEW Generates an auto-approved renewal certificate.

Note: The request submitted is automatically approved.

REQCERT Requests a certificate that an administrator must approve before it is created.

REQRENEW Requests certificate renewal. The administrator needs to approve the request before the certificate is renewed.

REVOKE Revokes a certificate that was previously issued.

VERIFY Confirms that a given user certificate was issued by this CA and, if so, returns the certificate fields.

For end-user functions, FACILITY class profiles protect this interface. The form of the FACILITY class profiles is:

`IRR.RPKISERV.<function>`

<function>

Is one of the following end-user function names in the preceding list. The user ID for the application (user ID from the ACEE associated with the address space) is used to determine access:

NONE Access is denied.

READ Access is permitted based on subsequent access checks against the caller's user ID. To determine the caller, the current TCB is checked for an ACEE. If one is found, the authority of that user is checked. If there is no ACEE associated with the current TCB, the ACEE associated with the address space is used to locate the user ID.

UPDATE Access is permitted based on subsequent access checks against the application's user ID.

CONTROL (or user ID is RACF SPECIAL)
Access is permitted, and no subsequent access checks are made.

For SAF GENCERT and EXPORT requests where the application has READ and UPDATE access, subsequent access checks are performed against the IRR.DIGTCERT.<*function*> FACILITY profiles. These are identical to the checks the RACDCERT TSO command makes. See *z/OS Security Server RACF Command Language Reference* for more information.

For PKI Services EXPORT, GENCERT, GENRENEW, REQCERT, REQRENEW, REVOKE, and VERIFY requests in which the application has READ and UPDATE access, subsequent access checks are performed against the IRR.DIGTCERT.<*function*> FACILITY profiles. The following table summarizes the access requirements for the user ID whose access is checked:

Table 51. Summary of accesses required for PKI Services request

Request	Access
EXPORT	• IRR.DIGTCERT.EXPORT – UPDATE access if no pass phrase is specified on the call – READ access if a pass phrase is specified.
GENCERT	• IRR.DIGTCERT.GENCERT — CONTROL access • IRR.DIGTCERT.ADD – UPDATE access if any HostIdMappings information is specified in the certificate request parameter list or the `UserId` field in the certificate request parameter list indicates the certificate is being requested for another user other than the caller – READ access otherwise
GENRENEW	• IRR.DIGTCERT.GENRENEW — READ access • IRR.DIGTCERT.GENCERT — CONTROL access **Note:** It is assumed that the calling application has already verified the input certificate using the VERIFY function.
REQCERT	• IRR.DIGTCERT.REQCERT — READ access

Table 51. Summary of accesses required for PKI Services request (continued)

Request	Access
REQRENEW	• IRR.DIGTCERT.REQRENEW — READ access **Note:** It is assumed that the calling application has already verified the input certificate using the VERIFY function.
REVOKE	• IRR.DIGTCERT.REVOKE — READ access **Note:** It is assumed that the calling application has already verified the target certificate using the VERIFY function.
VERIFY	• IRR.DIGTCERT.VERIFY — READ access **Note:** It is assumed that the calling application has already verified that the end user possesses the private key that correlates to the input certificate.

R_PKIServ administrative functions

The administrative functions are:

CERTDETAILS	Get detailed information about one PKI Services issued certificate.
MODIFYCERTS	Change PKI Services issued certificates.
MODIFYREQS	Change PKI Services certificate requests.
QUERYCERTS	Query PKI Services issued certificates.
QUERYREQS	Query PKI Services about certificate requests.
REQDETAILS	Get detail information about one PKI Services certificate request.

For the administrative functions, a single FACILITY class profile — IRR.RPKISERV.PKIADMIN — protects this interface:

- If the caller is RACF SPECIAL, no further access is necessary
- Otherwise, the caller needs:
  - READ access to perform read operations (QUERYREQS, QUERYCERTS, REQDETAILS, and CERTDETAILS)
  - UPDATE access for the action operations, (MODIFYREQS and MODIFYCERTS).

To determine the appropriate access level of the caller, the current TCB is checked for an ACEE. If one is found, the authority of that user is checked. If there is no ACEE associated with the current TCB, the ACEE associated with the address space is used to locate the user ID.

Attention: UPDATE access to the IRR.RPKISERV.PKIADMIN resource also controls who can act as PKI Services administrators. PKI Services administrators play a very powerful role in your organization. The decisions they make when managing certificates and certificate requests determine who will access your computer systems and what privileges they will have when doing so.

Recommendation: Give UPDATE authority only to those individuals whom you would trust with the RACF SPECIAL attribute. If you do assign PKI Services administrators who do not have the RACF SPECIAL attribute, do not also give these individuals direct access to the end-user functions of the R_PKIServ callable service as described in the previous section.

Part 6. Troubleshooting

This part of the book explains using logs and utilities, including the following:

- Chapter 16, “Using information from SYS1.LOGREC” on page 183 discusses ’SYS1.LOGREC’ — which is used to record unusual runtime events, such as an exception.
- Chapter 17, “Using information from the PKI Services logs” on page 189 discusses using the PKI Services logs, which are ongoing, to debug problems and explains how to change logging options and display log options settings.
- Chapter 18, “Using PKI Services utilities” on page 195 explains using PKI Services utilities:
  - vosview — displays the entries contained in the VSAM ObjectStore data set (request database)
  - iclview — displays the entries in the issued certificate list (ICL).

Chapter 16. Using information from SYS1.LOGREC

'SYS1.LOGREC' records unusual runtime events, such as exceptions or unexpected return codes from calls to system services. It records hardware errors, selected software errors, and selected system conditions in the LOGREC data set. You can use the LOGREC data set as a starting point for diagnosing a problem. It supplies symptom data about the failure and shows the order in which errors occurred. After you have collected this information, you should report the problem to the IBM support center.

The following table describes the contents of the LOGREC data for PKI Services:

Table 52. LOGREC data for PKI Services

CSECT	Description	
IKYAPIMS	Issued when an exception occurs during new_cert_post_rtn() processing (creating an ObjectStore entry for purposes of posting an issued certificate to LDAP).	
	Primary Symptom String:	
	Component ID (PIDS):	5752XXPKI
	Load module:	IKYAPI#L
	CSECT:	IKYAPIMS
	Failing routine:	IKYNEWCP
	Error information:	Consists of *either* an abend code and a reason code *or* a facility ID and a message number. **Abend code:** If present, either the character *U* followed by 4 decimal digits or the character *S* followed by 3 hexadecimal digits. **Reason code:** If present, 8 hexadecimal digits. **Facility ID:** If present, 3 characters. **Message number:** If present, 8 hexadecimal digits.
	Secondary Symptom String:	
	NEWID	An 8-digit hexadecimal string that is the ObjectStore entry ID.
	UFN	A character string that is the user-friendly-name.

Table 52. LOGREC data for PKI Services (continued)

CSECT	Description
IKYSCHDR	Issued from the dispatcher() function when an exception is caught while creating and posting a CRL to LDAP.
	Primary Symptom String: **Component ID (PIDS):** 5752XXPKI **Load module:** IKYMISC#L **CSECT:** IKYSCHDR **Failing routine:** IKYDSPER **Error information:** Consists of *either* an abend code and a reason code *or* a facility ID and a message number. **Abend code:** If present, either the character *U* followed by 4 decimal digits or the character *S* followed by 3 hexadecimal digits. **Reason code:** If present, 8 hexadecimal digits. **Facility ID:** If present, 3 characters. **Message number:** If present, 8 hexadecimal digits.
	Secondary Symptom String: **THREAD** The string ″DISPATCHR″.
IKYTIMER	Issued when an exception is caught while processing a timer event in wakeup_rtn().
	Primary Symptom String: **Component ID (PIDS):** 5752XXPKI **Load module:** IKYOSSRV#L **CSECT:** IKYTIMER **Failing routine:** IKYWAKUP **Error Information:** Consists of *either* an abend code and a reason code *or* a facility ID and a message number. **Abend code:** If present, either the character *U* followed by 4 decimal digits or the character *S* followed by 3 hexadecimal digits. **Reason code:** If present, 8 hexadecimal digits. **Facility ID:** If present, 3 characters. **Message number:** If present, 8 hexadecimal digits.
	Secondary Symptom String: **EVENTFUNC** The name of the event routine being processed (postEvt, createEvt, or removeEvt).

Table 52. LOGREC data for PKI Services (continued)

CSECT	Description	
IKYP0N IKYP81 IKYP8A IKYP8B	Issued when an ABEND occurs in the one of the CSECTs running on the Monitor Thread.	
	Primary Symptom String:	
	Component ID (PIDS):	5752XXPKI
	Load module:	IKYPKID#L
	CSECT:	IKYP0N, IKYP81, IKYP8A, or IKYP8B
	Recovery routine:	ESTEXIT
	Error Information:	Consists of an abend code and reason code:
		Abend code: The character *S* followed by 4 hexadecimal digits or the character *U* followed by 4 decimal digits.
		Reason code: 8 hexadecimal digits.
IKYP8B	Issued when an ABEND occurs in the PC routine (or helper routines).	
	Primary Symptom String:	
	Component ID (PIDS):	5752XXPKI
	Load module:	IKYPKID#L
	CSECT:	IKYP8B
	Recovery routine:	ARREXIT
	Error information:	Consists of an abend code and a reason code.
		Abend code: The character *S* followed by 4 hexadecimal digits or the character *U* followed by 4 decimal digits.
		Reason code: 8 hexadecimal digits.

Table 52. LOGREC data for PKI Services (continued)

CSECT	Description
IKYP8A	Issued when an exception is caught in the service thread routine IKYP8A01 or in the services thread request routine IKYP8A02.
	Primary Symptom String: **Component ID (PIDS):** 5752XXPKI **Load module:** IKYPKID#L **CSECT:** IKYP8A **Failing routine:** IKYP8A01 or IKYP8A02 **Error information:** Consists of *either* an abend code and a reason code *or* a facility ID and a message number. **Abend code:** If present, either the character *U* followed by 4 decimal digits or the character *S* followed by 3 hexadecimal digits. **Reason code:** If present, 8 hexadecimal digits. **Facility ID:** If present, 3 characters. **Message number:** If present, 8 hexadecimal digits.
	Secondary Symptom String: **USER** The user ID of the requestor. **FUNC** A function code of 8 hexadecimal digits.

Sample LOGREC data

The following is a sample of a LOGREC data for PKI Services:

```
TYPE:  SYMPTOM RECORD       REPORT:  SOFTWARE EDIT REPORT           DAY YEAR
                                                    REPORT DATE: 221  01
SCP:   VS 2 REL 3                                    ERROR DATE: 221  01
                            MODEL:  9672                            HH MM SS.TH
                            SERIAL: 048288                  TIME: 19:05:16.02
SEARCH ARGUMENT ABSTRACT:
    PIDS/5752XXPKI RIDS/IKYPKID#L RIDS/IKYP8A RIDS/IKYP8A01 AB/S0C4
    FLDS/RSNCODE VALU/H00000000

SYSTEM ENVIRONMENT:
    CPU MODEL:  9672              DATE:  221  01
    CPU SERIAL: 048288            TIME:  19:05:16.02
    SYSTEM:     DCEIMGUI          BCP:   MVS
    RELEASE LEVEL OF SERVICE ROUTINE:      HBB7703
    SYSTEM DATA AT ARCHITECTURE LEVEL:     10
    COMPONENT DATA AT ARCHITECTURE LEVEL:  10
    SYSTEM DATA:  00000000 00000000    |........|
COMPONENT INFORMATION:
    COMPONENT ID:             5752XXPKI
    COMPONENT RELEASE LEVEL:  7706
    SERVICE RELEASE LEVEL:    HKY7706
    DESCRIPTION OF FUNCTION:  PKI SERVICES DAEMON

PRIMARY SYMPTOM STRING:
    PIDS/5752XXPKI RIDS/IKYPKID#L RIDS/IKYP8A RIDS/IKYP8A01 AB/S0C4
    FLDS/RSNCODE VALU/H00000000

    SYMPTOM            SYMPTOM DATA     EXPLANATION
    ---------------    -------------    -----------
    PIDS/5752XXPKI     5752XXPKI        COMPONENT IDENTIFIER
    RIDS/IKYPKID#L     IKYPKID#L        ROUTINE IDENTIFIER
    RIDS/IKYP8A        IKYP8A           ROUTINE IDENTIFIER
    RIDS/IKYP8A01      IKYP8A01         ROUTINE IDENTIFIER
    AB/S0C4            0C4              ABEND CODE - SYSTEM
    FLDS/RSNCODE       RSNCODE          DATA FIELD NAME
    VALU/H00000000     00000000         ERROR RELATED HEXADECIMAL VALUE

SECONDARY SYMPTOM STRING:
    FLDS/USER VALU/CG422253 FLDS/FUNC VALU/H00000000

    SYMPTOM            SYMPTOM DATA     EXPLANATION

    ---------------    -------------    -----------
    FLDS/USER          USER             DATA FIELD NAME
    VALU/CG422253      G422253          ERROR RELATED CHARACTER VALUE
    FLDS/FUNC          FUNC             DATA FIELD NAME
    VALU/H00000000     00000000         ERROR RELATED HEXADECIMAL VALUE

THE SYMPTOM RECORD DOES NOT CONTAIN FREE FORMAT COMPONENT INFORMATION.
HEX DUMP OF RECORD:
  HEADER
    +000    4C831800    00000000    0001221F    19051602    |<C.............|
    +010    FF048288    96720000                            |..BHO...       |
```

Figure 30. Sample LOGREC data (Part 1 of 2)

```
SYMPTOM RECORD
   +000    E2D9F9F6    F7F2F0F4    F8F2F8F8    FFFFCA5B    |SR9672048288...$|
   +010    B64312D1    0360F103    40404040    40404040    |...J.-1.        |
   +020    4040C4C3    C5C9D4C7    E4C9F5F7    F5F2C8C2    |  DCEIMGUI5752HB|
   +030    C2F7F7F0    F3400080    00000000    00000000    |B7703 ..........|
   +040    F1F00030    00640070    005C0138    003101A0    |10.......*......|
   +050 LENGTH(0032) ==> ALL BYTES CONTAIN X'00'.
   +070    E2D9F2F1    F1F0F5F7    F5F2E7E7    D7D2C900    |SR21105752XXPKI.|
   +080    F7F7F0F6    C8D2E8F7    F7F0F640    00000000    |7706HKY7706 ....|
   +090    00000000    00000000    00000000    D7D2C940    |............PKI |
   +0A0    E28599A5    898385A2    40848185    94969540    |SERVICES DAEMON |
   +0B0    40404040    40404040    40404040    00000000    |            ....|
   +0C0    00000000    00000000    00000000    00000000    |................|
   +0D0    00000000    0B41465C    0B414668    0B414699    |.......*.......R|
   +0E0    0B4146A8    0B4146A8    0B4146A8    01000000    |...Y...Y...Y....|
   +0F0    0B4144C8    00000000    00000000    F0F1F2F3    |...H........0123|
   +100    F4F5F6F7    F8F9C1C2    C3C4C5C6    00680040    |456789ABCDEF... |
   +110    0000000F    0B414530    00000000    0B414374    |................|
   +120    00000000    F0F00000    00000008    00000008    |....00..........|
   +130    00000000    40E70030    D7C9C4E2    61F5F7F5    |.... X..PIDS/575|
   +140    F2E7E7D7    D2C940D9    C9C4E261    C9D2E8D7    |2XXPKI RIDS/IKYP|
   +150    D2C9C47B    D340D9C9    C4E261C9    D2E8D7F8    |KID#L RIDS/IKYP8|
   +160    C140D9C9    C4E261C9    D2E8D7F8    C1F0F140    |A RIDS/IKYP8A01 |
   +170    C1C261E2    F0C3F440    C6D3C4E2    61D9E2D5    |AB/S0C4 FLDS/RSN|
   +180    C3D6C4C5    40E5C1D3    E461C8F0    F0F0F0F0    |CODE VALU/H00000|
   +190    F0F0F040    0B414780    00000001    00000000    |000 ............|
   +1A0    C6D3C4E2    61E4E2C5    D940E5C1    D3E461C3    |FLDS/USER VALU/C|
   +1B0    C7F4F2F2    F2F5F340    C6D3C4E2    61C6E4D5    |G422253 FLDS/FUN|
   +1C0    C340E5C1    D3E461C8    F0F0F0F0    F0F0F0F0    |C VALU/H00000000|
   +1D0    40                                              |                |
```

Figure 30. Sample LOGREC data (Part 2 of 2)

Chapter 17. Using information from the PKI Services logs

This chapter explains viewing SYSOUT information. It describes the _PKISERV_MSG_LEVEL environment variable and lists subcomponents and message levels you can select. It explains how to display and change logging options.

Viewing SYSOUT information

To start PKI services, you use the PKISERVD sample proc (see “PKISERVD sample procedure to start PKI Services daemon” on page 292 for a code sample of the JCL). When you start PKI Services, error and informational messages for the PKISERVD job are written to the STDOUT and STDERR file streams. Unless you change the DD statements that specify STDOUT and STDERR in the PKISERVD sample proc, PKI Services writes these messages to SYSOUT.

To view the SYSOUT information of a job, you use the Spool Display Search Facility (SDSF) or a comparable facility. If you are using SDSF, you can use the question mark line command (by entering a question mark in the prefix area in front of the file name) to separate the job files, including STDOUT and STDERR. Figure 31 on page 190 shows this.

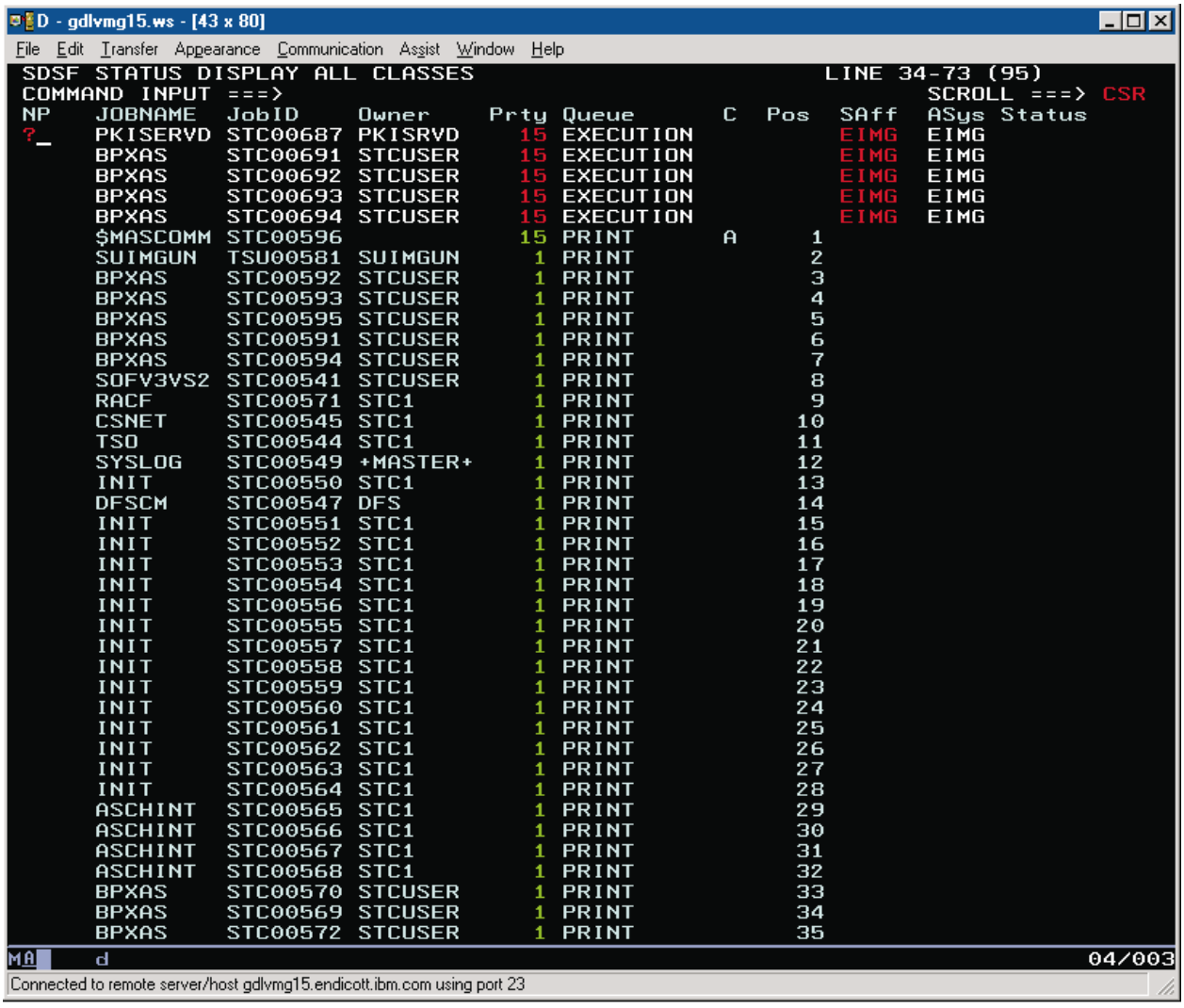


Figure 31. Separating the job files

After using the question mark line command, you can select the file you want to view by entering an *S* before this file name. Figure 32 on page 191 shows this:

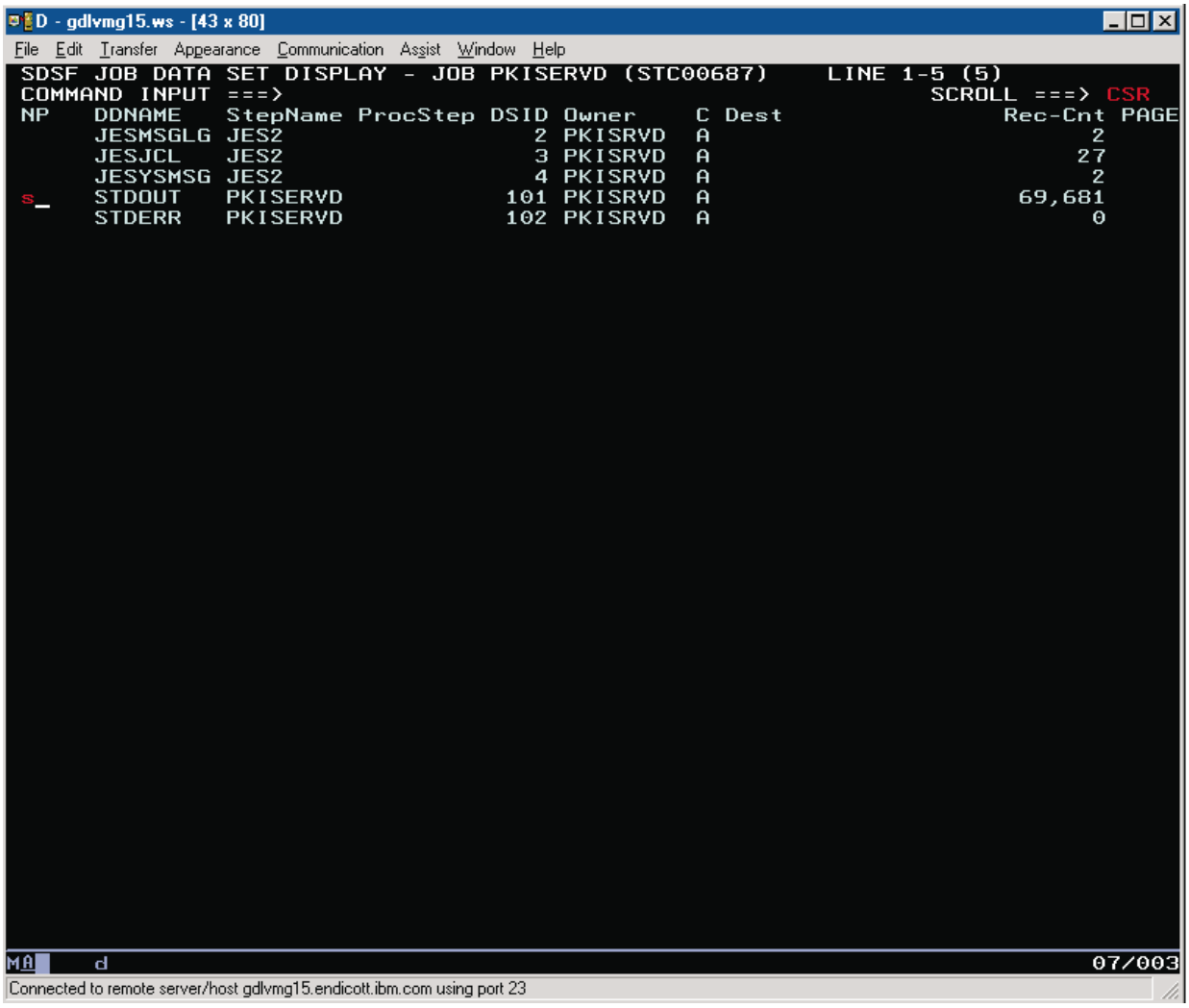


Figure 32. Selecting a file to view

Figure 33 on page 192 shows the messages contained in the file:

Using information from the PKI Services logs

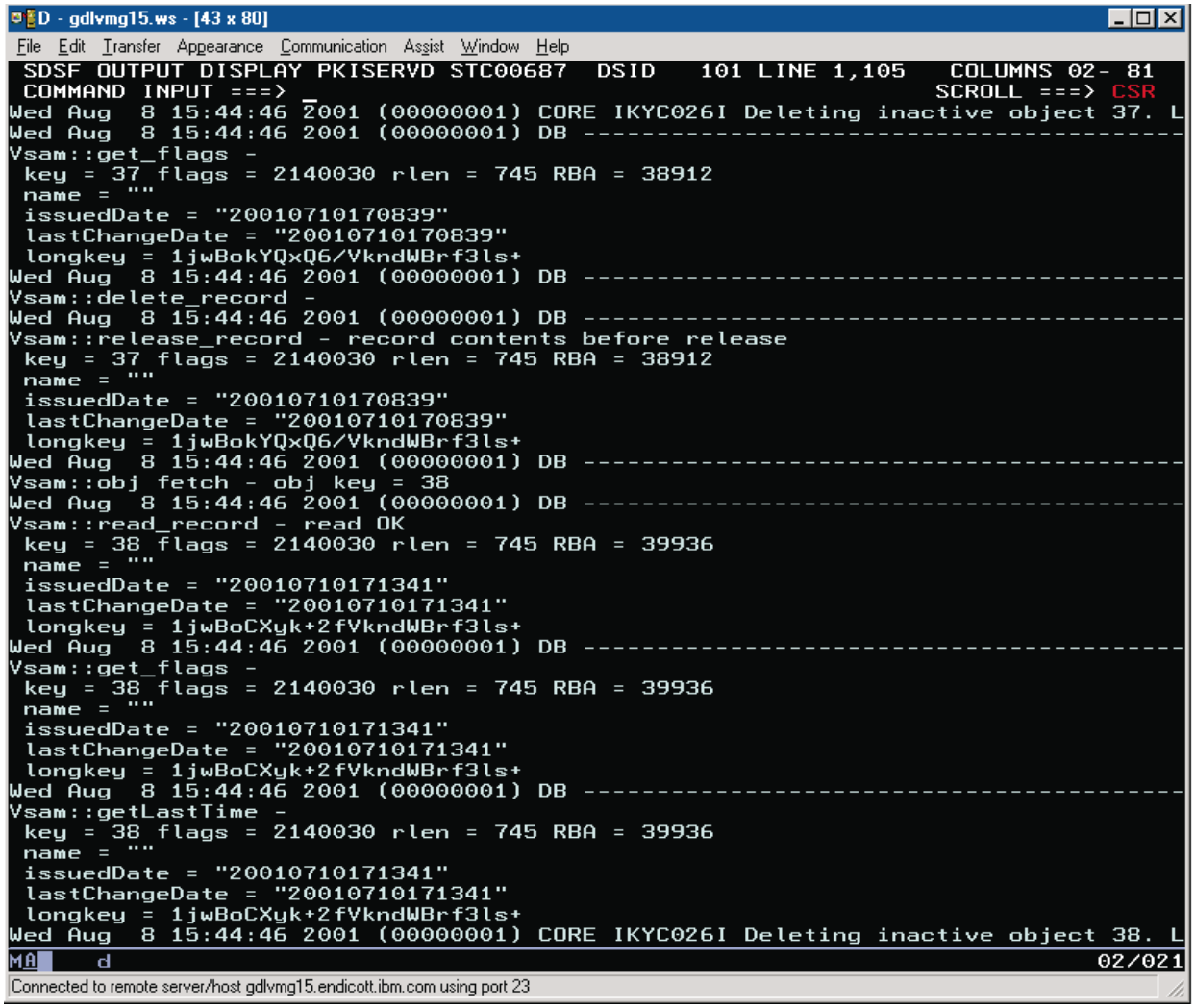


Figure 33. Messages contained in the file

Notes:

1. These messages were produced when Verbose tracing was active.
2. The SYSOUT records have a logical record length of 133, so you may have to scroll to the right to see the entire record.

From left to right, each record contains:

- A time stamp
- The thread identifier, in parenthesis
- The subcomponent name (in the example that follows, this is ″CORE″)
- The message itself, which may span multiple lines.

  Informational, warning, error, and severe level messages begin with a message number. (See Chapter 19, “Messages” on page 203.) Verbose and debug level messages do not have message numbers and are not documented.

The following is an example of an informational message:

```
Wed Aug  8 15:44:46 2001 (00000001) CORE IKYC026I Deleting inactive object 37.
     Last changed at 2001/07/10 17:08:39
```

_PKISERV_MSG_LEVEL subcomponents and message levels

_PKISERV_MSG_LEVEL is an environment variable that specifies the subcomponent and message level for logging messages.

The subcomponents are:

Subcomponent	Meaning
*	This is the wildcard character, which represents all subcomponents.
CORE	The core functions of PKI Services that are not specific to the other subcomponents.
DB	Activity related to the request or issued certificate VSAM data stores
LDAP	LDAP posting operations
PKID	The PKI Services daemon address setup and infrastructure.
POLICY	Certificate creation and revocation policy processing
SAF	SAF key ring, OCEP, and R_datalib calls

The message levels are:

Debug level (hierarchically listed)	Meaning
S	This indicates logging only Severe messages.
E	This indicates logging Severe and Error messages.
W	This indicates logging Severe, Error, and Warning messages. **This is the default message level for all subcomponents if you do not set the environment variable.**
I	This indicates logging Severe, Error, Warning, and Informational messages.
D	This indicates logging Severe, Error, Warning, Informational and Diagnostic.
V	This indicates logging ALL messages, including Verbose Diagnostic messages. This is very verbose. Do not use it unless IBM support personnel instruct you to do so.

(For information about updating the environment variables during configuration, see “Optionally updating PKI Services environment variables” on page 40.)

After PKI Services is up and running, if a problem occurs, the MVS programmer can:

- Change the logging options dynamically — by using the MODIFY (minimum abbreviation F) console command
- Display the current settings — by using another MODIFY console command

Changing logging options

To change logging options dynamically, enter the following MODIFY (or F) console command:

```
F PKISERVD,LOG sub-component.level[,sub-component.level...]
```

subcomponent.level
Sets the message level setting(s) for the subcomponent(s). Use one of the subcomponents and message levels listed previously.

Displaying log options settings

To display the current logging options, enter the following MODIFY (or F) console command:

```
F PKISERVD,DISPLAY
```

Example (of output):

```
12.55.51 IKYP027I PKI SERVICES SETTINGS:
  SUBCOMPONENT              MESSAGE LEVEL
     LDAP                   ERROR MESSAGES AND HIGHER
     SAF                    WARNING MESSAGES AND HIGHER
     DB                     INFORMATIONAL MESSAGES AND HIGHER
     CORE                   WARNING MESSAGES AND HIGHER
     PKID                   VERBOSE DIAGNOSTIC MESSAGES AND HIGHER
     POLICY                 WARNING MESSAGES AND HIGHER
  MESSAGE LOGGING SETTING: STDERR_LOGGING
  CONFIGURATION FILE IN USE:
/etc/pkiserv/pkiserv.conf
```

Chapter 18. Using PKI Services utilities

This chapter describes the following utility programs, which are shipped with PKI Services. These programs are installed in the /bin subdirectory (/usr/lpp/pkiserv/bin).

- vosview — A program to display the entries contained in a VSAM ObjectStore data set (the request database).
- iclview — A program to display the entries contained in a VSAM issued certificate list (ICL) data set.

vosview

Purpose

The vosview program displays the entries contained in a VSAM ObjectStore data set (the request database). Each VSAM request record consists of a fixed header, followed by a variable-length section containing the BER-encoded request. For each entry vosview displays the header information and optionally calls a user-provided program to process the BER-encoded request.

Format

```
vosview vsam-dataset-name [data-decode-command-string]
```

Parameters

vsam-dataset-name
Is an MVS-style data set name (DSN).

Note: Make sure to include the escape character, which is a backslash (\), before the quotation marks enclosing the MVS data set name, for example, `\'pkisrvd.vsam.ost\'`.

data-decode-command-string
Is an optional command to call for decoding the ASN.1 encoded data. The command must be able to read and decode binary (BER) data from STDIN.

Examples

To view the records in VSAM ObjectStore data set 'PKISRVD.VSAM.OST', passing the request data to a utility called `dumpasn1` with the `-o` option, use the following:

```
vosview \'pkisrvd.vsam.ost\' "dumpasn1 -o -"
```

Notes:

1. You can use vosview only against VSAM data sets that PKI Services is not currently using.
2. A dumpasn1 utility is not shipped with PKI Services.

The fixed header data that is displayed for each record would look like the following:

```
Object key = 105
name = "John Q. Public"
longkey = 1F45AEF2D3729FA35156BC47
appldata = "1YBSSL"
comment = ""
data len = 570
flags = 1020111 - Type = Cert State = RA CertReqActive [State Flag]
```

Object key
Is the index into the VSAM data set name.

name
The requestor's name.

longkey
The transaction ID data.

appldata
An 8-character string identifying to the application the short name or nickname of the certificate template. (PKI Services provides eight certificate templates but it is RACF, or an equivalent security product, rather than PKI Services that

handles the SAF templates.) The following table lists the nicknames for the certificate templates. (These are the nicknames that are in the pkiserv.tmpl certificate templates file by default. Your installation may have changed these nicknames or added others during customization. See “TEMPLATE sections” on page 74 for more information.)

Table 53. Nicknames of certificate templates for appldata

Type of certificate	Nickname
One-year PKI SSL browser certificate	1YBSSL
One-year PKI S/MIME browser certificate	1YBSM
Two-year PKI browser certificate for authenticating to z/OS	2YBZOS
Five-year PKI SSL server certificate	5YSSSL
Five-year PKI IPSEC server (firewall) certificate	5YSIPS
Five-year PKI intermediate CA certificate	5YSCA
One-year SAF browser certificate	(No nickname)
One-year SAF server certificate	(No nickname)

comment
A comment the administrator supplied the last time the request was updated.

data len
The length of the variable data portion (that is, the BER-encoded request).

flags
Represent the current state of the request:

Type

Cert Certificate request (new or renewal).

CRL Certificate revocation list (CRL).

Rev Revocation request.

Post Certificate waiting to be posted to LDAP.

State
The prefix (`RA` or `CA`) and one of the following:

CertReqActive Certificate request in some state of being completed.

CertSigned Certificate request where the certificate has been created.

CertReqRejected Certificate request that has been rejected.

RevReqActive Revocation request in some state of being completed.

CRLWaitingForRA CRL to be posted to LDAP.

CertPostPending Certificate to be posted to LDAP.

CaInfoPostPending PKI Services’ CA certificate to be posted to LDAP.

State Flag
Optional. If present, is one of the following:

Complete	Request is complete. For approved requests, the end user has retrieved the certificate.
Error	The certificate could not be posted to LDAP.
NeedsConfirm	Approved or rejected. End user has yet to be notified of the final outcome.

iclview

Purpose

The iclview program displays the entries contained in a VSAM issued certificate list (ICL) data set. Each VSAM ICL record consists of a fixed header followed by a variable-length section containing the BER-encoded certificate. For each entry iclview displays the header information and optionally calls a user-provided program to process the BER-encoded certificate.

Format

```
iclview vsam-dataset-name [data-decode-command-string]
```

Parameters

vsam-dataset-name
Is an MVS-style DSN.

Note: Make sure to include the escape character, which is a backslash (\), before the quotation marks enclosing the MVS data set name, for example, `\'pkisrvd.vsam.icl\'`.

data-decode-command-string
Is an optional command to call for decoding the ASN.1 encoded data. The command must be able to read and decode binary (BER) data from STDIN.

Examples

To view the records in VSAM ICL data set 'PKISRVD.VSAM.ICL', passing the certificate to a utility called `dumpasn1` with the `-o` option, use the following:

```
iclview \'pkisrvd.vsam.icl\' "dumpasn1 -o -"
```

Notes:

1. You can use iclview only against VSAM data sets that PKI Services is not currently using.
2. A dumpasn1 utility is not shipped with PKI Services.

The fixed header data that is displayed for each record would look like the following:

```
Cert 10: John Q. Public
   ISSUED (Issued certificate)
   Issued at 2000-10-25 14:07:05
   Last changed 2000-10-25 14:07:05
   Subject: CN=John Q. Public,OU=Tools Dept,O=IBM,C=US
   Issuer: CN=pkica,OU=zOS Security Server,O=IBM,C=US
   Requestor: John Q. Public
   Appldata: "1YBSSL"
   Serial Number: 0A
```

An explanation of these lines follows:

- The first line specifies certificate's sequential position within the ICL, relative to the other certificates, and requestor's name.
- The second line specifies the certificate state, which of one of the following, and comment (if any):
  - ISSUED
  - REVOKED, not posted
  - REVOKED, awaiting CRL post

  - REVOKED, on posted CRL
- `Issued at` is when the certificate was issued.
- `Last changed` is when the administrator last changed the certificate.
- `Subject:` is the name of the person owning the certificate.
- `Issuer:` is the name of the certificate authority that issued the certificate.
- `Requestor:` is the requestor's name.
- `Appldata` is an 8-character string identifying to the application the short name or nickname of the certificate template. (See Table 53 on page 197 for a list and explanation of nicknames.)
- `Serial Number:` is the serial number of the certificate as a hexadecimal number.

Part 7. Reference information

This part of the book provides reference information, including code samples for important files.

Note: The code samples in this chapter might not be identical to the code shipped with the product. If you want to see the most current code, look in the appropriate source directory.

- Chapter 19, “Messages” on page 203 explains PKI Services messages.
- Chapter 20, “File directory structure” on page 219 describes product and HFS directories for PKI Services and files contained in them.
- Chapter 21, “The pkiserv.conf configuration file” on page 221 provides a code sample of the pkiserv.conf configuration file.
- Chapter 22, “The pkiserv.tmpl certificate templates file” on page 223 provides a code sample of the pkiserv.tmpl file. (For detailed explanations about the contents of this file, see Chapter 10, “Customizing the end-user Web pages” on page 65.)
- Chapter 23, “Environment variables” on page 265 explains the pkiserv.envars environment variables file and provides a code sample.
- Chapter 24, “The IKYSETUP REXX exec” on page 269 explains the contents of the IKYSETUP REXX exec that performs RACF administration and provides a code sample.
- Chapter 25, “Other code samples” on page 287 provides additional code samples. The following table summarizes information about these code samples and those in the preceding chapters, summarizing their use, directory location, and the page where the code sample begins.

Table 54. Summary of information about important files

File	Description	Source location (default)	For code sample...
httpd.conf and httpd2.conf	Contain z/OS HTTP Server directives.	/usr/lpp/pkiserv/samples/	See page 287
IKYCVSAM	Sample IDCAMS JCL to create VSAM data sets.	SYS1.SAMPLIB	See page 289
IKYSETUP	REXX exec to set up RACF profiles.	SYS1.SAMPLIB	See page 269
pkiserv.conf	PKI Services configuration file.	/usr/lpp/pkiserv/samples/ (You copy this file to the runtime directory, /etc/pkiserv.)	See page 221
PKISERVD	Sample proc to start PKI Services daemon.	SYS1.PROCLIB	See page 292
pkiserv.envars	PKI Services environment variables file.	/usr/lpp/pkiserv/samples/ (You might need to copy this file to the runtime directory, /etc/pkiserv.)	See page 267
pkiserv.tmpl	PKI Services certificate template file.	/usr/lpp/pkiserv/samples/ (You copy this file to the runtime directory, /etc/pkiserv.)	See page 223

- Chapter 26, “The certificate validation service” on page 295 describes the certificate validation service. It gives an overview of the OCSF plug-in PKITP,

describes certificate policies and extensions, and explains additional configuration needed for PKITP and using the Trust Policy API, CSSM_TP_PassThrough.

Chapter 19. Messages

PKI Services message numbers begin with the three-character component prefix (IKY), followed by a fourth character that identifies the subcomponent. The following table lists the characters representing various subcomponents and describes where the messages appear.

Table 55. Meaning of fourth character in message number

Character — Meaning	Component producing messages	Where messages appear
C — CORE	Core subcomponent	PKI Services log
D — DB	Database accessing subcomponent	PKI Services log
I — INTERFACE	PKISERV CGIs	In the user's Web browser window
L — LDAP	LDAP bind subcomponent	PKI Services log
O — POLICY	Certificate creation and revocation policy subcomponent	PKI Services log
P — PKID	PKI Services daemon address space controller	• PKI Services log • (For those with destination and routing codes) operators console
S — SAF	SAF interfacing subcomponent	PKI Services log

Characters five through seven are numeric. The eighth character is the message type:

Table 56. Meaning of eighth character in message number

Character — Meaning	Action required
I — Informational (Status message)	No action required
E — Eventual Action	Possible problem that may require eventual action
A — Action Required	Problem that requires immediate attention

For information about setting messages options using environment variables, see page 265.

IKYC001I **Error** *nnnn* **action-being-performed:** *error-code-description*

Explanation: PKI Services is processing a request and has encountered an internal error. The action being performed and the error code encountered are displayed. A description of the error is also displayed, if known.

System Action: The request is not processed.

System Programmer Response: Report the error to the IBM support center.

IKYC002I **Error** *nnnn* **returned from CP_NewCertCreate:** *error-code-description*

Explanation: PKI Services is attempting to create a certificate and has encountered an internal error. The action being performed and the error code encountered are displayed. A description of the error is also displayed, if known.

System Action: The certificate is not created.

System Programmer Response: Report the error to the IBM support center.

IKYC003I **Error** *nnnn* **registering the next CRL cutting job:** *error-code-description*

Explanation: PKI Services has just finished creating the current CRL and is attempting to schedule the next CRL creation thread. An error was encountered. The error code encountered is displayed. A description of the error is also displayed, if known.

System Action: Future CRLs are not created until the problem is corrected and PKI Services is restarted.

System Programmer Response: Look for other error messages that may be issued such as IKYC011I. If no other messages were issued, report the error to the IBM support center.

IKYC004I **Error** *nnnn* **creating and sending CRLs:** *error-code-description*

Explanation: PKI Services is attempting to create the current CRL and has encountered an error. The error code encountered is displayed. A description of the error is also displayed, if known. Note: If the error code is an OCSF return code, no error description is displayed. This would indicate a problem posting the CRL to the LDAP directory.

System Action: If the CRL was created and the post to LDAP was unsuccessful, the post request remains in the PKI Services request database to be reattempted later. If posting continues to be unsuccessful for one week, the information is removed from the request database and deleted. For all other errors, PKI Services tries again to create the CRL during the next CRL interval.

System Programmer Response: If this is a problem with posting to LDAP, you should also see messages IKYC007I or IKYC008I or both. If so, follow the instructions for these messages. Otherwise, report the error to the IBM support center.

IKYC005I **Error** *nnnn* **posting {User | CA} Certificate to LDAP for distinguished-name:** *error-code-description*

Explanation: PKI Services is attempting to post a certificate to the LDAP directory and has encountered an error. The distinguished name for which the post was attempted and the error code encountered are displayed. A description of the error is also displayed, if known. Note: If the error code is an OCSF return code, no error description is displayed.

System Action: The post request remains in the PKI Services request database to be reattempted later. If posting continues to be unsuccessful for one week, the information is removed from the request database and deleted.

System Programmer Response: If no error description is displayed, look up the error code in *z/OS Open Cryptographic Services Facility Application Programming*. If the error is LDAP_NO_SUCH_OBJECT, the LDAP entry could not be created because the required suffix does not exist. Check the message to determine the entry that could not be created. If the entry should be posted to LDAP, you need to define the suffix in the LDAP configuration file (slapd.conf) and recycle the LDAP server. For all other LDAP errors, follow the instructions in *z/OS SecureWay Security Server LDAP Client Programming*. Report non-OCSF errors to the IBM support center. If message IKYC009I is also displayed, report that information as well.

IKYC007I **Error** *nnnn* **posting {CRL | ARL} to LDAP:** *error-code-description*

Explanation: PKI Services is attempting to post a CRL or ARL to the LDAP directory and has encountered an error. The error code encountered is displayed. A description of the error is also displayed, if known. Note: If the error code is an OCSF return code, no error description is displayed.

System Action: The post request remains in the PKI Services request database to be reattempted later. If posting continues to be unsuccessful for one week, the information is removed from the request database and deleted.

System Programmer Response: If no error description is displayed, look up the error code in *z/OS Open Cryptographic Services Facility Application*

Programming. If the error is LDAPDL_NO_SUCH_OBJECT, the LDAP entry to contain the CRL or ARL does not yet exist. This is expected if you are starting PKI Services for the first time. For all other LDAP errors, follow the instructions in *z/OS SecureWay Security Server LDAP Client Programming*. Report non-OCSF errors to the IBM support center. If message IKYC009I is also displayed, report that information as well.

IKYC008I **Error** *nnnn* **creating an {CSSM_DL_DB_PKICA entry for CA Certificate | CSSM_DL_DB_RECORD_CRL entry for {CRL | ARL} | CSSM_DL_DB_PKIUSER entry for User Cert} to LDAP for distinguished-name:** *error-code-description*

Explanation: PKI Services is attempting to post a certificate, CRL or ARL to the LDAP directory and has encountered an error. The distinguished name for which the post was attempted and the error code encountered are displayed. A description of the error is also displayed, if known. Note: If the error code is an OCSF return code, no error description is displayed.

System Action: The post request remains in the PKI Services request database to be reattempted later. If posting continues to be unsuccessful for one week, the information is removed from the request database and deleted.

System Programmer Response: You may also see message IKYC005I or IKYC007I. If so, follow the instructions for the message displayed. Otherwise, if no error description is displayed, look up the error code in *z/OS Open Cryptographic Services Facility Application Programming*. Follow related instructions in *z/OS SecureWay Security Server LDAP Client Programming*. Report non-OCSF errors to the IBM support center. If message IKYC009I is also displayed, report that information as well.

IKYC009I **LDAP post unsuccessful for object id =** *nnnn***, state =** *nnnn***, status =** *nnnn***:** *status-code-description*

Explanation: This message appears as supplemental information for messages IKYC005I, IKYC006I, and IKYC008I.

System Programmer Response: If reporting message IKYC005I, IKYC006I, or IKYC008I to the IBM support center, report this information as well.

IKYC010I **Error** *nnnn* **returned from action-being-performed:** *error-code-description*

Explanation: PKI Services is processing a request and has encountered an error. The action being performed and the error code encountered are displayed. A description of the error is also displayed, if known.

System Action: The request is not processed.

System Programmer Response: In some cases the error code description may be self-explanatory. If not, report the error to the IBM support center.

IKYC011I **Bad TimeBetweenCRLs value in pkiserv.conf file:** *incorrect-value*

Explanation: PKI Services is reading its configuration file to locate the value specified for ″TimeBetweenCRLs″ in the ″CertPolicy″ section. The value specified has an incorrect syntax.

System Action: CRL processing is suspended until the problem is corrected and PKI Services is restarted.

System Programmer Response: Correct the value and restart PKI Services. For more information, see “(Optional) Steps for updating the configuration file” on page 41.

IKYC012I **Bad CRLDuration value in pkiserv.conf file:** *incorrect-value*

Explanation: PKI Services is reading its configuration file to locate the value specified for ″CRLDuration″ in the ″CertPolicy″ section. The value specified has an incorrect syntax.

System Action: CRL processing is suspended until the problem is corrected and PKI Services is restarted.

System Programmer Response: Correct the value and restart PKI Services. For more information, see “(Optional) Steps for updating the configuration file” on page 41.

IKYC013I **Bad CreateInterval value in pkiserv.conf file**

Explanation: PKI Services is reading its configuration file to locate the value specified for ″CreateInterval″ in the ″CertPolicy″ section. The value specified has an incorrect syntax.

System Action: PKI Services uses the default value of 3 minutes.

System Programmer Response: Correct the value and restart PKI Services if desired. For more information, see “(Optional) Steps for updating the configuration file” on page 41.

IKYC014I **Bad RemoveCompletedReqs or RemoveInactiveReqs value in pkiserv.conf file**

Explanation: PKI Services is reading its configuration file to locate the value specified for either

″RemoveCompletedReqs″ or ″RemoveInactiveReqs″ in the ″ObjectStore″ section. The value specified has an incorrect syntax.

System Action: Completed and inactive requests are not removed until the problem is corrected and PKI Services is restarted.

System Programmer Response: Correct the value and restart PKI Services. For more information, see “(Optional) Steps for updating the configuration file” on page 41.

IKYC015I **Bad PostInterval value in pkiserv.conf file**

Explanation: PKI Services is reading its configuration file to locate the value specified for ″PostInterval″ in the ″LDAP″ section. The value specified has an incorrect syntax.

System Action: PKI Services uses the default value of 5 minutes.

System Programmer Response: Correct the value and restart PKI Services if desired. For more information, see “Steps for tailoring the LDAP section of the configuration file” on page 55.

IKYC016I *action-being-performed* **returned** *nnnn* **in sub-function:** *error-code-description*

Explanation: PKI Services is processing a request and has encountered an internal error. The action being performed, the sub-function that returned the error, and the error code encountered are displayed. A description of the error is also displayed, if known.

System Action: The request is not processed.

System Programmer Response: Report the error to the IBM support center.

IKYC017I **JNH_inquire_certreq_startdate (***object-id***) found neither certificate request nor response (***nnnn***):** *error-code-description*

Explanation: PKI Services is processing the start date in a request and has encountered an internal error. The request’s ID and the error code encountered are displayed. A description of the error is also displayed, if known.

System Action: The request is not processed.

System Programmer Response: Report the error to the IBM support center.

IKYC018I **{read | get_value} of certificate-or-CRL-extension-name returned nnnn:** *error-code-description*

Explanation: PKI Services is processing a CRL or certificate extension field and has encountered an internal error. The field name and the error code encountered are displayed. A description of the error is also displayed, if known.

System Action: The CRL or certificate is not processed.

System Programmer Response: Report the error to the IBM support center.

IKYC020I **Retrieving CA value failed** *nnnn***:** *error-code-description*

Explanation: PKI Services is processing a certificate extension field in preparation of posting the certificate to the LDAP directory. The processing has encountered an internal error. The error code encountered is displayed. A description of the error is also displayed, if known.

System Action: The certificate is not posted to the LDAP directory.

System Programmer Response: Report the error to the IBM support center.

IKYC021I **CRL claims to have only User and only CA certs**

Explanation: PKI Services is processing a CRL extension field in preparation of posting the CRL to the LDAP directory. The processing has encountered an internal error. The error code encountered is displayed. A description of the error is also displayed, if known.

System Action: The CRL is not posted to the LDAP directory.

System Programmer Response: Report the error to the IBM support center.

IKYC022I **Invalid type for object** *object-id* **in JNH_set_revreq_invalidityDate:** *error-code-description*

Explanation: PKI Services is processing a revocation request and has encountered an internal error. The revocation request’s ID and the error code encountered are displayed. A description of the error is also displayed, if known.

System Action: The revocation request is not processed.

System Programmer Response: Report the error to the IBM support center.

IKYC023I **Request index (index-number) greater than number of revocations (nnnn) in JNH_set_revreq_invalidityDate**

Explanation: PKI Services is processing a revocation request and has encountered an internal error.

System Action: The revocation request is not processed.

System Programmer Response: Report the error to the IBM support center.

IKYC024I **Failed to schedule event in** *nnnn* **seconds, status =** *nnnn***:** *error-code-description*

Explanation: PKI Services is attempting to schedule a timed event and has encountered an internal error. The error code encountered is displayed. A description of the error is also displayed, if known.

System Action: The event is not scheduled.

System Programmer Response: Report the error to the IBM support center.

IKYC025I **Failed to schedule event status =** *nnnn***:** *error-code-description*

Explanation: PKI Services is attempting to schedule a timed event and has encountered an internal error. The error code encountered is displayed. A description of the error is also displayed, if known.

System Action: The event is not scheduled.

System Programmer Response: Report the error to the IBM support center.

IKYC026I **Deleting {inactive | completed} object** *object-id***. Last changed at** *YYYY*/*MM*/*DD HH*:*MM*:*SS*

Explanation: PKI Services is attempting to purge the request database of inactive and completed requests. A request that has met the criteria for deletion has been found. The request's ID is displayed along with information on when it was last changed. This is an informational message only.

System Action: The request is deleted. PKI Services continues normal processing.

IKYC027I **Removing certificate post request after** *nnnn* **unsuccessful attempts**

Explanation: PKI Services is attempting to purge the request database of unsuccessful LDAP post requests. A request that has met the criteria for deletion has been found. The number of unsuccessful attempts for this request is displayed. This is an informational message only.

System Action: The request is deleted. PKI Services continues normal processing.

IKYC028I **Export for CertId** *certificate-id* **unsuccessful. Request is still pending approval or yet to be issued**

Explanation: A client has requested a certificate and is attempting to retrieve it. The retrieval was unsuccessful because the certificate is not yet available. The request either has yet to be approved by a PKI Services administrator or has been approved, but has not yet been issued by PKI Services. This is an informational message only.

System Action: The state of the request is unchanged. PKI Services continues normal processing.

PKI Services Administrator Response: Use PKI Services administrative functions to query the request to check its state. If the request is still pending approval, determine whether the request should be approved or rejected and take action accordingly. For more information, see "Processing certificate requests" on page 145.

IKYC029I **Error: certificate request type is invalid for certificate creation**

Explanation: PKI Services is processing a certificate request and has encountered an internal error.

System Action: The certificate request is not processed.

System Programmer Response: Report the error to the IBM support center.

IKYC801I *nnnn* **bytes of unconsumed data transferring extensions to certificate template**

Explanation: PKI Services is processing a certificate renewal request and has encountered an internal error. The error code encountered is displayed. A description of the error is also displayed, if known.

System Action: The certificate renewal request is not processed.

System Programmer Response: Report the error to the IBM support center.

IKYC802I **Error** *nnnn* **{ getting certificate-section from old certificate | setting certificate-section in certificate template | removing unnecessary extension from certificate template }:** *error-code-description*

Explanation: PKI Services is processing a certificate renewal request and has encountered an internal error. The error code encountered is displayed. A description

of the error is also displayed, if known.

System Action: The certificate renewal request is not processed.

System Programmer Response: Report the error to the IBM support center.

IKYC901I **Error** *nnnn* **initializing sub-function-name:** *error-code-description*

Explanation: PKI Services is initializing one of its sub-functions and has encountered an error. The sub-function name and error code encountered are displayed. A description of the error is also displayed, if known.

System Action: PKI Services is stopped.

System Programmer Response: This message may accompany a message more specific to the sub-function that failed. Check the log for other error messages issued prior to this one, and diagnose accordingly. Restart PKI Services after making corrections. If you are unable to diagnose the error, report the error to the IBM support center.

IKYC902I **Error initializing the configuration file**

Explanation: PKI Services is reading its configuration file to locate the object identifiers defined in the ″OIDs″ section. Either the section is missing, or a value has an incorrect syntax.

System Action: PKI Services is stopped.

System Programmer Response: The OID values must be defined in dotted-decimal form, for example:

```
sha-1WithRSAEncryption=1.2.840.113549.1.1.5
```

Correct the configuration file, and restart PKI Services. For more information, see “(Optional) Steps for updating the configuration file” on page 41.

IKYC903I **Error** *nnnn* **adding CA certificate to ICL:** *error-code-description*

Explanation: PKI Services is initializing and is attempting to store its own Certificate Authority certificate in the Issued Certificate List (ICL). The attempt was not successful. The error code encountered is displayed. A description of the error is also displayed, if known.

System Action: PKI Services is stopped.

System Programmer Response: This message may accompany a more specific error message. Check the log for other error messages issued prior to this one and diagnose accordingly. Restart PKI Services after making corrections. If you are unable to diagnose the error, report the error to the IBM support center.

IKYD001I **Unable to open VSAM data set** *data-set-name*

Explanation: PKI Services is attempting to open one of the VSAM data sets specified in the ″ObjectStore″ section of the pkiserv.conf file. The open has failed. The data set name is displayed.

System Action: PKI Services is stopped.

System Programmer Response: Locate the failing ″DSN″ value in the pkiserv.conf file. Make sure that the value specifies the correct VSAM data set name and that the data set has been created. If no errors are found, contact your RACF administrator to ensure that the user ID assigned to the PKI Services daemon has permission to open the data set for update. Once corrected, restart PKI Services. For more information, see “(Optional) Steps for updating the configuration file” on page 41 and “Steps for creating the VSAM object store and ICL data sets and indexes” on page 60.

IKYI001I **Request denied by installation exit. RC =** *nn*

Explanation: A user is requesting PKI Services. The PKIServ Web application called an installation-provided exit program. The exit program has determined that the request should be denied. The return code from the exit program is displayed in the message.

System Action: The request in not performed.

User Response: Contact your Web administrator.

Web Administrator Response: Determine why the exit program denied the request and correct the program if necessary.

Destination:

IKYI002I **SAF Service IRRSPX00 Returned SAF RC =** *nn* **RACF RC =** *nn* **RACF RSN =** *nn* **{***diagnostic-information***}**

Explanation: A user is requesting PKI Services. The PKIServ Web application called the IRRSPX00 SAF callable service as requested. The service was unsuccessful. The diagnostic information that follows the message should describe the problem in greater detail:

1 Incorrect field name specified in CertPlist: <*field-name*>.

2 <*field-name*> has an incorrect value.

3 Required field <*field-name*> missing from the request.

4 Request denied, not authorized.

5 Certificate generation provider is not available.

6 Certificate generation provider indicated the following error: <*provider-specific-error-msg*>.

7 Unexpected Error.

System Action: The request in not performed.

User Response: Correct the problem if applicable. If you cannot correct the problem, contact your Web administrator.

Web Administrator Response: Problems 1, 2, and 3 probably indicate an error with the certificate template. Change the certificate template definition in the pkiserv.tmpl file to correct the error.

Problem 4 indicates the user ID assigned to the unit of work calling the IRRSPX00 callable service is not RACF-authorized to perform the request. Determine if the user should have access. If so, use RACF commands to permit the user ID to the required resources.

Problem 5 indicates the PKI Services daemon process has not been started. Start PKI Services; then retry the request.

For problems 6 and 7 or for more information on any of the preceding problems, see earlier chapters in this book and *z/OS Security Server RACF Callable Services*.

IKYI003I **PKI Services CGI error in cgi-program-name:** *diagnostic-error-information*

Explanation: A user is requesting PKI Services. The PKIServ Web application CGI program processing the request detected a problem.The name of the CGI program and additional diagnostic information is displayed in the message.

System Action: The request in not performed.

User Response: Contact your Web administrator.

Web Administrator Response: Locate the CGI program mentioned in the message. (Its default installation location is in a subdirectory under /usr/lpp/pkiserv/PKIServ.) Examine the CGI program's source code to determine the spot where it is failing and why. In most cases, the problem is caused by an error in the PKI Services template file (usually located in /etc/pkiserv/pkiserv.tmpl). Correct the problem and retry the request. For more information, see Chapter 10, "Customizing the end-user Web pages" on page 65 and Chapter 11, "Customizing the administration Web pages" on page 101.

IKYI004I **Installation exit failed. RC =** *nn*

Explanation: A user is requesting PKI Services. The PKIServ Web application called an installation-provided exit program. The exit program either terminated abnormally or returned an unsupported return code value. The return code from the invocation of the exit program is displayed in the message.

System Action: The request in not performed.

User Response: Contact your Web administrator.

Web Administrator Response: Determine why the exit program has failed and correct the program as necessary.

IKYL001I **Error** *nnnn* **{importing | converting} LDAP username** *user's-distinguished-name***:** *error-code-description*

Explanation: PKI Services is reading its configuration file to locate one of the values specified for ″AuthName″ in the ″LDAP″ section. The value specified has a syntax error. The incorrect value is displayed. A description of the error is also displayed, if known.

System Action: PKI Services binds to the LDAP directory anonymously and continues processing. When PKI Services attempts to post certificates and CRLs to this directory, it might fail due to insufficient access. Look for message IKYC007I to determine this is happening. (RC = LDAPDL_INSUFFICIENT_ACCESS)

System Programmer Response: Locate the incorrect ″AuthName″ value in the pkiserv.conf file and correct it. The value must be specified as an LDAP distinguished name, for example, CN=root,O=IBM. Note: The OID qualifiers must be specified in uppercase and there cannot be any spaces surrounding the equal signs or commas separating the attribute value assertions (AVAs). Make corrections as needed, then stop and restart PKI Services. For more information, see "Steps for tailoring the LDAP section of the configuration file" on page 55.

IKYL002I **LDAP bind to** *LDAP-server-domain-name:port* **failed, status = nnnn:** *status-code-description*

Explanation: PKI Services is attempting to bind to one of the LDAP servers specified in the ″LDAP″ section of the pkiserv.conf file. The bind has failed. The failing server name is displayed. A description of the error is also displayed, if known. Note: If the error code is an OCSF return code, no error description will be displayed.

System Action: PKI Services attempts to bind to your other LDAP servers, if any. If PKI Services is unable to bind to any LDAP servers, the LDAP posting of certificates and CRLs is temporarily suspended. PKI Services attempts to bind again during the next posting interval. All post requests will remain in the request database to be attempted later, subject to being deleted after one week of unsuccessful attempts.

System Programmer Response: If no error description is displayed, look up the error code in *z/OS Open Cryptographic Services Facility Application Programming*. Diagnose the problem indicated by the return code. For LDAPDL_SERVER_DOWN, ensure that your LDAP server is running. If so, you may have specified the server name incorrectly in the PKI

Services configuration file. Locate the failing ″Server″ value in the pkiserv.conf file. Correct the value if it does not specify the correct LDAP server domain name and port, then stop and restart PKI Services. For all other LDAP errors, follow the instructions in *z/OS SecureWay Security Server LDAP Client Programming*. Report non-OCSF errors to the IBM support center. If message IKYC009I is also displayed, report that information as well. For more information, see “Steps for tailoring the LDAP section of the configuration file” on page 55.

IKYO001I **Error** *nnnn* **{setting | getting}** *certificate-field* **{in certificate | from template}:** *error-code-description*

Explanation: PKI Services is processing a certificate request field and has encountered an internal error. The field name and the error code encountered are displayed. A description of the error is also displayed, if known.

System Action: The certificate request is not processed.

System Programmer Response: Report the error to the IBM support center.

IKYO002I *nnnn* **bytes of unconsumed data transferring** *certificate-field* **to certificate**

Explanation: PKI Services is processing a certificate request field and has found that the field is larger than it should be. This is an internal error. The field name and the number of extra bytes are displayed.

System Action: The certificate request is not processed.

System Programmer Response: Report the error to the IBM support center.

IKYO003I **The certificate request failed validity checks. Status is** *nnnn***:** *status-code-description*

Explanation: PKI Services is processing a certificate request field and has encountered an internal error. The field name and the status (error) code encountered are displayed. A description of the error is also displayed, if known.

System Action: The certificate request is not processed.

System Programmer Response: Report the error to the IBM support center.

IKYO004I *action-being-performed* **returned nnnn:** *error-code-description*

Explanation: PKI Services is processing a request and has encountered an internal error. The action being performed and the error code encountered are displayed. A description of the error is also displayed, if known.

System Action: The request is not processed.

System Programmer Response: Report the error to the IBM support center.

IKYP001E **ICSF UNAVAILABLE. CERTIFICATE PROCESSING SUSPENDED**

Explanation: PKI Services background certificate processing is attempting to create a digital signature. ICSF manages the private key required for digital signing, and it is not available, either because ICSF is inactive or not configured properly or because the pkiserv daemon user ID does not have authority to use the key

Destination: Descriptor code is 6. Routing code is 2.

System Action: PKI Services background certificate processing is suspended. No certificates or CRLs are issued until the problem is corrected. However, certificate request management functions are still available through the R_PKIServ callable service.

System Programmer Response: Ensure that ICSF is properly configured and is operational. Follow the documentation pertaining to any issued messages having the ″CSF″ prefix. If ICH408I messages are issued for insufficient authority to CSFKEYS or CSFSERV class resources, then the pkiserv daemon user ID does not have authority to use the key. Give the user ID the requires access to the specified resource. For more information, see *z/OS ICSF System Programmer's Guide*, *z/OS ICSF Administrator's Guide*, and “Installing and configuring ICSF (optional)” on page 20.

IKYP002I **PKI SERVICES INITIALIZATION COMPLETE**

Explanation: PKI Services has just been started and has finished initializing.

Destination: Descriptor code is 6. Routing code is 2.

System Action: PKI Services processing continues.

IKYP003I **PKI SERVICES SHUTDOWN REQUESTED**

Explanation: An operator command was issued to stop PKI Services.

Destination: Descriptor code is 5. Routing code is 2.

System Action: PKI Services is stopped.

IKYP004I LOG OPTION PROCESSED: *log-option*

Explanation: A MODIFY operator command was issued to alter the current log setting for PKI Services.

Destination: Descriptor code is 5. Routing code is 2.

System Action: The log setting for PKI Services is changed as requested.

IKYP005I INCORRECT LOG OPTION SPECIFIED

Explanation: A MODIFY operator command was issued to alter the current log setting for PKI Services. The log parameter syntax or value is incorrect.

Destination: Descriptor code is 5. Routing code is 2.

System Action: The MODIFY command is not processed. The log setting for PKI Services is unchanged.

System Programmer Response: Reenter the MODIFY command, specifying a correct log parameter. For more information, see “Changing logging options” on page 193.

IKYP006I UNRECOGNIZED PKI SERVICES COMMAND: SPECIFY LOG, DISPLAY, OR STOP

Explanation: A MODIFY operator command was issued for PKI Services. The command specified is not a supported PKI Services command.

Destination: Descriptor code is 5. Routing code is 2.

System Action: The MODIFY command is not processed. PKI Services continues processing unchanged.

System Programmer Response: Reenter the MODIFY command, specifying a supported PKI Services command. For more information, see “Stopping the PKI Services daemon” on page 62 and “Changing logging options” on page 193.

IKYP007E INSUFFICIENT STORAGE AVAILABLE

Explanation: PKI Services is attempting to allocate storage for processing a MODIFY operator command, but is unsuccessful because of a storage shortage.

Destination: Descriptor code is 5. Routing code is 2.

System Action: The console command is not processed. However, PKI Services might continue processing normally.

Operator Response: Report the problem to your system programmer. After the problem is corrected, you can reenter the command.

System Programmer Response: Increase the region size for the PKI Services started procedure. Stop and restart PKI Services. For more information, see “Steps for starting the PKI Services daemon” on page 60 and “Stopping the PKI Services daemon” on page 62.

IKYP008E DIRECTORY POST UNSUCCESSFUL. LDAP DATA LIBRARY MODULE RC = *nnnn*

Explanation: PKI Services background certificate processing is attempting to post information (certificate, CRL, and so forth) to a directory. The post was unsuccessful. The OCSF Data Library Module (LDAPDL) return code is displayed in the message.

Destination: Descriptor code is 6. Routing code is 2.

System Programmer Response: Determine the cause of the failure from the return code displayed and take appropriate action. These return codes are documented in *z/OS Open Cryptographic Services Facility Application Programming*. If the error is LDAPDL_NO_SUCH_OBJECT, the LDAP entry could not be created because the required suffix does not exist. Check the PKI Services log to determine the entry that could not be created. (Indicated on messages IKYC005I and IKYC008I.) If the entry should be posted to LDAP, you need to define the suffix in the LDAP configuration file (slapd.conf) and recycle the LDAP server. For more information, see “Steps for installing and configuring LDAP” on page 18 and *z/OS Security Server LDAP Server Administration and Use*.

If you want PKI Services to bypass LDAP posting for certificates with missing suffixes, set RetryMissingSuffix=F in the PKI Services pkiserv.conf configuration file. Then, stop and restart the PKI Services daemon. For more information, see “Steps for tailoring the LDAP section of the configuration file” on page 55.

System Action: The information is not posted at this time. The post request remains in the PKI Services request database to be reattempted later. If posting continues to be unsuccessful for one week, the information is removed from the request database.

IKYP009I PKI SERVICES IS STARTING, FMID *product-fmid*

Explanation: The START operator command was issued to start PKI Services. The START command could have been entered directly at the operator’s console or indirectly through a COMMND*xx* parmlib member.

Destination: Descriptor code is 6. Routing code is 2.

System Action: PKI Services initialization proceeds.

IKYP010I THE CONFIGURATION FILE NAME EXCEEDS THE MAXIMUM LENGTH OF *nnnn* **CHARACTERS**

Explanation: The PKI Services daemon process is starting. Initialization processing is reading the

_PKISERV_CONFIG_PATH environment variable. The value specified is too long.

Destination: Descriptor code is 6. Routing code is 2.

System Programmer Response: Determine the location of your PKI Services environment variables file, and correct the value specified for _PKISERV_CONFIG_PATH. Then, restart PKI Services.

System Action: PKI Services is stopped.

IKYP011I PKI SERVICES ADDRESS SPACE COULD NOT BE MADE NON-SWAPPABLE: ERROR *nnnn*

Explanation: The PKI Services daemon process is starting. Initialization processing is attempting to make the PKI Services address space non-swappable. The attempt was unsuccessful. The SYSEVENT TRANSWAP error code is displayed.

Destination: Descriptor code is 6. Routing code is 2.

System Programmer Response: Look up the error code for SYSEVENT TRANSWAP in *z/OS MVS Programming: Authorized Assembler Services Reference SET-WTO* to determine what to do. Then, restart PKI Services.

System Action: PKI Services is stopped.

IKYP012I SYSTEM FUNCTION *function-name* **DETECTED ERROR —** *error-string*

Explanation: PKI Services processing received an error when calling a system service. The service name and error message are displayed.

Destination: Descriptor code is 6. Routing code is 2.

System Programmer Response: See documentation related to the service that failed. Make any necessary corrections. Then, restart PKI Services.

System Action: PKI Services is stopped.

IKYP013I PKI SERVICES DETECTED AN ERROR DURING INITIALIZATION: ERROR *nnnn*, **REASON 0x***nnnn*

Explanation: PKI Services is starting. Initialization processing is attempting to set up the Program Call (PC) interface. The attempt was unsuccessful. The error and reason codes are displayed.

Destination: Descriptor code is 6. Routing code is 2.

System Programmer Response: Determine the failing service by examining the error code. The values are as follows

1 The PKI Services daemon (IKYPKID) is not APF-authorized.

3 Unable to establish recovery. The reason code displayed is the ESTAEX macro return code.

5 Unable to create a PC linkage table index. The reason code displayed is the LXRES macro return code.

6 Unable to create a PC entry table. The reason code displayed is the ETCRE macro return code.

7 Unable to connect the PC entry table to the linkage table. The reason code displayed is the ETCON macro return code.

8, 10, or 11
Unable to create a name token entry. The reason code displayed is the IEANTCR callable service return code.

For error 1, make the IKYPKID load module in SYS1.LINKLIB APF-authorized. For all other error codes, see the documentation associated with the MVS service that failed. Make corrections as necessary. Then, restart PKI Services.

System Action: PKI Services is stopped.

IKYP014I PKI Services detected an error during termination: Error *nnnn*, **Reason** *nnnn*

Explanation: PKI Services is stopping. Termination processing is attempting to free resources allocated. The attempt was unsuccessful. The error and reason codes are displayed.

System Programmer Response: PKI Services should end normally. If so, no action is needed. However, you might want to diagnose the problem. Determine the failing service by examining the error code:

16 Unable to establish recovery. The reason code displayed is the ESTAEX macro return code.

See associated documentation for the MVS service that failed. Make corrections as necessary.

System Action: PKI Services termination processing continues.

IKYP015I A PKI Services program call request failed: Error *nnnn*

Explanation: PKI Services is processing a PC request. The PC request was cancelled before PKI Services completed processing on it. The error code that was posted at the time of the cancel is displayed.

System Programmer Response: If the error code is 8, no action is required. This is an informational message only. For all other error codes, contact your IBM support center.

System Action: PKI Services processing continues.

IKYP016I THE PKI SERVICES RUNTIME ENVIRONMENT COULD NOT BE INITIALIZED

Explanation: The PKI Services daemon process is starting. Initialization processing is trying to initialize the PKI Services runtime environment within the daemon address space. The attempt was unsuccessful.

Destination: Descriptor code is 6. Routing code is 2.

System Programmer Response: Look for other PKI Services log messages related to this error. For more information, see Chapter 17, “Using information from the PKI Services logs” on page 189.

System Action: PKI Services is stopped.

IKYP017I PKI SERVICES IS ALREADY RUNNING

Explanation: An attempt was made to start more than one instance of the PKI Services daemon.

Destination: Descriptor code is 6. Routing code is 2.

System Action: The first instance of PKI Services continues processing. The second instance is stopped.

IKYP018I PKI Services initialization failed because the program is not APF authorized

Explanation: PKI Services is starting. Initialization processing is attempting to initialize the PKI Services runtime environment within the daemon address space. The attempt was unsuccessful because the PKI Services daemon (IKYPKID) is not APF-authorized.

Destination: Descriptor code is 6. Routing code is 2.

System Programmer Response: Make the IKYPKID load module in SYS1.LINKLIB APF-authorized. Then, restart PKI Services.

System Action: PKI Services is stopped.

IKYP019I PKI Services dump created.

Explanation: PKI Services encountered a severe error during processing and has dumped the process (using the CEE3DMP callable service).

Operator Response: Contact your system programmer.

System Programmer Response: Examine the dump to determine the error. Contact your IBM support center if needed. After the error has been corrected, restart PKI Services. For more information, see “Steps for starting the PKI Services daemon” on page 60 and “Stopping the PKI Services daemon” on page 62.

System Action: PKI Services processing ends.

IKYP020I PKI SERVICES RESTART REGISTRATION COMPLETE ON *system-name*

Explanation: PKI Services is starting. Initialization processing has successfully registered PKI Services for automatic restart (ARM).

Destination: Descriptor code is 6. Routing code is 2.

System Action: PKI Services processing continues.

IKYP021I PKI SERVICES RESTARTING ON *system-name*

Explanation: The PKI Services daemon stopped and is being restarted by the Automatic Restart Manager (ARM). The restart was successful.

Destination: Descriptor code is 6. Routing code is 2.

System Action: PKI Services processing continues.

IKYP022I UNABLE TO REGISTER PKI SERVICES FOR RESTART: ERROR *nnnn***, REASON 0x***nnnn*

Explanation: PKI Services is starting. Initialization processing is attempting to register PKI Services for automatic restart (ARM), using the IXCARM macro service. The attempt was unsuccessful. The IXCARM return and reason codes are displayed. Note: The reason code is displayed in hexadecimal.

Destination: Descriptor code is 6. Routing code is 2.

System Programmer Response: Determine and correct the problem with IXCARM as indicated by the error codes displayed. Then, stop and restart PKI Services if automatic restart capability is desired. For more information, see *z/OS MVS Programming: Sysplex Services Reference*.

System Action: PKI Services initialization continues without automatic restart capability.

IKYP023I PKI Services failed to format the display message

Explanation: A MODIFY operator command was issued to display the current settings for PKI Services. Formatting of the display information failed.

System Programmer Response: Report the error to the IBM support center.

System Action: The settings are not displayed. PKI Services processing continues.

IKYP024I PKI SERVICES DUMPING FOR ABEND *abend-code* **RC** *nnnn*

Explanation: PKI Services has incurred an abend. The abend and reason codes are displayed.

Destination: Descriptor code is 6. Routing code is 2.

System Programmer Response: Use IPCS to examine the dump and diagnose the problem. Contact IBM support if necessary. Restart PKI Services after the error has been corrected.

System Action: PKI Services is stopped.

IKYP025I PKI SERVICES SETTINGS:

Explanation: A MODIFY operator command was issued to display the current settings for PKI Services. See Figure 34 on page 217 for the settings that are displayed.

Destination: Descriptor code is 5. Routing code is 2.

System Action: The settings are displayed. The possible subcomponent message levels are:

- SEVERE MESSAGES ONLY
- ERROR MESSAGES AND HIGHER
- WARNING MESSAGES AND HIGHER
- INFORMATIONAL MESSAGES AND HIGHER
- DIAGNOSTIC MESSAGES AND HIGHER
- VERBOSE DIAGNOSTIC MESSAGES AND HIGHER

Operator Response: You can change the subcomponent message levels with the MODIFY operator command if desired. For more information, see “Changing logging options” on page 193.

IKYP026E PKI SERVICES CA CERTIFICATE EXPIRES ON *yyyy/mm/dd*

Explanation: The certificate that contains the PKI Services CA public key expires on the date shown.

Destination: Descriptor code is 6. Routing code is 2.

System Programmer Response: You should renew the certificate before it expires. If your security product is RACF, your certificate is contained in a RACF profile established when you first configured PKI Services. Follow RACF documentation on how to renew a certificate. This is done using either the RACDCERT TSO command or RACF ISPF panels. For more information, see “Renewing your PKI Services certificate authority certificate” on page 173 and *z/OS Security Server RACF Security Administrator’s Guide*.

System Action: If the certificate has not yet expired, processing continues as normal. After the CA certificate expires, certificates issued by PKI Services might be unusable depending on their usage.

IKYP027E ERROR ACCESSING PKI SERVICES CA CERTIFICATE

Explanation: The PKI Services CA certificate is stored in the security product’s database. PKI Services background certificate processing is attempting to access the certificate using the `R_datalib` SAF callable service. The attempt failed. Message IKYS015I should also appear in the PKI Services log.

Destination: Descriptor code is 6. Routing code is 2.

System Programmer Response: You need to determine why the access failed. Look up the `R_datalib` return code displayed on message IKYS015I in *z/OS Security Server RACF Callable Services*. If your security product is RACF, your certificate is contained in a RACF profile established when you first configured PKI Services. That certificate must be connected as the default certificate to the key ring identified by the KeyRing keyword in the PKI Services configuration file. (The default location for this file is /etc/pkiserv/pkiserv.conf.) If you have only renewed your certificate and have not recycled PKI Services, stopping and restarting the PKI Services daemon might solve the problem. If not, use the RACF RACDCERT LIST and LISTRING commands to determine if the correct certificate is connected to the key ring. Also, use the RACF RLIST command to check that the PKI Services daemon user ID has proper authority to access the profile. Make any required changes. Then, stop and restart PKI Services. For more information, see Chapter 15, “RACF administration for PKI Services” on page 167 and *z/OS Security Server RACF Security Administrator’s Guide*.

System Action: PKI Services background certificate processing is suspended. No certificates are issued until the problem is corrected. However, certificate request management functions are still available through the R_PKIServ callable service.

IKYP028E PKI SERVICES DISTINGUISHED NAME OR KEY CHANGE ERROR

Explanation: PKI Services is starting. Initialization processing has retrieved the PKI Services signing certificate from the key ring assigned to PKI Services. The certificate is incompatible with certificate processing that has previously transpired. The subject’s distinguished name or the public key or both differ from the original values provided when you first configured PKI Services. The original values cannot be changed without reconfiguring PKI Services.

Destination: Descriptor code is 6. Routing code is 2.

System Programmer Response: Determine if PKI Services is processing the correct certificate. If your security product is RACF, your certificate is contained in a RACF profile established when you first configured PKI Services. That certificate must be connected as the default certificate to the key ring identified by the KeyRing keyword in the PKI Services configuration file. (The default location for this file is /etc/pkiserv/pkiserv.conf.) Use the RACF RACDCERT LIST and LISTRING commands to determine if the correct certificate is connected to the key ring. Make any required changes. Then, restart PKI Services. For more information, see Chapter 15, “RACF administration

for PKI Services" on page 167 and *z/OS Security Server RACF Security Administrator's Guide*.

System Action: PKI Services is stopped.

IKYS001I Error *nnnn* **{attaching | detaching}** *OCSF-service-provider-description*

Explanation: PKI Services is attaching or detaching an OCSF or OCEP service provider module. The attach or detach failed. The service provider in error and the error code encountered are displayed.

System Action: PKI Services is stopped.

System Programmer Response: Look up the error code in either *z/OS Open Cryptographic Services Facility Application Programming* or *z/OS SecureWay Security Server Open Cryptographic Enhanced Plug-ins Application Programming*. Diagnose the problem indicated by the return code. Restart PKI Services after corrections are made.

IKYS002I Error nnnn in *OCSF-API-name*

Explanation: PKI Services is calling an OCSF or OCEP API. The invocation has failed. The API name and error code encountered are displayed.

System Action: If the error occurs during PKI Services initialization, PKI Services is stopped. Otherwise, PKI Services continues processing. However, needed cryptographic services may not be available.

System Programmer Response: If you are using ICSF for your CA's private key operations and the failing service is either CSP_CreateSignatureContext or CSSM_SignData, check that ICSF functioning and configured properly for PKA operations. For this problem you will also see console message IKYP001E. Follow the instructions for message IKYP001E. For all other errors look up the error code in either *z/OS Open Cryptographic Services Facility Application Programming* or *z/OS SecureWay Security Server Open Cryptographic Enhanced Plug-ins Application Programming*. Diagnose the problem indicated by the return code. Restart PKI Services after corrections are made, if needed.

IKYS003I Error *nnnn* **in getting {subject name | public key} from certificate:** *error-code-description*

Explanation: PKI Services is retrieving its CA certificate from the SAF key ring. An error occurred while PKI Services was extracting the subject name or public key from the certificate. The error code encountered is displayed. A description of the error is also displayed, if known. This may indicate a problem with the certificate stored in the SAF key ring or it may be an internal error.

System Action: PKI Services is stopped.

System Programmer Response: Ensure that the certificate stored in the SAF key ring is correct. If no problems are found, report the error to the IBM support center. For more information, see Chapter 15, "RACF administration for PKI Services" on page 167 and *z/OS Security Server RACF Security Administrator's Guide*.

IKYS004I Error *nnnn* **in opening key ring** *key-ring-name*

Explanation: PKI Services is initializing and is calling OCSF to open the SAF key ring containing the CA certificate. The open failed. The key ring name and OCSF or OCEP error code encountered is displayed.

System Action: PKI Services is stopped.

System Programmer Response: Look up the error code in either *z/OS Open Cryptographic Services Facility Application Programming* or *z/OS SecureWay Security Server Open Cryptographic Enhanced Plug-ins Application Programming*. Diagnose the problem indicated by the return code. Restart PKI Services once corrections are made.

IKYS005I Error *nnnn* **in closing key ring**

Explanation: PKI Services is terminating and is invoking OCSF to close the SAF key ring containing the CA certificate. The close failed. The OCSF or OCEP error code encountered is displayed.

System Action: PKI Services continues termination.

System Programmer Response: Look up the error code in either *z/OS Open Cryptographic Services Facility Application Programming* or *z/OS SecureWay Security Server Open Cryptographic Enhanced Plug-ins Application Programming*. Diagnose the problem indicated by the return code. Make corrections as indicated. Restart PKI Services if desired.

IKYS006I Cannot delete the signing context

Explanation: PKI Services is attempting to sign a certificate or CRL and is invoking the OCSF API CSSM_DeleteContext. The invocation failed.

System Action: The certificate or CRL is not created.

System Programmer Response: Report the error to the IBM support center.

IKYS007I No KeyRing value specified under SAF section in pkiserv.conf file

Explanation: PKI Services is reading its configuration file to locate the value specified for ″KeyRing″ in the ″SAF″ section. The value is missing or has an incorrect syntax.

System Action: PKI Services is stopped.

System Programmer Response: Correct the value

and restart PKI Services if desired. For more information, see “(Optional) Steps for updating the configuration file” on page 41 .

IKYS008I Signing key is from unknown crypto service provider

Explanation: PKI Services is retrieving its private key from the SAF key ring. The private key type is not known to PKI Services. This may indicate a problem with the certificate and private key stored in the SAF key ring or it may be an internal error.

System Action: PKI Services is stopped.

System Programmer Response: Ensure that the certificate and private key stored in the SAF key ring are correct. If no problems are found, report the error to the IBM support center. For more information, see Chapter 15, “RACF administration for PKI Services” on page 167 and *z/OS Security Server RACF Security Administrator's Guide*.

IKYS009I Profile for key ring *key-ring-name* not found

Explanation: PKI Services is reading its configuration file to locate the value specified for ″KeyRing″ in the ″SAF″ section. The key ring specified is incorrect. No such key ring exists.

System Action: PKI Services is stopped.

System Programmer Response: Correct the value and restart PKI Services if desired. For more information, see “(Optional) Steps for updating the configuration file” on page 41.

IKYS010I Profile for key ring or default certificate or private key not found

Explanation: PKI Services is attempting to retrieve data from the SAF key ring specified by the ″KeyRing″ value in the ″SAF″ section of the pkiserv.conf file. The key ring specified does not appear to be set up properly. Possible problems are:

- Key ring is empty.
- CA certificate in the key ring not connected as PERSONAL DEFAULT.
- CA certificate in key ring does not have a private key.
- User ID assigned to the PKI Services daemon does not have permission to read the key ring or private key.

System Action: PKI Services is stopped.

System Programmer Response: Ensure that the SAF key ring and the certificate stored in it are correct. For more information, see Chapter 4, “Running IKYSETUP to perform RACF administration” on page 23 and *z/OS Security Server RACF Security Administrator's Guide*.

IKYS011I Error *error-description* in pthread_rwlock_rdlock/wrlock

Explanation: PKI Services is retrieving its CA certificate from the SAF key ring. An internal error occurred while PKI Services was calling the pthread_rwlock_rdlock or pthread_rwlock_wrlock UNIX function. A description of the error is displayed.

System Action: PKI Services is stopped.

System Programmer Response: Report the error to the IBM support center.

IKYS012I Error *error-description* in pthread_rwlock_unlock

Explanation: PKI Services is retrieving its CA certificate from the SAF key ring. An internal error occurred while PKI Services was invoking the pthread_rwlock_unlock UNIX function. A description of the error is displayed.

System Action: PKI Services is stopped.

System Programmer Response: Report the error to the IBM support center.

IKYS013I Cannot find the private key associated with the default certificate

Explanation: PKI Services is attempting to retrieve data from the SAF key ring specified by the ″KeyRing″ value in the ″SAF″ section of the pkiserv.conf file. The key ring specified does not appear to be set up properly. Possible problems are:

- Key ring is empty.
- CA certificate in the key ring not connected as PERSONAL DEFAULT.
- CA certificate in key ring does not have a private key.
- User ID assigned to the PKI Services daemon does not have permission to read the key ring or private key.

System Action: PKI Services is stopped.

System Programmer Response: Ensure that the SAF key ring and the certificate stored in it are correct. For more information, see Chapter 4, “Running IKYSETUP to perform RACF administration” on page 23 and *z/OS Security Server RACF Security Administrator's Guide*.

IKYS014I Cannot find the default certificate with private key associated in key ring

Explanation: PKI Services is attempting to retrieve data from the SAF key ring specified by the ″KeyRing″ value in the ″SAF″ section of the pkiserv.conf file. The key ring specified does not appear to be set up properly. Possible problems are:

- Key ring is empty.

- CA certificate in the key ring not connected as PERSONAL DEFAULT.
- CA certificate in key ring does not have a private key.
- User ID assigned to the PKI Services daemon does not have permission to read the key ring or private key.

System Action: PKI Services is stopped.

System Programmer Response: Ensure that the SAF key ring and the certificate stored in it are correct. For more information, see “Locating your PKI Services certificate and key ring” on page 170 and *z/OS Security Server RACF Security Administrator's Guide*.

IKYS015I **RACF callable service, R_datalib, with function code nnnn returns with SAF return code=*nnnn*, RACF return code=*nnnn*, RACF reason code=*nnnn***

Explanation: PKI Services is attempting to retrieve data from the SAF key ring specified by the ″KeyRing″ value in the ″SAF″ section of the pkiserv.conf file. The key ring specified does not appear to be set up properly. Possible problems are:

- Key ring is empty.
- CA certificate in the key ring not connected as PERSONAL DEFAULT.
- CA certificate in key ring does not have a private key.
- User ID assigned to the PKI Services daemon does not have permission to read the key ring or private key.

System Action: PKI Services is stopped.

System Programmer Response: Look up the return and reason code displayed in *z/OS Security Server RACF Callable Services*. Make corrections as needed. Ensure that the SAF key ring and the certificate stored in it are correct. For more information, see Chapter 15, “RACF administration for PKI Services” on page 167 and *z/OS Security Server RACF Security Administrator's Guide*.

```
IKYP025I PKI SERVICES SETTINGS:
    SUBCOMPONENT                 MESSAGE LEVEL
       LDAP                      {current-message-level-for-subcomponent}
       SAF                       {current-message-level-for-subcomponent}
       DB                        {current-message-level-for-subcomponent}
       CORE                      {current-message-level-for-subcomponent}
       PKID                      {current-message-level-for-subcomponent}
       POLICY                    {current-message-level-for-subcomponent}
    MESSAGE LOGGING SETTING:   {STDERR_LOGGING | STDOUT_LOGGING}
    CONFIGURATION FILE IN USE:
  {full-UNIX-pathname-of-configuration-file-being-used}
```

Figure 34. Settings that IKYP025I displays

Chapter 20. File directory structure

This chapter discusses the location of files in:

- z/OS product libraries
- HFS directory /usr/lpp/pkiserv/ and its subdirectories.

Product libraries

SMP/E installs PKI Services into the following product libraries:

- SAMPLIB/ASAMPLIB
  - IKYCVSAM
  - IKYSETUP
  - IKYISMKD
  - IKYMKDIR
  - IKYALLOC
  - IKYDDDEF
- PROCLIB/APROCLIB
  - IKYSPROC with alias PKISERVD
- LINKLIB/ALINKLIB
  - IKYPKID - The PKI Services daemon
  - IKYPRTM - The Resource Termination Manager for the daemon

HFS directory and subdirectories

Additionally, unless you change the default, SMP/E installs PKI Services into the HFS directory /usr/lpp/pkiserv. The following table describes the directory structure and contents:

Table 57. Files contained in subdirectories

Subdirectory	Contains File
bin	Utilities executables: • iclview — Utility for viewing issued certificate list (certificate database). (For more information, see Chapter 18, “Using PKI Services utilities” on page 195.) • pkitp_install — Program to register the PKI Services Trust Policy plug-in with OCSF. (For more information, see “Configuring and getting started with PKITP” on page 299.) • pkitp_ivp — Program to verify that the PKI Services Trust Policy plug-in installed successfully. (For more information, see “Configuring and getting started with PKITP” on page 299.) • vosview — Utility for viewing VSAM object store (request database). (For more information, see Chapter 18, “Using PKI Services utilities” on page 195.)
include	C header files: • pkitp.h — C language header file for writing application programs that use the PKI Trust Policy Plug-in. (For more information, see “Files for PKITP” on page 298.)

Table 57. Files contained in subdirectories (continued)

lib	Loadable files: • pkitp.so — OCSF Trust Policy plug-in for PKI Services. (For more information, see “Files for PKITP” on page 298.) • *.dll — Dynamic Link Libraries (DLLs) that the PKI Services daemon uses. • nls/msg/En_US.IBM-1047/*.cat - The PKI Services message catalogs. (These message catalogs are also symbolically linked in the /usr/lpp/pkiserv/lib/nls/msg/C directory as well as the /usr/lib/nls/msg/En_US.IBM-1047 and /usr/lib/nls/msg/C directories.)
PKIServ	CGIs that make up the PKIServ Web application. (For information about CGIs, see “Relationship between CGIs and the pkiserv.tmpl file” on page 88 and Table 35 on page 101.) PKIServ contains the following subdirectories: • public-cgi — Public (non-SSL) directory • ssi-cgi-bin — SSL-protected – auth — SSL with user ID and password protection. Work runs under client’s ID. – surrogateauth — SSL with user ID and password protection. Work runs under surrogate ID (PKISERV). • clientauth-cgi-bin — SSL with client certificate protection. Work runs under surrogate ID (PKISERV). – auth — SSL with client certificate protection. Work runs under administrator’s ID.
samples	Various sample files, including: • httpd.conf — Contains z/OS HTTP Server directives. (For a code sample, see “z/OS HTTP Server configuration directives” on page 287.) • httpd2.conf — Contains z/OS HTTP Server directives for the second webserver. (For a code sample, see “z/OS HTTP Server configuration directives” on page 287.) • httpd.envvars — A sample of the environment variables needed for PKI Services that you should ″integrate″ into your existing z/OS HTTP Server environment variables file (httpd.envvars). (For a code sample, see “The pkiserv.envars environment variables file” on page 267.) • Makefile.pkiexit — The makefile for the PKI Services exit. (For more information, see “Steps for updating the exit code sample” on page 110.) • Makefile.pkitpsamp — The makefile for pkitpsamp.c, which is a sample application to call the PKI Trust Policy plug-in. (For more information, see “Files for PKITP” on page 298.) • pkiexit.c — The sample PKI Services exit, which PKI Services provides. (For more information, see “Steps for updating the exit code sample” on page 110.) • pkiserv.envars — The PKI Services environment variables file. (For more information, see “Optionally updating PKI Services environment variables” on page 40 and “The pkiserv.envars environment variables file” on page 267.) • pkiserv.tmpl — The PKI Services certificate templates file. (For more information, see Chapter 10, “Customizing the end-user Web pages” on page 65. For a code sample, see Chapter 22, “The pkiserv.tmpl certificate templates file” on page 223.) • pkiserv.conf — The PKI Services configuration file. (For more information, see “(Optional) Steps for updating the configuration file” on page 41 and Chapter 21, “The pkiserv.conf configuration file” on page 221.) • pkitpsamp.c — Sample application to call the PKI Trust Policy plug-in. (For more information, see “Files for PKITP” on page 298 and “Providing the certificate validation service” on page 305.)

Chapter 21. The pkiserv.conf configuration file

This chapter includes a code sample of the pkiserv.conf configuration file.

The pkiserv.conf file is the configuration file for the PKI Services daemon. By default, you can find this file in the /usr/lpp/pkiserv/samples/ directory. For more information about the sections of the pkiserv.conf configuration file and the parameters, see “(Optional) Steps for updating the configuration file” on page 41 and Table 19 on page 43.

The example that follows might not be identical to the code shipped with the product. If you want to see the exact code, look at the pkiserv.conf file in the source directory /usr/lpp/pkiserv/samples/.

```
# Licensed Materials - Property of IBM
# 5694-A01
# (C) Copyright IBM Corp. 2001
# Status = HKY7706

[OIDs]
C=2.5.4.6
O=2.5.4.10
OU=2.5.4.11
CN=2.5.4.3
L=2.5.4.7
ST=2.5.4.8
TITLE=2.5.4.12
POSTALCODE=2.5.4.17
STREET=2.5.4.9
MAIL=0.9.2342.19200300.100.1.3
sha-1WithRSAEncryption=1.2.840.113549.1.1.5
id-dsa-with-sha1=1.2.840.10040.4.3
MyPolicy=1.2.3.4

[ObjectStore]
Name=pkica
Path=/var/pkiserv/
ObjectDSN='pkisrvd.vsam.ost'
ObjectTidDSN='pkisrvd.vsam.ost.path'
ICLDSN='pkisrvd.vsam.icl'
RemoveCompletedReqs=1w
RemoveInactiveReqs=4w

[CertPolicy]
SigAlg1=sha-1WithRSAEncryption
CreateInterval=3m
TimeBetweenCRLs=1d
CRLDuration=2d
PolicyRequired=F
PolicyCritical=F
PolicyName1=MyPolicy
Policy1Org=MyOrganization
Policy1Notice1=3
Policy1Notice2=17
UserNoticeText1=This is some very lawyerly statement for...
CPS1=http://www.mycompany.com/cps.html

[General]
PreferredCryptoProvider=dda0c1e0-7b73-11d0-8e0c-0004ac602b18
InitialThreadCount=10

[SAF]
KeyRing=PKISRVD/CAring
```

The pkiserv.conf configuration file

```
[LDAP]
NumServers=1
PostInterval=5m
Server1=myldapserver.mycompany.com:389
AuthName1=CN=root
AuthPwd1=root
CreateOUValue= Created by PKI Services
RetryMissingSuffix=T
```

Chapter 22. The pkiserv.tmpl certificate templates file

This chapter includes a code sample of the pkiserv.tmpl certificate templates file. (For a description of the main sections and subsections of pkiserv.tmpl, see “Contents of the pkiserv.tmpl certificates templates file” on page 65.) The example that follows might not be identical to the code shipped with the product. To view the most current code, see the pkiserv.tmpl certificate template file in the source directory /usr/lpp/pkiserv/samples/.

```
#
# ====================================================================
#
# COMPONENT_NAME: pkiserv.tmpl
#
# Licensed Materials - Property of IBM
# 5694-A01
# (C) Copyright IBM Corp. 2001
# Status = HKY7706
#
# ====================================================================
#
# Configuration file for interfacing with R_PKIServ. This file may be
# customized as required by the installation. Any line with an '#' in
# column 1 is considered a comment.
#
# Structure:
#
#   The file contains a mixture of true HTML and HTML like tags. The
#   main tags divide the file into sections, APPLICATION, TEMPLATE,
#   and INSERT, where APPLICATION and TEMPLATE may contain various
#   subsections, named fields, and substitution variables as explained
#   below.
#
#   <APPLICATION NAME=appl-name> ... </APPLICATION>
#
#     This section identifies the applications that will make use of
#     PKI Services for Z/OS. The product ships with one application
#     defined, "PKISERV". This section may contain the following subsections:
#
#     <CONTENT> ... </CONTENT>
#
#        This subsection contains the HTML to be presented to the end
#        user requesting and retrieving certificates
#
#        The subsection should contain one or more named fields
#        identifying certificate templates to be used for requesting
#        or managing certificates through this application. (See below
#        for a description of named fields.) These template names
#        should match the HTML selection value associated with them.
#
#     <RECONTENT> ... </RECONTENT>
#
#        This subsection contains the HTML which will display the
#        certificate details so that the end user may confirm that
#        that is the certificate to be renewed or revoked. This will
#        make use of a new substitution variable,[printablecert],
#        which contains the data extracted from the ICL entry.
#
#     <RESUCCESSCONTENT> ... </RESUCCESSCONTENT>
#
#        This subsection contains the HTML to be presented to the end
#        user when the certificate revoke request
#        was submitted successfully.
#        Any named fields in this subsection are interpreted as
#        content inserts defined by INSERT sections. For PKISERV, the
```

```
#        INSERT sections are included as part of the HTML presented
#        to the end user.
#
#     <REFAILURECONTENT> ... </REFAILURECONTENT>
#
#        This subsection contains the HTML to be presented to the end
#        user when the certificate renew/revoke request submit failed.
#        Any named fields in this subsection are interpreted as
#        content inserts defined by INSERT sections. For PKISERV, the
#        INSERT sections are included as part of the HTML presented
#        to the end user.
#
#     <ADMINHEADER> ... </ADMINHEADER>
#
#     This subsection contains the general installation specific HTML
#     content for the header of all admin pages.
#
#     <ADMINFOOTER> ... </ADMINFOOTER>
#
#     This subsection contains the general installation specific HTML
#     content for the footer of all admin pages.
#
#
#   <TEMPLATE NAME=tmpl-name> ... </TEMPLATE>
#   <TEMPLATE NAME=tmpl-name alias>
#   <NICKNAME=nick-name>
#
#     This section defines the certificate templates referenced in the
#     APPLICATION sections. You may refer to a single template by
#     more than one name using alias. Also since the template name
#     needs to be recalled in order to renew a certificate, it will
#     need to be stored with the certificate. The nick name of the
#     template will serve this purpose.
#
#     Applicable subsections are:
#
#     <CONTENT> ... </CONTENT>
#
#        This subsection contains the HTML to be presented to the end
#        user requesting certificates of this type. Any named fields
#        in this subsection are interpreted as certificate field names
#        defined by INSERT sections. (See below for a description of
#        named fields.) For PKISERV, the INSERT sections
#        are included as part of the HTML presented to the end user.
#        (i.e., the end user provides values for these fields.)
#        Named fields in this subsection are considered optional if
#        the named field contains more that one word within the %%
#        delimiters, e.g., %%AltName (Optional)%%. The user need not
#        supply a value for AltName
#
#     <APPL> ... </APPL>
#
#        This subsection identifies certificate fields that the
#        application itself should provide values for. This subsection
#        should contain named fields only, one per line.  Currently,
#        the only supported named field allowed in this section is
#        "UserId"
#
#     <CONSTANT> ... </CONSTANT>
#
#        This subsection identifies certificate fields that have a
#        constant (hardcoded) value for everyone. This subsection
#        should contain named fields only, one per line. The syntax
#        for specifying the values is %%field-name=field-value%%,
#        e.g., %%KeyUsage=handshake%%
#
#     <SUCCESSCONTENT> ... </SUCCESSCONTENT>
```

```
#
#        This subsection contains the HTML to be presented to the end
#        user when the certificate request was submitted successfully.
#        Any named fields in this subsection are interpreted as
#        content inserts defined by INSERT sections. For PKISERV, the
#        INSERT sections are included as part of the HTML presented
#        to the end user.
#
#     <FAILURECONTENT> ... </FAILURECONTENT>
#
#        This subsection contains the HTML to be presented to the end
#        user when the certificate request submit failed.
#        Any named fields in this subsection are interpreted as
#        content inserts defined by INSERT sections. For PKISERV, the
#        INSERT sections are included as part of the HTML presented
#        to the end user.
#
#     <RETRIEVECONTENT> ... </RETRIEVECONTENT>
#
#        This subsection contains the HTML to be presented to the end
#        user to enable certificate retrieval.
#        Any named fields in this subsection are interpreted as
#        content inserts defined by INSERT sections. For PKISERV, the
#        INSERT sections are included as part of the HTML presented
#        to the end user.
#
#     <RETURNCERT> ... </RETURNCERT>
#
#     This subsection contains the HTML to be presented to the
#     end user upon successful certificate retrieval. For PKISERV, if
#     the certificate being retrieved is a browser certificate,then
#     this section must contain a  single line containing a browser
#     qualified INSERT name, e.g.,  %%returnbrowsercert
#     [browsertype]%%.  Additionally, INSERTs for Netscape
#     (returnbrowsercertNS) and Internet Explorer (returnbrowsercertIE)
#     containing browser specific HTML for returning certificates must
#     be defined elsewhere in the configuration file.  If the
#     certificate being retrieved is a server certificate, this section
#     should contain the HTML necessary to present the certificate
#     to the user as text
#
#   <INSERT NAME=insert-name> ... </INSERT>
#     This section contains HTML that either describes a certificate
#     field or defines other common HTML that may be referenced in
#     the TEMPLATE sections.  INSERTs are referenced elsewhere by
#     using a named field of the form %%insert-name%%
#
#   Named Fields - Delineated with %%, e.g., %%Label%%. Their meaning
#   is specific to the section they are contained in.  Named fields
#   are case sensitive. Named fields are also using to reference common
#   includable HTML. Note, PKISERV treats named fields that begin with
#   a dash as just includable code. Any special meaning a named field
#   may have, given the section its contained in, is ignored if it
#   begins with a dash. For example, if %%-pagefooter%% was specified
#   in a TEMPLATE CONTENT section, -pagefooter would not be considered
#   a certificate field name. However, the INSERT with the name
#   -pagefooter would be included in the HTML page presented to the
#   end user.
#
#   Substitution Variables - Delineated with square brackets, e.g.,
#   [base64cert]. They represent variables that get replaced with
#   an actual value at run time. Substitution variables are case
#   sensitive. The valid substitution variables are:
#
#     transactionid - Unique value returned from a certificate request.
#
#     tmplname - Certificate template name. Primed from the HTML tag
```

The pkiserv.tmpl certificate templates file

```
#     <SELECT NAME="Template"> in the <APPLICATION NAME=PKISERV>
#     section.  This is selected by the end user on the first web page.
#
#     iecert - The requested certificate in a form the Microsoft
#              Internet Explorer accepts.
#
#     base64cert - The requested base64 encoded certificate.
#
#     browsertype - Special substitution variable to be used to qualify
#     named field only. Its use enables the different browsers,
#     Netscape and Internet Explorer, to perform browser specific
#     operations, i.e., Netscape uses a KEYGEN HTML tag to generate a
#     public/private key pair while Internet Explorer uses ACTIVEX
#     controls. For example, if %%PublicKey[browsertype]%% was
#     specified in a TEMPLATE CONTENT section referenced by a user
#     with the Netscape Navigator browser then INSERT PublicKeyNS
#     would be included. Likewise, if the users browser was the
#     Microsoft Internet Explorer, INSERT PublicKeyIE would be included.
#
#     optfield - Special substitution variable that should be placed in
#     any certificate field name INSERT where the value may be supplied
#     by the end user. It enables the field to be displayed as optional
#     if desired.
#
#     printablecert - Summary information about the certificate to be
#                     renewed/revoked, such as issuer's name, subject's
#                     name...
#
#     errorinfo - Information about the failing SAF call such as the
#                 return code and reason code.
#
#   Note, depending on where a substitution variable is used, it may
#   not have a valid meaning, e.g., base64cert would be meaningless
#   prior to the certificate being retrieved. The value of
#   [base64cert] would be the empty string (aka NULL) in this case.
#
# ====================================================================
#
# Application - PKISERV
#
# The installation should customize the CONTENT and ADMINCONTENT
# subsections as appropriate
#
# ====================================================================
#
<APPLICATION NAME=PKISERV>
<CONTENT>
<HTML><HEAD>
<TITLE> Web Based Certificate Generation Application </TITLE>
%%-copyright%%
</HEAD>
<BODY>
<H1>PKISERV Certificate Generation Application</H1>
<p>
<A HREF="/PKIServ/cacerts/cacert.der">Install
 our CA certificate into your browser </A>
<H2>Choose one of the following:</H2>
<ul>
<li><h3>Request a new certificate using a model</h3>
<FORM name=mainform METHOD=GET ACTION="/PKIServ/ssl-cgi/catmpl.rexx">
<p> Select the certificate template to use as a model
<SELECT NAME="Template">
 %%1 Year PKI SSL Browser Certificate%%
     <OPTION>1 Year PKI SSL Browser Certificate
 %%1 Year PKI S/MIME Browser Certificate%%
     <OPTION>1 Year PKI S/MIME Browser Certificate
 %%2 Year PKI Browser Certificate For Authenticating To z/OS%%
```

```
     <OPTION>2 Year PKI Browser Certificate For Authenticating To z/OS
 %%5 Year PKI SSL Server Certificate%%
     <OPTION>5 Year PKI SSL Server Certificate
 %%5 Year PKI IPSEC Server (Firewall) Certificate%%
     <OPTION>5 Year PKI IPSEC Server (Firewall) Certificate
 %%5 Year PKI Intermediate CA Certificate%%
     <OPTION>5 Year PKI Intermediate CA Certificate
 %%1 Year SAF Browser Certificate%%
     <OPTION>1 Year SAF Browser Certificate
 %%1 Year SAF Server Certificate%%
     <OPTION>1 Year SAF Server Certificate
</SELECT>
<p>
<INPUT TYPE="submit" VALUE="Request Certificate">
</FORM>
<li><h3>Pick up a previously requested certificate</h3>
<FORM name=selform METHOD=GET ACTION="/PKIServ/ssl-cgi/caretrieve.rexx">
<p> Enter the assigned transaction ID
<INPUT NAME="TransactionId" TYPE="text" SIZE=56 maxlength="56">
<br>Select the certificate return type
<SELECT NAME="Template">
 %%PKI Browser Certificate%%
     <OPTION>PKI Browser Certificate
 %%PKI Server Certificate%%
     <OPTION>PKI Server Certificate
 %%SAF Browser Certificate%%
     <OPTION>SAF Browser Certificate
 %%SAF Server Certificate%%
     <OPTION>SAF Server Certificate
</SELECT>
<p>
<INPUT TYPE="submit" VALUE="Pick up Certificate">
</FORM>
<li><h3>Renew or revoke a previously issued browser certificate</h3>
<FORM name=selform METHOD=GET ACTION="/PKIServ/clientauth-cgi/cadisplay.rexx">
<p>
<SCRIPT LANGUAGE="JavaScript">
<!--
function RenewRevokeAlert(){
var STRING_RenewRevokePrompt=
                  "You will be prompted by the browser to select " +
                  "the certificate you want to renew or revoke. " +
                  "Once you select the certificate you will be " +
                  "given the opportunity to confirm your selection. " +
                  "Note that you can only renew or revoke a single " +
                  "certificate per one browser session. If you wish " +
                  "to renew or revoke another certificate, you must " +
                  "close your browser and restart it.";
 alert(STRING_RenewRevokePrompt);
 return true;
}

function ValidateEntry(){
var STRING_MissingFieldPrompt=
                   "Enter the required field."
var STRING_MissingConfirmPwdPrompt=
                   "Reenter password."
var STRING_UnmatchPwdPrompt=
                   "The passwords do not match. Enter again."
if(document.renform.PassPhrase.value=="") {
alert(STRING_MissingFieldPrompt);
document.renform.PassPhrase.focus();
return true;
}
else if(document.renform.ConfirmPassPhrase.value=="") {
alert(STRING_MissingConfirmPwdPrompt);
document.renform.ConfirmPassPhrase.focus();
```

```
return true;
}
else if(document.renform.PassPhrase.value!=
        document.renform.ConfirmPassPhrase.value){
alert(STRING_UnmatchPwdPrompt);
document.renform.ConfirmPassPhrase.focus();
return true;
}
else {
return false;
}
}
//-->
</SCRIPT>
<INPUT TYPE="submit" VALUE="Renew or Revoke Certificate"
onClick="return RenewRevokeAlert()">
</FORM>
<li><h3>Administrators click here</h3>
# The following action will force userid/pw authentication for administrators
<FORM name=admform METHOD=GET ACTION="/PKIServ/ssl-cgi/auth/admmain.rexx">
# The following action will force client certificate authentication for administrators
#<FORM name=admform METHOD=GET
# ACTION="/PKIServ/clientauth-cgi/auth/admmain.rexx">
<p>
<INPUT TYPE="submit" VALUE="Go to Administration Page">
</FORM>
</ul>
<p> %%-pagefooter%%
</BODY>
</HTML>
</CONTENT>
<RECONTENT>
<HTML><HEAD>
<TITLE> PKISERV Renew or Revoke a Browser Certificate </TITLE>
%%-copyright%%
</HEAD>
<BODY>
<H1>Renew or Revoke a Browser Certificate</H1>
<h3>Here is the certificate you selected:</h3>
<p>
[printablecert]
<h2>If this is the correct certificate, choose one of the following:</h2>
<b>(otherwise you need to restart your browser to pick another certificate)</b>
<ul>
<h3><li>Renew the above certificate</h3>
<FORM name=renform METHOD=POST
 ACTION="/PKIServ/clientauth-cgi/camodify.rexx">

<SCRIPT LANGUAGE="JavaScript">
<!--

function ValidateEntry(){
var STRING_MissingFieldPrompt=
                   "Enter the required field."
var STRING_MissingConfirmPwdPrompt=
                   "Reenter password."
var STRING_UnmatchPwdPrompt=
                   "The passwords do not match. Enter again."
if(document.renform.PassPhrase.value=="") {
alert(STRING_MissingFieldPrompt);
document.renform.PassPhrase.focus();
return true;
}
else if(document.renform.ConfirmPassPhrase.value=="") {
alert(STRING_MissingConfirmPwdPrompt);
document.renform.ConfirmPassPhrase.focus();
return true;
```

```
}
else if(document.renform.PassPhrase.value!=
        document.renform.ConfirmPassPhrase.value){
alert(STRING_UnmatchPwdPrompt);
document.renform.ConfirmPassPhrase.focus();
return true;
}
else {
return false;
}
}
//-->
</SCRIPT>
<INPUT NAME="action" TYPE="hidden" VALUE="renew">
%%PassPhrase%%
<p>
<INPUT TYPE="submit" VALUE="Renew" onClick=
"if(ValidateEntry()) return false; else return true;">
</FORM>
<h3><li>Revoke the above certificate</h3>
<FORM name=revform METHOD=POST
 ACTION="/PKIServ/clientauth-cgi/camodify.rexx">
<INPUT NAME="action" TYPE="hidden" VALUE="revoke">
<INPUT TYPE="submit" VALUE="Revoke">
<SELECT NAME="reason">
 <OPTION Selected VALUE="0">No Reason
 <OPTION VALUE="1">User key was compromised
 <OPTION VALUE="2">CA key was compromised
 <OPTION VALUE="3">User changed affiliation
 <OPTION VALUE="4">Certificate was superseded
 <OPTION VALUE="5">Original use no longer valid
</SELECT>
</ul>
</FORM>
<p>
<FORM METHOD=GET ACTION="/PKIServ/public-cgi/camain.rexx">
<center>
<INPUT TYPE="submit" VALUE="Home Page">
</FORM>
</center>
<p> %%-pagefooter%%
</BODY>
</HTML>
</RECONTENT>
<RESUCCESSCONTENT>
 %%-renewrevokeok%%
</RESUCCESSCONTENT>
<REFAILURECONTENT>
 %%-renewrevokebad%%
</REFAILURECONTENT>
<ADMINHEADER>
<HTML><HEAD>
<TITLE> Web Based Certificate Generation Administration </TITLE>
%%-copyright%%
</HEAD>
<BODY>
</ADMINHEADER>
<ADMINFOOTER>
<p> %%-pagefooter%%
</BODY>
</HTML>
</ADMINFOOTER>
</APPLICATION>


#
# ====================================================================
```

The pkiserv.tmpl certificate templates file

```
#
# Sample Templates - Browser and Server Certificate Requesting
#
# ===================================================================
#
# Template Name - 1 Year SAF Server Certificate
#
# Function - Allows end users to request certificates for servers
# using native SAF certificate generation facilities. The end user
# may provide values for any of the following fields:
#
#  CommonName - optional
#  OrgUnit - required
#  Org - required
#  Locality  - optional
#  StateProv  - optional
#  Country - required
#  AltEmail  - optional
#  AltDomain  - optional
#  AltURI  - optional
#  AltIPAddr  - optional
#  Label - required
#  PublicKey - required (This is the PKCS#10 request)
#
# PKISERV will provide the authenticated client UserId. The certificate
# will be used for handshaking only (e.g., SSL) and is good for 1
# year. The CERTAUTH certificate with Label "Local SAF CA" will be
# used for signing the certificate
#
# ===================================================================
#
<TEMPLATE NAME=1 Year SAF Server Certificate>
<TEMPLATE NAME=SAF Server Certificate>
<CONTENT>
<HTML><HEAD>
<TITLE> Web Based SAF Certificate Generation Application Pg 2</TITLE>
%%-copyright%%
<SCRIPT LANGUAGE="JavaScript">
<!--
function MissingRequiredFAlert(){
var STRING_MissingRequiredFPrompt=
            "Enter the field value that's not optional."

if ((document.serverform.Label.value=="")||
    (document.serverform.OrgUnit.value=="")||
    (document.serverform.Org.value=="")||
    (document.serverform.Country.value=="")||
    (document.serverform.PublicKey.value==""))
{
  alert(STRING_MissingRequiredFPrompt);
  if (document.serverform.OrgUnit.value=="")
    document.serverform.OrgUnit.focus();
  else if (document.serverform.Org.value=="")
    document.serverform.Org.focus();
  else if (document.serverform.Country.value=="")
    document.serverform.Country.focus();
  else if (document.serverform.Label.value=="")
    document.serverform.Label.focus();
  else
    document.serverform.PublicKey.focus();

  return true;
}
else {
return false;
}
}
```

```
//-->
</SCRIPT>

</HEAD>
<BODY>
<H1> SAF Server Certificate 1 Year (Auto Approved)</H1>
<p>
<H2>Choose one of the following:</H2>
<p>
<ul>
<h3><li>Request a New Certificate</h3>
# This ACTION forces userid/pw authentication and runs the task under
# the client's ID
#<FORM NAME=serverform METHOD=POST ACTION=
#                "/PKIServ/ssl-cgi-bin/auth/careq.rexx" onSubmit=

# This ACTION forces userid/pw authentication but runs the task under
# the surrogate ID
 <FORM NAME=serverform METHOD=POST ACTION=
               "/PKIServ/ssl-cgi-bin/surrogateauth/careq.rexx" onSubmit=

# This ACTION is for non z/OS clients. The task runs under the
# surrogate ID
#<FORM NAME=serverform METHOD=POST ACTION=
#               "/PKIServ/ssl-cgi-bin/careq.rexx" onSubmit=
     "if(MissingRequiredFAlert()) return false; else return true;">

<INPUT NAME="Template" TYPE="hidden" VALUE="[tmplname]">
<p> Enter values for the following field(s)
 %%CommonName (Optional)%%
 %%OrgUnit%%
 %%OrgUnit2 (Optional)%%
 %%Org%%
 %%Locality (Optional)%%
 %%StateProv (Optional)%%
 %%Country%%
 %%AltEmail (Optional)%%
 %%AltDomain (Optional)%%
 %%AltURI (Optional)%%
 %%AltIPAddr (Optional)%%
 %%Label%%
 %%PublicKey%%
<p>
<INPUT TYPE="submit" VALUE="Submit certificate request"
ONCLICK="if(MissingRequiredFAlert()) return false; else return true;">
<INPUT TYPE="reset" VALUE="Clear">
</FORM>
<p>
<H3><li>Pick Up a Previously Issued Certificate</H3>

<FORM METHOD=GET ACTION="/PKIServ/ssl-cgi/caretrieve.rexx">
<INPUT NAME="Template" TYPE="hidden" VALUE="[tmplname]">
<INPUT TYPE="submit" VALUE="Retrieve your certificate">
</FORM>
</ul>
<p>%%-pagefooter%%
</BODY>
</HTML>
</CONTENT>
<APPL>
 %%UserId%%
</APPL>
<CONSTANT>
 %%KeyUsage=handshake%%
 %%NotAfter=365%%
 %%SignWith=SAF:CERTAUTH/taca%%
</CONSTANT>
```

The pkiserv.tmpl certificate templates file

```
<SUCCESSCONTENT>
 %%-requestok%%
</SUCCESSCONTENT>
<FAILURECONTENT>
 %%-requestbad%%
</FAILURECONTENT>

<RETRIEVECONTENT>
<HTML><HEAD>
<TITLE> Web Based SAF Certificate Generation Application Pg 3</TITLE>
%%-copyright%%
<SCRIPT LANGUAGE="JavaScript">
<!--
function MissingTransIdAlert(){
var STRING_MissingTransIdPrompt=
             "Enter the transaction ID assigned to the certificate.";
if(document.retrieveform.TransactionId.value==""){
 alert(STRING_MissingTransIdPrompt);
 document.retrieveform.TransactionId.focus();
 return true;
}
else {
 return false;
 }
}
//-->
</SCRIPT>
</HEAD>

<BODY>
<H1> Retrieve Your [tmplname]</H1>
<H3>Please bookmark this page</h3>
<p>Since your certificate may not have been issued yet, we recommend
that you create a bookmark to this location so that when you return to
this bookmark, the browser will display your transaction ID.
This is the easiest way to check your status.

# This ACTION forces userid/pw authentication and runs the task
# under the client's ID
<FORM NAME=retrieveform METHOD=GET ACTION=
      "/PKIServ/ssl-cgi/auth/cagetcert.rexx" onSubmit=
#
# This ACTION forces userid/pw authentication but runs the task
# under the surrogate ID
#<FORM NAME=retrieveform METHOD=GET ACTION=
#      "/PKIServ/ssl-cgi/surrogateauth/cagetcert.rexx" onSubmit=
#
# This ACTION is for non z/OS clients. The task runs under surrogate ID
#<FORM NAME=retrieveform METHOD=GET ACTION=
#      "/PKIServ/ssl-cgi/cagetcert.rexx" onSubmit=
      "if(MissingTransIdAlert()) return false; else return true;">
<INPUT NAME="Template" TYPE="hidden" VALUE="[tmplname]">
<p> Enter values for the following field(s)
 %%TransactionId%%
<p>
<INPUT TYPE="submit" VALUE="Continue">
</FORM>
<p>%%-pagefooter%%
</BODY>
</HTML>
</RETRIEVECONTENT>
<RETURNCERT>
%%-returnpkcs10cert%%
</RETURNCERT>
</TEMPLATE>

#
```

```
# ====================================================================
#
# Template Name - 1 Year SAF Browser Certificate
#
# Function - Allows end users to request certificates for their
# browsers using native SAF certificate generation facilities. The end
# user may provide values for any of the following fields:
#
#  Label - required
#  PublicKey - required (Provided by the browser itself)
#
# PKISERV will provide the authenticated client UserId. The certificate
# will be used for handshaking only (e.g., SSL) and is good for 1
# year. The CERTAUTH certificate with Label "Local SAF CA" will be
# used for signing the certificate. The Subject's Distinguished Name
# will be formed as:
#
#  C=US/O=The Firm/OU=SAF template certificate/
#           OU=Nuts and Bolts Division/CN=<determined by SAF>
#
#  The presence of CommonName without a value tells SAF to determine
#  the CN value from the PGMRNAME field of the user's USER profile.
#  See z/OS Security Server RACF Callable Services Guide
#  for more information
#
# ====================================================================
#
<TEMPLATE NAME=1 Year SAF Browser Certificate>
<TEMPLATE NAME=SAF Browser Certificate>
<CONTENT>
<HTML><HEAD>
<TITLE> Web Based SAF Certificate Generation Application Pg 2</TITLE>
%%-copyright%%
%%-AdditionalHead[browsertype]%%
<SCRIPT LANGUAGE="JavaScript">
<!--
function MissingRequiredFAlert(){
var STRING_MissingRequiredFPrompt=
                   "Enter the field that's required."
if(document.CertReq.Label.value==""){
alert(STRING_MissingRequiredFPrompt);
document.CertReq.Label.focus();
return true;
}
else {
return false;
}
}
//-->
</SCRIPT>
</HEAD>

<BODY>
<H1> SAF Browser Certificate 1 Year (Auto Approved)</H1>
<p>
<H2>Choose one of the following:</H2>
<p>
<ul>
<h3><li>Request a New Certificate</h3>
# This ACTION forces userid/pw authentication and runs the task under
# the client's ID
#<FORM NAME="CertReq" METHOD=POST ACTION=
#                "/PKIServ/ssl-cgi-bin/auth/careq.rexx" onSubmit=

# This ACTION forces userid/pw authentication but runs the task under
# the surrogate ID
 <FORM NAME="CertReq" METHOD=POST ACTION=
```

```
               "/PKIServ/ssl-cgi-bin/surrogateauth/careq.rexx" onSubmit=

# This ACTION is for non z/OS clients. The task runs under the
# surrogate ID
#<FORM NAME="CertReq" METHOD=POST ACTION=
#               "/PKIServ/ssl-cgi-bin/careq.rexx" onSubmit=
    "if(MissingRequiredFAlert()) return false; else return true;">

<INPUT NAME="Template" TYPE="hidden" VALUE="[tmplname]">
<p> Enter values for the following field(s)
 %%Label%%
 %%PublicKey[browsertype]%%
<INPUT TYPE="reset" VALUE="Clear">
</FORM>
<p>
<H3><li>Pick Up a Previously Issued Certificate</H3>
<FORM METHOD=GET ACTION="/PKIServ/ssl-cgi/caretrieve.rexx">
<INPUT NAME="Template" TYPE="hidden" VALUE="[tmplname]">
<INPUT TYPE="submit" VALUE="Retrieve your certificate">
</FORM>
</ul>
<p>%%-pagefooter%%
</BODY>
</HTML>
</CONTENT>
<APPL>
 %%UserId%%
</APPL>
<CONSTANT>
 %%KeyUsage=handshake%%
 %%NotAfter=365%%
 %%OrgUnit=SAF template certificate%%
 %%OrgUnit=Nuts and Bolts Division%%
 %%Org=The Firm%%
 %%Country=US%%
 %%SignWith=SAF:CERTAUTH/taca%%
 %%CommonName=%%
</CONSTANT>
<SUCCESSCONTENT>
 %%-requestok%%
</SUCCESSCONTENT>
<FAILURECONTENT>
 %%-requestbad%%
</FAILURECONTENT>
<RETRIEVECONTENT>

<HTML><HEAD>
%%-copyright%%
<TITLE> Web Based SAF Certificate Generation Application Pg 3</TITLE>
<SCRIPT LANGUAGE="JavaScript">
<!--
function MissingTransIdAlert(){
var STRING_MissingTransIdPrompt=
    "Enter the transaction ID assigned to the certificate.";
if(document.retrieveform.TransactionId.value==""){
 alert(STRING_MissingTransIdPrompt);
 document.retrieveform.TransactionId.focus();
 return true;
}
else {
 return false;
 }
}
//-->
</SCRIPT>
</HEAD>
```

```
<BODY>
<H1> Retrieve Your [tmplname]</H1>
<H3>Please bookmark this page</h3>
<p>Since your certificate may not have been issued yet, we recommend
that you create a bookmark to this location so that when you return to
this bookmark, the browser will display your transaction ID.
This is the easiest way to check your status.

# This ACTION forces userid/pw authentication and runs the task
# under the client's ID
<FORM NAME=retrieveform METHOD=GET ACTION=
      "/PKIServ/ssl-cgi/auth/cagetcert.rexx" onSubmit=
#
# This ACTION forces userid/pw authentication but runs the task
# under the surrogate ID
#<FORM NAME=retrieveform METHOD=GET ACTION=
#      "/PKIServ/ssl-cgi/surrogateauth/cagetcert.rexx" onSubmit=
#
# This ACTION is for non z/OS clients. The task runs under surrogate ID
#<FORM NAME=retrieveform METHOD=GET ACTION=
#      "/PKIServ/ssl-cgi/cagetcert.rexx" onSubmit=
      "if(MissingTransIdAlert()) return false; else return true;">
<INPUT NAME="Template" TYPE="hidden" VALUE="[tmplname]">
 %%TransactionId%%
<p>
<INPUT TYPE="submit" VALUE="Retrieve and Install Certificate">
</FORM>
<p>
<H2>To check that your certificate installed properly, follow the
procedure below:</h2>
<p><B>Netscape V6</B> - Click Edit->Preferences, then Privacy and Security->
Certificates. Click the Manage Certificates button to start the Certificate Manager.
Your new certificate should appear in the Your Certificates list.
Select it then click View to see more information.
<p><B>Netscape V4</B> - Click the Security button, then Certificates->
Yours. Your certificate should appear in the list.  Select it then
click Verify.
<p><B>Internet Explorer V5</B> - Click Tools->Internet Options, then
Content, Certificates.
Your certificate should appear in the Personal list. Click Advanced to
see additional information.
<p>
<FORM METHOD=GET ACTION="/PKIServ/public-cgi/camain.rexx">
<INPUT NAME="Template" TYPE="hidden" VALUE="[tmplname]">
<INPUT TYPE="submit" VALUE="Home page">
</FORM>
<p>%%-pagefooter%%
</BODY>
</HTML>
</RETRIEVECONTENT>
<RETURNCERT>
%%returnbrowsercert[browsertype]%%
</RETURNCERT>
</TEMPLATE>

#
#
# ====================================================================
#
# Template Name - 1 Year PKI SSL Browser Certificate
#
# Function - Creates a 1 year certificate good for general SSL client
#            authentication using a browser. If approved, the
#            certificate becomes valid after it's requested.
#            (You may delay the valid date by specifying a non zero
#             number for the value of 'NotBefore',
#             e.g. NotBefore=5. That means if the request is approved,
```

The pkiserv.tmpl certificate templates file

```
#             the certificate will become valid 5 days after it's
#             requested.)
#
#            These certificates will be stored in LDAP if The O= and
#            OU= suffixes have already been created
#
# Other than the user input fields, all other information is hard coded.
#
# User input fields:
#  CommonName - required
#  Requestor - optional
#  PassPhrase - required
#  PublicKey - required (Provided by the browser itself)
#
#  RACF userid/password authentication : not require
#  Administrator approval              : require
#
# ==================================================================
#
<TEMPLATE NAME=1 Year PKI SSL Browser Certificate>
<TEMPLATE NAME=PKI Browser Certificate>
<NICKNAME=1YBSSL>
<CONTENT>
<HTML><HEAD>
<TITLE> Web Based PKIX Certificate Generation Application Pg 2</TITLE>
%%-copyright%%
%%-AdditionalHead[browsertype]%%
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidateEntry(){
var STRING_MissingFieldPrompt=
                  "Enter the required field."
var STRING_MissingConfirmPwdPrompt=
                  "Reenter password."
var STRING_UnmatchPwdPrompt=
                  "The passwords do not match. Enter again."
if(document.CertReq.CommonName.value=="") {
alert(STRING_MissingFieldPrompt);
document.CertReq.CommonName.focus();
return true;
}
else if(document.CertReq.PassPhrase.value=="") {
alert(STRING_MissingFieldPrompt);
document.CertReq.PassPhrase.focus();
return true;
}
else if(document.CertReq.ConfirmPassPhrase.value=="") {
alert(STRING_MissingConfirmPwdPrompt);
document.CertReq.ConfirmPassPhrase.focus();
return true;
}
else if(document.CertReq.PassPhrase.value!=
        document.CertReq.ConfirmPassPhrase.value){
alert(STRING_UnmatchPwdPrompt);
document.CertReq.ConfirmPassPhrase.focus();
return true;
}
else {
return false;
}
}
//-->
</SCRIPT>
</HEAD>

<BODY>
<H1>1 Year SSL Browser Certificate</H1>
```

```
<p>
<H2>Choose one of the following:</H2>
<p>
<ul>
<h3><li>Request a New Certificate</h3>
# This ACTION forces userid/pw authentication and runs the task under
# the client's ID
#<FORM NAME="CertReq" METHOD=POST ACTION=
#                "/PKIServ/ssl-cgi-bin/auth/careq.rexx" onSubmit=

# This ACTION forces userid/pw authentication but runs the task under
# the surrogate ID
#<FORM NAME="CertReq" METHOD=POST ACTION=
#              "/PKIServ/ssl-cgi-bin/surrogateauth/careq.rexx" onSubmit=

# This ACTION is for non z/OS clients. The task runs under the
# surrogate ID
 <FORM NAME="CertReq" METHOD=POST ACTION=
               "/PKIServ/ssl-cgi-bin/careq.rexx" onSubmit=
    "if(ValidateEntry()) return false; else return true;">

<INPUT NAME="Template" TYPE="hidden" VALUE="[tmplname]">
<p> Enter values for the following field(s)
 %%CommonName%%
 %%Requestor (optional)%%
 %%PassPhrase%%
 %%PublicKey2[browsertype]%%
<INPUT TYPE="reset" VALUE="Clear">
</FORM>
<p>
<H3><li>Pick Up a Previously Issued Certificate</H3>
<FORM METHOD=GET ACTION="/PKIServ/ssl-cgi/caretrieve.rexx">
<INPUT NAME="Template" TYPE="hidden" VALUE="[tmplname]">
<INPUT TYPE="submit" VALUE="Retrieve your certificate">
</FORM>
</ul>
<p>%%-pagefooter%%
</BODY>
</HTML>
</CONTENT>
<CONSTANT>
 %%NotBefore=0%%
 %%NotAfter=365%%
 %%KeyUsage=handshake%%
 %%OrgUnit=Class 1 Internet Certificate CA%%
 %%Org=The Firm%%
 %%SignWith=PKI:%%
</CONSTANT>
<ADMINAPPROVE>
 %%CommonName (Optional)%%
 %%OrgUnit (Optional)%%
 %%OrgUnit (Optional)%%
 %%Org (Optional)%%
 %%NotBefore (optional)%%
 %%NotAfter (Optional)%%
 %%KeyUsage (Optional)%%
 %%HostIdMap (Optional)%%
 %%HostIdMap (Optional)%%
 %%HostIdMap (Optional)%%
 %%HostIdMap (Optional)%%
</ADMINAPPROVE>
<SUCCESSCONTENT>
 %%-requestok%%
</SUCCESSCONTENT>
<FAILURECONTENT>
 %%-requestbad%%
</FAILURECONTENT>
```

The pkiserv.tmpl certificate templates file

```
<RETRIEVECONTENT>
<HTML><HEAD>
%%-copyright%%
<TITLE> Web Based PKIX Certificate Generation Application Pg 3</TITLE>
<SCRIPT LANGUAGE="JavaScript">
<!--
function MissingTransIdAlert(){
var STRING_MissingTransIdPrompt=
    "Enter the transaction ID assigned to the certificate.";
if(document.retrieveform.TransactionId.value==""){
 alert(STRING_MissingTransIdPrompt);
 document.retrieveform.TransactionId.focus();
 return true;
}
else {
 return false;
 }
}
//-->
</SCRIPT>
</HEAD>

<BODY>
<H1> Retrieve Your [tmplname]</H1>
<H3>Please bookmark this page</h3>
<p>Since your certificate may not have been issued yet, we recommend
that you create a bookmark to this location so that when you return to
this bookmark, the browser will display your transaction ID.
This is the easiest way to check your status.

# This ACTION forces userid/pw authentication and runs the task
# under the client's ID
#<FORM NAME=retrieveform METHOD=POST ACTION=
#     "/PKIServ/ssl-cgi-bin/auth/cagetcert.rexx" onSubmit=
#
# This ACTION forces userid/pw authentication but runs the task
# under the surrogate ID
#<FORM NAME=retrieveform METHOD=POST ACTION=
#      "/PKIServ/ssl-cgi-bin/surrogateauth/cagetcert.rexx" onSubmit=
#
# This ACTION is for non z/OS clients. The task runs under surrogate ID
 <FORM NAME=retrieveform METHOD=POST ACTION=
       "/PKIServ/ssl-cgi-bin/cagetcert.rexx" onSubmit=
      "if(MissingTransIdAlert()) return false; else return true;">
<INPUT NAME="Template" TYPE="hidden" VALUE="[tmplname]">
 %%TransactionId%%
 %%ChallengePassPhrase (optional)%%
<p>
<INPUT TYPE="submit" VALUE="Retrieve and Install Certificate">
</FORM>
<p>
<H2>To check that your certificate installed properly, follow the
procedure below:</h2>
<p><B>Netscape V6</B> - Click Edit->Preferences, then Privacy and Security->
Certificates. Click the Manage Certificates button to start the Certificate Manager.
Your new certificate should appear in the Your Certificates list.
Select it then click View to see more information.
<p><B>Netscape V4</B> - Click the Security button, then Certificates->
Yours. Your certificate should appear in the list.  Select it then
click Verify.
<p><B>Internet Explorer V5</B> - Click Tools->Internet Options, then
Content, Certificates.
Your certificate should appear in the Personal list. Click Advanced to
see additional information.
<p>
<FORM METHOD=GET ACTION="/PKIServ/public-cgi/camain.rexx">
```

```
<INPUT NAME="Template" TYPE="hidden" VALUE="[tmplname]">
<INPUT TYPE="submit" VALUE="Home page">
</FORM>
<p>%%-pagefooter%%
</BODY>
</HTML>
</RETRIEVECONTENT>
<RETURNCERT>
%%returnbrowsercert[browsertype]%%
</RETURNCERT>
</TEMPLATE>

#
#
# ====================================================================
#
# Template Name - 1 Year PKI S/MIME Browser Certificate
#
# Function - Creates a 1 year certificate good for S/MIME
#            authentication using a browser. If approved, the
#            certificate becomes valid after it's requested.
#            (You may delay the valid date by specifying a non zero
#             number for the value of 'NotBefore',
#             e.g. NotBefore=5. That means if the request is approved,
#             the certificate will become valid 5 days after it's
#             requested.)
#
#            These certificates will be stored in LDAP if The O= and
#            OU= suffixes have already been created
#
# Other than the user input fields, all other information is hard coded.
#
# User input fields:
#  CommonName - required
#  AltEmail - required
#  Requestor - optional
#  PassPhrase - required
#  PublicKey - required (Provided by the browser itself)
#
#  RACF userid/password authentication : not require
#  Administrator approval              : require
#
#
# ====================================================================
#
<TEMPLATE NAME=1 Year PKI S/MIME Browser Certificate>
<TEMPLATE NAME=PKI Browser Certificate>
<NICKNAME=1YBSM>
<CONTENT>
<HTML><HEAD>
<TITLE> Web Based PKIX Certificate Generation Application Pg 2</TITLE>
%%-copyright%%
%%-AdditionalHead[browsertype]%%
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidateEntry(){
var STRING_MissingFieldPrompt=
                  "Enter the required field."
var STRING_MissingConfirmPwdPrompt=
                  "Reenter password."
var STRING_MissingPwdPrompt=
                  "Need to enter password before confirm it."
var STRING_UnmatchPwdPrompt=
                  "The passwords do not match. Enter again."
if(document.CertReq.CommonName.value=="") {
alert(STRING_MissingFieldPrompt);
document.CertReq.CommonName.focus();
```

```
return true;
}
if(document.CertReq.AltEmail.value=="") {
alert(STRING_MissingFieldPrompt);
document.CertReq.AltEmail.focus();
return true;
}
else if(document.CertReq.PassPhrase.value=="") {
alert(STRING_MissingFieldPrompt);
document.CertReq.PassPhrase.focus();
return true;
}
else if(document.CertReq.ConfirmPassPhrase.value=="") {
alert(STRING_MissingConfirmPwdPrompt);
document.CertReq.ConfirmPassPhrase.focus();
return true;
}
else if(document.CertReq.PassPhrase.value!=
        document.CertReq.ConfirmPassPhrase.value){
alert(STRING_UnmatchPwdPrompt);
document.CertReq.ConfirmPassPhrase.focus();
return true;
}
else {
return false;
}
}
//-->
</SCRIPT>
</HEAD>

<BODY>
<H1>1 Year S/MIME Browser Certificate</H1>
<p>
<H2>Choose one of the following:</H2>
<p>
<ul>
<h3><li>Request a New Certificate</h3>
# This ACTION forces userid/pw authentication and runs the task under
# the client's ID
#<FORM NAME="CertReq" METHOD=POST ACTION=
#                "/PKIServ/ssl-cgi-bin/auth/careq.rexx" onSubmit=

# This ACTION forces userid/pw authentication but runs the task under
# the surrogate ID
#<FORM NAME="CertReq" METHOD=POST ACTION=
#              "/PKIServ/ssl-cgi-bin/surrogateauth/careq.rexx" onSubmit=

# This ACTION is for non z/OS clients. The task runs under the
# surrogate ID
 <FORM NAME="CertReq" METHOD=POST ACTION=
               "/PKIServ/ssl-cgi-bin/careq.rexx" onSubmit=
    "if(ValidateEntry()) return false; else return true;">

<INPUT NAME="Template" TYPE="hidden" VALUE="[tmplname]">
<p> Enter values for the following field(s)
 %%CommonName%%
 %%AltEmail%%
 %%Requestor (optional)%%
 %%PassPhrase%%
 %%PublicKey2[browsertype]%%
<INPUT TYPE="reset" VALUE="Clear">
</FORM>
<p>
<H3><li>Pick Up a Previously Issued Certificate</H3>
<FORM METHOD=GET ACTION="/PKIServ/ssl-cgi/caretrieve.rexx">
<INPUT NAME="Template" TYPE="hidden" VALUE="[tmplname]">
```

```
<INPUT TYPE="submit" VALUE="Retrieve your certificate">
</FORM>
</ul>
<p>%%-pagefooter%%
</BODY>
</HTML>
</CONTENT>
<CONSTANT>
 %%NotBefore=0%%
 %%NotAfter=365%%
 %%KeyUsage=handshake%%
 %%OrgUnit=Class 1 Internet Certificate CA%%
 %%Org=The Firm%%
 %%SignWith=PKI:%%
</CONSTANT>
<ADMINAPPROVE>
 %%CommonName (Optional)%%
 %%OrgUnit (Optional)%%
 %%Org (Optional)%%
 %%AltEmail (optional)%%
 %%NotBefore (optional)%%
 %%NotAfter (Optional)%%
 %%KeyUsage (Optional)%%
</ADMINAPPROVE>
<SUCCESSCONTENT>
 %%-requestok%%
</SUCCESSCONTENT>
<FAILURECONTENT>
 %%-requestbad%%
</FAILURECONTENT>

<RETRIEVECONTENT>
<HTML><HEAD>
%%-copyright%%
<TITLE> Web Based PKIX Certificate Generation Application Pg 3</TITLE>
<SCRIPT LANGUAGE="JavaScript">
<!--
function MissingTransIdAlert(){
var STRING_MissingTransIdPrompt=
    "Enter the transaction ID assigned to the certificate.";
if(document.retrieveform.TransactionId.value==""){
 alert(STRING_MissingTransIdPrompt);
 document.retrieveform.TransactionId.focus();
 return true;
}
else {
 return false;
 }
}
//-->
</SCRIPT>
</HEAD>

<BODY>
<H1> Retrieve Your [tmplname]</H1>
<H3>Please bookmark this page</h3>
<p>Since your certificate may not have been issued yet, we recommend
that you create a bookmark to this location so that when you return to
this bookmark, the browser will display your transaction ID.
This is the easiest way to check your status.

# This ACTION forces userid/pw authentication and runs the task
# under the client's ID
#<FORM NAME=retrieveform METHOD=POST ACTION=
#      "/PKIServ/ssl-cgi-bin/auth/cagetcert.rexx" onSubmit=
#
# This ACTION forces userid/pw authentication but runs the task
```

The pkiserv.tmpl certificate templates file

```
# under the surrogate ID
#<FORM NAME=retrieveform METHOD=POST ACTION=
#      "/PKIServ/ssl-cgi-bin/surrogateauth/cagetcert.rexx" onSubmit=
#
# This ACTION is for non z/OS clients. The task runs under surrogate ID
 <FORM NAME=retrieveform METHOD=POST ACTION=
       "/PKIServ/ssl-cgi-bin/cagetcert.rexx" onSubmit=
      "if(MissingTransIdAlert()) return false; else return true;">
<INPUT NAME="Template" TYPE="hidden" VALUE="[tmplname]">
 %%TransactionId%%
 %%ChallengePassPhrase (optional)%%
<p>
<INPUT TYPE="submit" VALUE="Retrieve and Install Certificate">
</FORM>
<p>
<H2>To check that your certificate installed properly, follow the
procedure below:</h2>
<p><B>Netscape V6</B> - Click Edit->Preferences, then Privacy and Security->
Certificates. Click the Manage Certificates button to start the Certificate Manager.
Your new certificate should appear in the Your Certificates list.
Select it then click View to see more information.
<p><B>Netscape V4</B> - Click the Security button, then Certificates->
Yours. Your certificate should appear in the list.  Select it then
click Verify.
<p><B>Internet Explorer V5</B> - Click Tools->Internet Options, then
Content, Certificates.
Your certificate should appear in the Personal list. Click Advanced to
see additional information.
<p>
<FORM METHOD=GET ACTION="/PKIServ/public-cgi/camain.rexx">
<INPUT NAME="Template" TYPE="hidden" VALUE="[tmplname]">
<INPUT TYPE="submit" VALUE="Home page">
</FORM>
<p>%%-pagefooter%%
</BODY>
</HTML>
</RETRIEVECONTENT>
<RETURNCERT>
%%returnbrowsercert[browsertype]%%
</RETURNCERT>
</TEMPLATE>
#
#
# ====================================================================
#
# Template Name - 2 Year PKI Browser Certificate For Authenticating
#                 to z/OS
#
# Function - Creates a 2 year certificate good for authenticating to
#            z/OS. If approved, the certificate becomes valid after
#            it's requested.
#            (You may delay the valid date by specifying a non zero
#             number for the value of 'NotBefore',
#             e.g. NotBefore=5. That means if the request is approved,
#             the certificate will become valid 5 days after it's
#             requested.)
#            HostIdMap is formed by putting %%Userid%% and
#            %%HostIdMap=@host-name in the APPL section.
#
#            These certificates will be stored in LDAP if The O= and
#            OU= suffixes have already been created
#
# Other than the user input fields, all other information is hard coded.
#
# User input fields:
#  Requestor - optional
#  PassPhrase - required
```

```
#  PublicKey - required (Provided by the browser itself)
#
#  The presence of CommonName without a value tells SAF to determine
#  the CN value from the PGMRNAME field of the user's USER profile.
#  See z/OS Security Server RACF Callable Services Guide
#  for more information
#
#  RACF userid/password authentication : require
#  Administrator approval              : not require
#
#
# ====================================================================
#
<TEMPLATE NAME=2 Year PKI Browser Certificate For Authenticating To z/OS>
<TEMPLATE NAME=PKI Browser Certificate>
<NICKNAME=2YBZOS>
<CONTENT>
<HTML><HEAD>
<TITLE> Web Based PKIX Certificate Generation Application Pg 2</TITLE>
%%-copyright%%
%%-AdditionalHead[browsertype]%%
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidateEntry(){
var STRING_MissingFieldPrompt=
                   "Enter the required field."
var STRING_MissingConfirmPwdPrompt=
                   "Reenter password."
var STRING_MissingPwdPrompt=
                   "Need to enter password before confirm it."
var STRING_UnmatchPwdPrompt=
                   "The passwords do not match. Enter again."
if(document.CertReq.PassPhrase.value=="") {
alert(STRING_MissingFieldPrompt);
document.CertReq.PassPhrase.focus();
return true;
}
else if(document.CertReq.ConfirmPassPhrase.value=="") {
alert(STRING_MissingConfirmPwdPrompt);
document.CertReq.ConfirmPassPhrase.focus();
return true;
}
else if(document.CertReq.PassPhrase.value!=
        document.CertReq.ConfirmPassPhrase.value){
alert(STRING_UnmatchPwdPrompt);
document.CertReq.ConfirmPassPhrase.focus();
return true;
}
else {
return false;
}
}
//-->
</SCRIPT>
</HEAD>

<BODY>
<H1>2 Year Browser Certificate For Authenticating To z/OS</H1>
<p>
<H2>Choose one of the following:</H2>
<p>
<ul>
<h3><li>Request a New Certificate</h3>
# This ACTION forces userid/pw authentication and runs the task under
# the client's ID
#<FORM NAME="CertReq" METHOD=POST ACTION=
#                "/PKIServ/ssl-cgi-bin/auth/careq.rexx" onSubmit=
```

The pkiserv.tmpl certificate templates file

```
# This ACTION forces userid/pw authentication but runs the task under
# the surrogate ID
 <FORM NAME="CertReq" METHOD=POST ACTION=
              "/PKIServ/ssl-cgi-bin/surrogateauth/careq.rexx" onSubmit=

# This ACTION is for non z/OS clients. The task runs under the
# surrogate ID
#<FORM NAME="CertReq" METHOD=POST ACTION=
#              "/PKIServ/ssl-cgi-bin/careq.rexx" onSubmit=
    "if(ValidateEntry()) return false; else return true;">

<INPUT NAME="Template" TYPE="hidden" VALUE="[tmplname]">
<p> Enter values for the following field(s)
 %%Requestor (optional)%%
 %%PassPhrase%%
 %%PublicKey2[browsertype]%%
<INPUT TYPE="reset" VALUE="Clear">
</FORM>
<p>
<H3><li>Pick Up a Previously Issued Certificate</H3>
<FORM METHOD=GET ACTION="/PKIServ/ssl-cgi/caretrieve.rexx">
<INPUT NAME="Template" TYPE="hidden" VALUE="[tmplname]">
<INPUT TYPE="submit" VALUE="Retrieve your certificate">
</FORM>
</ul>
<p>%%-pagefooter%%
</BODY>
</HTML>
</CONTENT>
<APPL>
 %%UserId%%
 %%HostIdMap=@host-name%%
</APPL>
<CONSTANT>
 %%NotBefore=0%%
 %%NotAfter=730%%
 %%KeyUsage=handshake%%
 %%OrgUnit=Class 1 Internet Certificate CA%%
 %%Org=The Firm%%
 %%SignWith=PKI:%%
 %%CommonName=%%
</CONSTANT>
<SUCCESSCONTENT>
 %%-requestok%%
</SUCCESSCONTENT>
<FAILURECONTENT>
 %%-requestbad%%
</FAILURECONTENT>

<RETRIEVECONTENT>
<HTML><HEAD>
%%-copyright%%
<TITLE> Web Based PKIX Certificate Generation Application Pg 3</TITLE>
<SCRIPT LANGUAGE="JavaScript">
<!--
function MissingTransIdAlert(){
var STRING_MissingTransIdPrompt=
    "Enter the transaction ID assigned to the certificate.";
if(document.retrieveform.TransactionId.value==""){
 alert(STRING_MissingTransIdPrompt);
 document.retrieveform.TransactionId.focus();
 return true;
}
else {
 return false;
 }
```

```
}
//-->
</SCRIPT>
</HEAD>

<BODY>
<H1> Retrieve Your [tmplname]</H1>
<H3>Please bookmark this page</h3>
<p>Since your certificate may not have been issued yet, we recommend
that you create a bookmark to this location so that when you return to
this bookmark, the browser will display your transaction ID.
This is the easiest way to check your status.

# This ACTION forces userid/pw authentication and runs the task
# under the client's ID
#<FORM NAME=retrieveform METHOD=POST ACTION=
#      "/PKIServ/ssl-cgi-bin/auth/cagetcert.rexx" onSubmit=
#
# This ACTION forces userid/pw authentication but runs the task
# under the surrogate ID
 <FORM NAME=retrieveform METHOD=POST ACTION=
       "/PKIServ/ssl-cgi-bin/surrogateauth/cagetcert.rexx" onSubmit=
#
# This ACTION is for non z/OS clients. The task runs under surrogate ID
#<FORM NAME=retrieveform METHOD=POST ACTION=
#      "/PKIServ/ssl-cgi-bin/cagetcert.rexx" onSubmit=
      "if(MissingTransIdAlert()) return false; else return true;">
<INPUT NAME="Template" TYPE="hidden" VALUE="[tmplname]">
 %%TransactionId%%
 %%ChallengePassPhrase (optional)%%
<p>
<INPUT TYPE="submit" VALUE="Retrieve and Install Certificate">
</FORM>
<p>
<H2>To check that your certificate installed properly, follow the
procedure below:</h2>
<p><B>Netscape V6</B> - Click Edit->Preferences, then Privacy and Security->
Certificates. Click the Manage Certificates button to start the Certificate Manager.
Your new certificate should appear in the Your Certificates list.
Select it then click View to see more information.
<p><B>Netscape V4</B> - Click the Security button, then Certificates->
Yours. Your certificate should appear in the list.  Select it then
click Verify.
<p><B>Internet Explorer V5</B> - Click Tools->Internet Options, then
Content, Certificates.
Your certificate should appear in the Personal list. Click Advanced to
see additional information.
<p>
<FORM METHOD=GET ACTION="/PKIServ/public-cgi/camain.rexx">
<INPUT NAME="Template" TYPE="hidden" VALUE="[tmplname]">
<INPUT TYPE="submit" VALUE="Home page">
</FORM>
<p>%%-pagefooter%%
</BODY>
</HTML>
</RETRIEVECONTENT>
<RETURNCERT>
%%returnbrowsercert[browsertype]%%
</RETURNCERT>
</TEMPLATE>
#
#
# ====================================================================
#
# Template Name - 5 Year PKI SSL Server Certificate
#
# Function - Creates a 5 year Server certificate. If approved, the
```

The pkiserv.tmpl certificate templates file

```
#            certificate becomes valid after it's requested.
#            (You may delay the valid date by specifying a non zero
#             number for the value of 'NotBefore',
#             e.g. NotBefore=5. That means if the request is approved,
#             the certificate will become valid 5 days after it's
#             requested.)
#
#            These certificates will be stored in LDAP if The O= and
#            OU= suffixes have already been created
#
# Other than the user input fields, all other information is hard coded.
#
# User input fields:
#  CommonName - optional
#  OrgUnit - optional
#  Org - optional
#  Locality  - optional
#  StateProv  - optional
#  Country - optional
#  AltEmail  - optional
#  AltDomain  - optional
#  AltURI  - optional
#  AltIPAddr  - optional
#  PassPhrase - required
#  PublicKey - required (This is the PKCS#10 request)
#
#  RACF userid/password authentication : not require
#  Administrator approval              : require
#
#
# ======================================================================
#
<TEMPLATE NAME=5 Year PKI SSL Server Certificate>

<TEMPLATE NAME=PKI Server Certificate>

<NICKNAME=5YSSSL>
<CONTENT>
<HTML><HEAD>
<TITLE> Web Based PKIX Certificate Generation Application Pg 2</TITLE>
%%-copyright%%
<SCRIPT LANGUAGE="JavaScript">
<!--
function MissingRequiredFAlert(){
var STRING_MissingRequiredFPrompt=
            "Enter the field value that's not optional."
var STRING_MissingConfirmPwdPrompt=
                   "Reenter password."
var STRING_MissingPwdPrompt=
                   "Need to enter password before confirm it."
var STRING_UnmatchPwdPrompt=
                   "The passwords do not match. Enter again."

if (document.serverform.PassPhrase.value=="")
{
  alert(STRING_MissingRequiredFPrompt);
  document.serverform.PassPhrase.focus();
  return true;
}
else if(document.serverform.ConfirmPassPhrase.value=="") {
alert(STRING_MissingConfirmPwdPrompt);
document.serverform.ConfirmPassPhrase.focus();
return true;
}
else if(document.serverform.PassPhrase.value!=
        document.serverform.ConfirmPassPhrase.value){
alert(STRING_UnmatchPwdPrompt);
```

```
document.serverform.ConfirmPassPhrase.focus();
return true;
}
else if(document.serverform.PublicKey.value=="") {
alert(STRING_MissingRequiredFPrompt);
document.serverform.PublicKey.focus();
return true;
}
else {
return false;
}
}
//-->
</SCRIPT>

</HEAD>
<BODY>
<H1> 5 Year PKI SSL Server Certificate</H1>
<p>
<H2>Choose one of the following:</H2>
<p>
<ul>
<h3><li>Request a New Certificate</h3>
# This ACTION forces userid/pw authentication and runs the task under
# the client's ID
#<FORM NAME=serverform METHOD=POST ACTION=
#                "/PKIServ/ssl-cgi-bin/auth/careq.rexx" onSubmit=

# This ACTION forces userid/pw authentication but runs the task under
# the surrogate ID
#<FORM NAME=serverform METHOD=POST ACTION=
#              "/PKIServ/ssl-cgi-bin/surrogateauth/careq.rexx" onSubmit=

# This ACTION is for non z/OS clients. The task runs under the
# surrogate ID
 <FORM NAME=serverform METHOD=POST ACTION=
               "/PKIServ/ssl-cgi-bin/careq.rexx" onSubmit=
     "if(MissingRequiredFAlert()) return false; else return true;">

<INPUT NAME="Template" TYPE="hidden" VALUE="[tmplname]">
<p> Enter values for the following field(s)
 %%CommonName (Optional)%%
 %%OrgUnit (Optional)%%
 %%OrgUnit2 (Optional)%%
 %%Org (Optional)%%
 %%Locality (Optional)%%
 %%StateProv (Optional)%%
 %%Country (Optional)%%
 %%AltEmail (Optional)%%
 %%AltDomain (Optional)%%
 %%AltURI (Optional)%%
 %%AltIPAddr (Optional)%%
 %%Requestor (Optional)%%
 %%PassPhrase%%
 %%PublicKey%%
<p>
<INPUT TYPE="submit" VALUE="Submit certificate request"
ONCLICK="if(MissingRequiredFAlert()) return false; else return true;">
<INPUT TYPE="reset" VALUE="Clear">
</FORM>
<p>
<H3><li>Pick Up a Previously Issued Certificate</H3>

<FORM METHOD=GET ACTION="/PKIServ/ssl-cgi/caretrieve.rexx">
<INPUT NAME="Template" TYPE="hidden" VALUE="[tmplname]">
<INPUT TYPE="submit" VALUE="Retrieve your certificate">
</FORM>
```

```
</ul>
<p>%%-pagefooter%%
</BODY>
</HTML>
</CONTENT>
<CONSTANT>
 %%NotBefore=0%%
 %%NotAfter=1825%%
 %%KeyUsage=handshake%%
 %%SignWith=PKI:%%
</CONSTANT>
<ADMINAPPROVE>
 %%CommonName (Optional)%%
 %%OrgUnit (Optional)%%
 %%OrgUnit (Optional)%%
 %%Org (Optional)%%
 %%Locality (Optional)%%
 %%StateProv (Optional)%%
 %%Country (Optional)%%
 %%AltEmail (Optional)%%
 %%AltDomain (Optional)%%
 %%AltURI (Optional)%%
 %%AltIPAddr (Optional)%%
 %%NotBefore (optional)%%
 %%NotAfter (Optional)%%
 %%KeyUsage (Optional)%%
</ADMINAPPROVE>
<SUCCESSCONTENT>
 %%-requestok%%
</SUCCESSCONTENT>
<FAILURECONTENT>
 %%-requestbad%%
</FAILURECONTENT>

<RETRIEVECONTENT>
<HTML><HEAD>
<TITLE> Web Based PKIX Certificate Generation Application Pg 3</TITLE>
%%-copyright%%
<SCRIPT LANGUAGE="JavaScript">
<!--
function MissingTransIdAlert(){
var STRING_MissingTransIdPrompt=
             "Enter the transaction ID assigned to the certificate.";
if(document.retrieveform.TransactionId.value==""){
 alert(STRING_MissingTransIdPrompt);
 document.retrieveform.TransactionId.focus();
 return true;
}
else {
 return false;
 }
}
//-->
</SCRIPT>
</HEAD>

<BODY>
<H1> Retrieve Your [tmplname]</H1>
<H3>Please bookmark this page</h3>
<p>Since your certificate may not have been issued yet, we recommend
that you create a bookmark to this location so that when you return to
this bookmark, the browser will display your transaction ID.
This is the easiest way to check your status.

# This ACTION forces userid/pw authentication and runs the task
# under the client's ID
#<FORM NAME=retrieveform METHOD=POST ACTION=
```

```
#      "/PKIServ/ssl-cgi-bin/auth/cagetcert.rexx" onSubmit=
#
# This ACTION forces userid/pw authentication but runs the task
# under the surrogate ID
#<FORM NAME=retrieveform METHOD=POST ACTION=
#      "/PKIServ/ssl-cgi-bin/surrogateauth/cagetcert.rexx" onSubmit=
#
# This ACTION is for non z/OS clients. The task runs under surrogate ID
 <FORM NAME=retrieveform METHOD=POST ACTION=
       "/PKIServ/ssl-cgi-bin/cagetcert.rexx" onSubmit=
      "if(MissingTransIdAlert()) return false; else return true;">
<INPUT NAME="Template" TYPE="hidden" VALUE="[tmplname]">
<p> Enter values for the following field(s)
 %%TransactionId%%
 %%ChallengePassPhrase (optional)%%
<p>
<INPUT TYPE="submit" VALUE="Continue">
</FORM>
<p>%%-pagefooter%%
</BODY>
</HTML>
</RETRIEVECONTENT>
<RETURNCERT>
%%-returnpkcs10cert%%
</RETURNCERT>
</TEMPLATE>

#
# ============================================================================
#
# Template Name - 5 Year PKI IPSEC Server (Firewall) Certificate
#
# Function - Creates a 5 year Server certificate. If approved, the
#            certificate becomes valid after it's requested.
#            (You may delay the valid date by specifying a non zero
#             number for the value of 'NotBefore',
#             e.g. NotBefore=5. That means if the request is approved,
#             the certificate will become valid 5 days after it's
#             requested.)
#
#            These certificates will be stored in LDAP if The O= and
#            OU= suffixes have already been created
#
#
# Other than the user input fields, all other information is hard coded.
#
# User input fields:
#  CommonName - optional
#  OrgUnit - optional
#  Org - optional
#  Locality  - optional
#  StateProv  - optional
#  Country - optional
#  AltEmail  - optional
#  AltDomain  - optional
#  AltURI  - optional
#  AltIPAddr  - optional
#  PassPhrase - required
#  PublicKey - required (This is the PKCS#10 request)
#
#  RACF userid/password authentication : not require
#  Administrator approval              : require
#
#
# ============================================================================
#
<TEMPLATE NAME=5 Year PKI IPSEC Server (Firewall) Certificate>
```

The pkiserv.tmpl certificate templates file

```
<TEMPLATE NAME=PKI Server Certificate>
<NICKNAME=5YSIPS>
<CONTENT>
<HTML><HEAD>
<TITLE> Web Based PKIX Certificate Generation Application Pg 2</TITLE>
%%-copyright%%
<SCRIPT LANGUAGE="JavaScript">
<!--
function MissingRequiredFAlert(){
var STRING_MissingRequiredFPrompt=
            "Enter the field value that's not optional."
var STRING_MissingConfirmPwdPrompt=
                    "Reenter password."
var STRING_MissingPwdPrompt=
                    "Need to enter password before confirm it."
var STRING_UnmatchPwdPrompt=
                    "The passwords do not match. Enter again."

if (document.serverform.PassPhrase.value=="")
{
  alert(STRING_MissingRequiredFPrompt);
  document.serverform.PassPhrase.focus();
  return true;
}
else if(document.serverform.ConfirmPassPhrase.value=="") {
alert(STRING_MissingConfirmPwdPrompt);
document.serverform.ConfirmPassPhrase.focus();
return true;
}
else if(document.serverform.PassPhrase.value!=
        document.serverform.ConfirmPassPhrase.value){
alert(STRING_UnmatchPwdPrompt);
document.serverform.ConfirmPassPhrase.focus();
return true;
}
else if(document.serverform.PublicKey.value=="") {
alert(STRING_MissingRequiredFPrompt);
document.serverform.PublicKey.focus();
return true;
}
else {
return false;
}
}
//-->
</SCRIPT>

</HEAD>
<BODY>
<H1> 5 Year PKI IPSEC Server (Firewall) Certificate</H1>
<p>
<H2>Choose one of the following:</H2>
<p>
<ul>
<h3><li>Request a New Certificate</h3>
# This ACTION forces userid/pw authentication and runs the task under
# the client's ID
#<FORM NAME=serverform METHOD=POST ACTION=
#                "/PKIServ/ssl-cgi-bin/auth/careq.rexx" onSubmit=

# This ACTION forces userid/pw authentication but runs the task under
# the surrogate ID
#<FORM NAME=serverform METHOD=POST ACTION=
#              "/PKIServ/ssl-cgi-bin/surrogateauth/careq.rexx" onSubmit=

# This ACTION is for non z/OS clients. The task runs under the
# surrogate ID
```

```
 <FORM NAME=serverform METHOD=POST ACTION=
               "/PKIServ/ssl-cgi-bin/careq.rexx" onSubmit=
     "if(MissingRequiredFAlert()) return false; else return true;">

<INPUT NAME="Template" TYPE="hidden" VALUE="[tmplname]">
<p> Enter values for the following field(s)
 %%CommonName (Optional)%%
 %%OrgUnit (Optional)%%
 %%OrgUnit2 (Optional)%%
 %%Org (Optional)%%
 %%Locality (Optional)%%
 %%StateProv (Optional)%%
 %%Country (Optional)%%
 %%AltEmail (Optional)%%
 %%AltDomain (Optional)%%
 %%AltURI (Optional)%%
 %%AltIPAddr (Optional)%%
 %%Requestor (Optional)%%
 %%PassPhrase%%
 %%PublicKey%%
<p>
<INPUT TYPE="submit" VALUE="Submit certificate request"
ONCLICK="if(MissingRequiredFAlert()) return false; else return true;">
<INPUT TYPE="reset" VALUE="Clear">
</FORM>
<p>
<H3><li>Pick Up a Previously Issued Certificate</H3>

<FORM METHOD=GET ACTION="/PKIServ/ssl-cgi/caretrieve.rexx">
<INPUT NAME="Template" TYPE="hidden" VALUE="[tmplname]">
<INPUT TYPE="submit" VALUE="Retrieve your certificate">
</FORM>
</ul>
<p>%%-pagefooter%%
</BODY>
</HTML>
</CONTENT>
<CONSTANT>
 %%KeyUsage=handshake%%
 %%KeyUsage=dataencrypt%%
 %%NotBefore=0%%
 %%NotAfter=1825%%
 %%SignWith=PKI:%%
</CONSTANT>
<ADMINAPPROVE>
 %%CommonName (Optional)%%
 %%OrgUnit (Optional)%%
 %%OrgUnit (Optional)%%
 %%Org (Optional)%%
 %%Locality (Optional)%%
 %%StateProv (Optional)%%
 %%Country (Optional)%%
 %%AltEmail (Optional)%%
 %%AltDomain (Optional)%%
 %%AltURI (Optional)%%
 %%AltIPAddr (Optional)%%
 %%NotBefore (optional)%%
 %%NotAfter (Optional)%%
 %%KeyUsage (Optional)%%
</ADMINAPPROVE>
<SUCCESSCONTENT>
 %%-requestok%%
</SUCCESSCONTENT>
<FAILURECONTENT>
 %%-requestbad%%
</FAILURECONTENT>
```

The pkiserv.tmpl certificate templates file

```
<RETRIEVECONTENT>
<HTML><HEAD>
<TITLE> Web Based PKIX Certificate Generation Application Pg 3</TITLE>
%%-copyright%%
<SCRIPT LANGUAGE="JavaScript">
<!--
function MissingTransIdAlert(){
var STRING_MissingTransIdPrompt=
             "Enter the transaction ID assigned to the certificate.";
if(document.retrieveform.TransactionId.value==""){
 alert(STRING_MissingTransIdPrompt);
 document.retrieveform.TransactionId.focus();
 return true;
}
else {
 return false;
 }
}
//-->
</SCRIPT>
</HEAD>

<BODY>
<H1> Retrieve Your [tmplname]</H1>
<H3>Please bookmark this page</h3>
<p>Since your certificate may not have been issued yet, we recommend
that you create a bookmark to this location so that when you return to
this bookmark, the browser will display your transaction ID.
This is the easiest way to check your status.

# This ACTION forces userid/pw authentication and runs the task
# under the client's ID
#<FORM NAME=retrieveform METHOD=POST ACTION=
#      "/PKIServ/ssl-cgi-bin/auth/cagetcert.rexx" onSubmit=
#
# This ACTION forces userid/pw authentication but runs the task
# under the surrogate ID
#<FORM NAME=retrieveform METHOD=POST ACTION=
#      "/PKIServ/ssl-cgi-bin/surrogateauth/cagetcert.rexx" onSubmit=
#
# This ACTION is for non z/OS clients. The task runs under surrogate ID
 <FORM NAME=retrieveform METHOD=POST ACTION=
       "/PKIServ/ssl-cgi-bin/cagetcert.rexx" onSubmit=
      "if(MissingTransIdAlert()) return false; else return true;">
<INPUT NAME="Template" TYPE="hidden" VALUE="[tmplname]">
<p> Enter values for the following field(s)
 %%TransactionId%%
 %%ChallengePassPhrase (optional)%%
<p>
<INPUT TYPE="submit" VALUE="Continue">
</FORM>
<p>%%-pagefooter%%
</BODY>
</HTML>
</RETRIEVECONTENT>
<RETURNCERT>
%%-returnpkcs10cert%%
</RETURNCERT>
</TEMPLATE>

#
# ====================================================================
#
# Template Name - 5 Year PKI Intermediate CA Certificate
#
# Function - Creates a 5 year CA certificate. If approved, the
#            certificate becomes valid after it's requested.
```

```
#            (You may delay the valid date by specifying a non zero
#             number for the value of 'NotBefore',
#             e.g. NotBefore=5. That means if the request is approved,
#             the certificate will become valid 5 days after it's
#             requested.)
#
#            These certificates will be stored in LDAP if The O= and
#            OU= suffixes have already been created
#
#
# Other than the user input fields, all other information is hard coded.
#
# User input fields:
#  CommonName - optional
#  OrgUnit - optional
#  Org - optional
#  Locality  - optional
#  StateProv  - optional
#  Country - optional
#  AltEmail  - optional
#  AltDomain  - optional
#  AltURI  - optional
#  AltIPAddr  - optional
#  PassPhrase - required
#  PublicKey - required (This is the PKCS#10 request)
#
#  RACF userid/password authentication : require
#  Administrator approval              : not require
#
#
# ======================================================================
#
<TEMPLATE NAME=5 Year PKI Intermediate CA Certificate>
<TEMPLATE NAME=PKI Server Certificate>
<NICKNAME=5YSCA>
<CONTENT>
<HTML><HEAD>
<TITLE> Web Based PKIX Certificate Generation Application Pg 2</TITLE>
%%-copyright%%
<SCRIPT LANGUAGE="JavaScript">
<!--
function MissingRequiredFAlert(){
var STRING_MissingRequiredFPrompt=
           "Enter the field value that's not optional."
var STRING_MissingConfirmPwdPrompt=
                  "Reenter password."
var STRING_MissingPwdPrompt=
                  "Need to enter password before confirm it."
var STRING_UnmatchPwdPrompt=
                  "The passwords do not match. Enter again."

if (document.serverform.PassPhrase.value=="")
{
  alert(STRING_MissingRequiredFPrompt);
  document.serverform.PassPhrase.focus();
  return true;
}
else if(document.serverform.ConfirmPassPhrase.value=="") {
alert(STRING_MissingConfirmPwdPrompt);
document.serverform.ConfirmPassPhrase.focus();
return true;
}
else if(document.serverform.PassPhrase.value!=
        document.serverform.ConfirmPassPhrase.value){
alert(STRING_UnmatchPwdPrompt);
document.serverform.ConfirmPassPhrase.focus();
return true;
```

```
}
else if(document.serverform.PublicKey.value=="") {
alert(STRING_MissingRequiredFPrompt);
document.serverform.PublicKey.focus();
return true;
}
else {
return false;
}
}
//-->
</SCRIPT>

</HEAD>
<BODY>
<H1> 5 Year PKI Intermediate CA Certificate</H1>
<p>
<H2>Choose one of the following:</H2>
<p>
<ul>
<h3><li>Request a New Certificate</h3>
# This ACTION forces userid/pw authentication and runs the task under
# the client's ID
 <FORM NAME=serverform METHOD=POST ACTION=
                 "/PKIServ/ssl-cgi-bin/auth/careq.rexx" onSubmit=

# This ACTION forces userid/pw authentication but runs the task under
# the surrogate ID
#<FORM NAME=serverform METHOD=POST ACTION=
#              "/PKIServ/ssl-cgi-bin/surrogateauth/careq.rexx" onSubmit=

# This ACTION is for non z/OS clients. The task runs under the
# surrogate ID
#<FORM NAME=serverform METHOD=POST ACTION=
#              "/PKIServ/ssl-cgi-bin/careq.rexx" onSubmit=
     "if(MissingRequiredFAlert()) return false; else return true;">

<INPUT NAME="Template" TYPE="hidden" VALUE="[tmplname]">
<p> Enter values for the following field(s)
 %%CommonName (Optional)%%
 %%OrgUnit (Optional)%%
 %%OrgUnit2 (Optional)%%
 %%Org (Optional)%%
 %%Locality (Optional)%%
 %%StateProv (Optional)%%
 %%Country (Optional)%%
 %%AltEmail (Optional)%%
 %%AltDomain (Optional)%%
 %%AltURI (Optional)%%
 %%AltIPAddr (Optional)%%
 %%Requestor (optional)%%
 %%PassPhrase%%
 %%PublicKey%%
<p>
<INPUT TYPE="submit" VALUE="Submit certificate request"
ONCLICK="if(MissingRequiredFAlert()) return false; else return true;">
<INPUT TYPE="reset" VALUE="Clear">
</FORM>
<p>
<H3><li>Pick Up a Previously Issued Certificate</H3>

<FORM METHOD=GET ACTION="/PKIServ/ssl-cgi/caretrieve.rexx">
<INPUT NAME="Template" TYPE="hidden" VALUE="[tmplname]">
<INPUT TYPE="submit" VALUE="Retrieve your certificate">
</FORM>
</ul>
<p>%%-pagefooter%%
```

```
</BODY>
</HTML>
</CONTENT>
<APPL>
 %%UserId%%
</APPL>
<CONSTANT>
 %%NotBefore=0%%
 %%NotAfter=1825%%
 %%KeyUsage=certsign%%
 %%SignWith=PKI:%%
</CONSTANT>
<SUCCESSCONTENT>
 %%-requestok%%
</SUCCESSCONTENT>
<FAILURECONTENT>
 %%-requestbad%%
</FAILURECONTENT>

<RETRIEVECONTENT>
<HTML><HEAD>
<TITLE> Web Based PKIX Certificate Generation Application Pg 3</TITLE>
%%-copyright%%
<SCRIPT LANGUAGE="JavaScript">
<!--
function MissingTransIdAlert(){
var STRING_MissingTransIdPrompt=
             "Enter the transaction ID assigned to the certificate.";
if(document.retrieveform.TransactionId.value==""){
 alert(STRING_MissingTransIdPrompt);
 document.retrieveform.TransactionId.focus();
 return true;
}
else {
 return false;
 }
}
//-->
</SCRIPT>
</HEAD>

<BODY>
<H1> Retrieve Your [tmplname]</H1>
<H3>Please bookmark this page</h3>
<p>Since your certificate may not have been issued yet, we recommend
that you create a bookmark to this location so that when you return to
this bookmark, the browser will display your transaction ID.
This is the easiest way to check your status.

# This ACTION forces userid/pw authentication and runs the task
# under the client's ID
 <FORM NAME=retrieveform METHOD=POST ACTION=
       "/PKIServ/ssl-cgi-bin/auth/cagetcert.rexx" onSubmit=
#
# This ACTION forces userid/pw authentication but runs the task
# under the surrogate ID
#<FORM NAME=retrieveform METHOD=POST ACTION=
#      "/PKIServ/ssl-cgi-bin/surrogateauth/cagetcert.rexx" onSubmit=
#
# This ACTION is for non z/OS clients. The task runs under surrogate ID
#<FORM NAME=retrieveform METHOD=POST ACTION=
#      "/PKIServ/ssl-cgi-bin/cagetcert.rexx" onSubmit=
      "if(MissingTransIdAlert()) return false; else return true;">
<INPUT NAME="Template" TYPE="hidden" VALUE="[tmplname]">
<p> Enter values for the following field(s)
 %%TransactionId%%
 %%ChallengePassPhrase (optional)%%
```

```
<p>
<INPUT TYPE="submit" VALUE="Continue">
</FORM>
<p>%%-pagefooter%%
</BODY>
</HTML>
</RETRIEVECONTENT>
<RETURNCERT>
%%-returnpkcs10cert%%
</RETURNCERT>
</TEMPLATE>

#
# ============================================================================
#
# Sample INSERTS
#
# ============================================================================
#
<INSERT NAME=-AdditionalHeadIE>
<OBJECT
  classid="clsid:43F8F289-7A20-11D0-8F06-00C04FC295E1"
  CODEBASE="xenroll.cab"
  id="certmgr"
>
</OBJECT>
</INSERT>

<INSERT NAME=-requestok>
<HTML><HEAD>
<TITLE> Web Based Certificate Generation Success</TITLE>
</HEAD>
<BODY>
<H1> Request submitted successfully</H1>
[errorinfo]
<p> Here's your transaction ID. You will need it to retrieve your
certificate.  Press 'Continue' to retrieve the certificate.
<p> <TABLE BORDER><TR><TD>[transactionid]</TD></TR></TABLE>
<FORM METHOD=GET ACTION="/PKIServ/ssl-cgi/caretrieve.rexx">
<INPUT NAME="Template" TYPE="hidden" VALUE="[tmplname]">
<INPUT NAME="TransactionId" TYPE="hidden" VALUE="[transactionid]">
<INPUT TYPE="submit" VALUE="Continue">
</FORM>
<p>%%-pagefooter%%
</BODY>
</HTML>
</INSERT>

<INSERT NAME=-requestbad>
<HTML><HEAD>
<TITLE> Web Based Certificate Generation Failure</TITLE>
</HEAD>
<BODY>
<H1> Request was not successful</H1>
<p> Please correct the problem or report the error to your Web admin
person<br>
<PRE>
[errorinfo]
</PRE>
<p>%%-pagefooter%%
</BODY>
</HTML>
</INSERT>

<INSERT NAME=-renewrevokeok>
<HTML><HEAD>
<TITLE> Web Based Certificate Renew/Revoke Success</TITLE>
```

```
</HEAD>
<BODY>
<H1> Request submitted successfully</H1>
<FORM METHOD=GET ACTION="/PKIServ/public-cgi/camain.rexx">
<INPUT TYPE="submit" VALUE="Home Page">
</FORM>
<p>%%-pagefooter%%
</BODY>
</HTML>
</INSERT>

<INSERT NAME=-renewrevokebad>
<HTML><HEAD>
<TITLE> Web Based Certificate Renew/Revoke Failure</TITLE>
</HEAD>
<BODY>
<H1> Request was not successful</H1>
<p> Please correct the problem or report the error to your Web admin
person<br>
<PRE>
[errorinfo]
</PRE>
<FORM METHOD=GET ACTION="/PKIServ/public-cgi/camain.rexx">
<INPUT TYPE="submit" VALUE="Home Page">
</FORM>
<p>%%-pagefooter%%
</BODY>
</HTML>
</INSERT>

<INSERT NAME=-returnpkcs10cert>
<HTML><HEAD>
<TITLE> Web Based SAF Certificate Generation Application Pg 4</TITLE>
</HEAD>
<BODY>
<H1> Here's your Certificate. Cut and paste it to a file</H1>
<TABLE BORDER><TR><TD>
<PRE>
[base64cert]
</PRE>
</TD></TR></TABLE>
<p>%%-pagefooter%%
</BODY>
</HTML>
</INSERT>

<INSERT NAME=returnbrowsercertNS>
[base64cert]
</INSERT>

<INSERT NAME=returnbrowsercertIE>
<HTML>
<HEAD>
<TITLE>MSIE Certificate Install</TITLE>
<OBJECT
  classid="clsid:43F8F289-7A20-11D0-8F06-00C04FC295E1"
  CODEBASE="xenroll.cab"
  id="certmgr"
>
</OBJECT>
</HEAD>
<BODY>
<SCRIPT LANGUAGE="VBScript">
<!--
  Sub INSTALL_OnClick
    Dim pkcs7data, errmsg, rc
    On Error Resume Next
```

```
    certmgr.DeleteRequestCert = false
    err.clear
    certmgr.WriteCertToCSP = true
    pkcs7data = "[iecert]"
    certmgr.acceptPKCS7(pkcs7data)
    if err.number <> 0 then
certmgr.WriteCertToCSP = false
        err.clear
        certmgr.acceptPKCS7(pkcs7data)
    end if
    if err.number <> 0 then
errmsg = "Your new certificate failed to install. " & _
   "Please ensure that you are using the same browser " & _
   "that you used when making the certificate request. "
rc = MsgBox (errmsg, 48, "Certificate Installation")
    else
errmsg = "Your new certificate installed successfully."
rc = MsgBox (errmsg, 64, "Certificate Installation")
    end if
  End Sub
  // -->
</SCRIPT>
<h1>Internet Explorer certificate install</h1>
<p> Click &quot;Install Certificate&quot; to store your new
certificate into your browser
<TABLE>
<TR> <br>
<TD><INPUT TYPE="BUTTON" VALUE="Install Certificate" NAME="INSTALL" >
<FORM METHOD=GET ACTION="/PKIServ/public-cgi/camain.rexx">
<INPUT NAME="Template" TYPE="hidden" VALUE="[tmplname]">
<INPUT TYPE="submit" VALUE="Home page">
</FORM>
</TD>
</TR>
</TABLE>
</BODY>
</HTML>
</INSERT>
#
# ====================================================================
#
# X.509 fields (INSERTs) valid for certificate requests
#
# ====================================================================
#
<INSERT NAME=KeyUsage>
<p> Indicate the intended purpose for the certificate [optfield] <BR>
<SELECT NAME="KeyUsage" MULTIPLE>
 <OPTION VALUE="handshake">Protocol handshaking (e.g., SSL)
 <OPTION VALUE="dataencrypt">Data encryption
 <OPTION VALUE="certsign">Certificate signing
 <OPTION VALUE="docsign">Document signing (nonrepudiation)
</SELECT>
</INSERT>

<INSERT NAME=NotBefore>
<p> Number of days after today before the certificate becomes current
 [optfield] <BR>
<SELECT NAME="NotBefore">
 <OPTION> 0
 <OPTION> 30
</SELECT>
</INSERT>

<INSERT NAME=NotAfter>
<p> Length of time that the certificate is current [optfield] <BR>
<SELECT NAME="NotAfter">
```

```
 <OPTION value="365">1 Year
 <OPTION value="730">2 Years
</SELECT>
</INSERT>

<INSERT NAME=Country>
<p> Country [optfield] <BR>
<INPUT NAME="Country" TYPE="text" SIZE=2 maxlength="2">
</INSERT>

<INSERT NAME=Org>
<p> Organization [optfield] <BR>
<INPUT NAME="Org" TYPE="text" SIZE=64  maxlength="64">
</INSERT>

<INSERT NAME=OrgUnit>
<p> Organizational Unit [optfield] <BR>
<INPUT NAME="OrgUnit" TYPE="text" SIZE=64  maxlength="64">
</INSERT>

<INSERT NAME=OrgUnit2>
<p> Organizational Unit [optfield] <BR>
<INPUT NAME="OrgUnit2" TYPE="text" SIZE=64  maxlength="64">
</INSERT>

<INSERT NAME=Locality>
<p> Locality [optfield] <BR>
<INPUT NAME="Locality" TYPE="text" SIZE=64  maxlength="64">
</INSERT>

<INSERT NAME=StateProv>
<p> State or Province [optfield] <BR>
<INPUT NAME="StateProv" TYPE="text" SIZE=64  maxlength="64">
</INSERT>

<INSERT NAME=CommonName>
<p> Common Name [optfield] <BR>
<INPUT NAME="CommonName" TYPE="text" SIZE=64  maxlength="64">
</INSERT>

<INSERT NAME=Title>
<p> Title [optfield] <BR>
<INPUT NAME="Title" TYPE="text" SIZE=64  maxlength="64">
</INSERT>

<INSERT NAME=AltIPAddr>
<p> IP address in dotted decimal form [optfield] <BR>
<INPUT NAME="AltIPAddr" TYPE="text" SIZE=15  maxlength="15">
</INSERT>

<INSERT NAME=AltEmail>
<p> Email address [optfield] <BR>
<INPUT NAME="AltEmail" TYPE="text" SIZE=100  maxlength="100">
</INSERT>

<INSERT NAME=AltURI>
<p> Uniform Resource Identifier [optfield] <BR>
<INPUT NAME="AltURI" TYPE="text" SIZE=100  maxlength="255">
</INSERT>

<INSERT NAME=AltDomain>
<p> Domain name [optfield] <BR>
<INPUT NAME="AltDomain" TYPE="text" SIZE=100  maxlength="100">
</INSERT>

<INSERT NAME=SignWith>
<p> Component:/key-Label used to sign this certificate [optfield] <BR>
```

```
<p> e.g., "SAF:CERTAUTH/Local CA Cert" sign by CERTAUTH certificate
 "Local CA Cert"
<INPUT NAME="SignWith" TYPE="text" SIZE=45  maxlength="45">
</INSERT>

<INSERT NAME=PublicKey>
<p> Base64 encoded PKCS#10 certificate request [optfield] <BR>
<TEXTAREA NAME="PublicKey"
  COLS="70"
  ROWS="12"
  WRAP="OFF">
</TEXTAREA>
</INSERT>


<INSERT NAME=PublicKeyNS>
<p> Select a key size
<KEYGEN NAME="PublicKey">
<p>
<INPUT TYPE="Submit" VALUE="Submit certificate request">
</INSERT>

<INSERT NAME=PublicKey2NS>
<p> Select a key size
<KEYGEN NAME="PublicKey">
<p>
<INPUT TYPE="Submit" VALUE="Submit certificate request">
</INSERT>

<INSERT NAME=PublicKeyIE>
<SCRIPT LANGUAGE="VBScript">
<!--
Sub SendReq

On Error Resume Next
Dim pkcs10data,DN,i,Message

        DN= ""
CommonName= "Unspecified Distinguished Name"
DN= "CN=" + CommonName + ";"
certmgr.KeySpec = 1
KeyUsage = "1.3.6.1.5.5.7.3.2"
i = document.all.CSP.options.selectedIndex
certmgr.providerName = document.all.CSP.options(i).text
certmgr.providerType = document.all.CSP.options(i).value
If document.CertReq.KeyProt.value = 1 Then
certmgr.GenKeyFlags = 3
Else
certmgr.GenKeyFlags = 1
End If
pkcs10data = ""
pkcs10data = certmgr.CreatePKCS10(DN, KeyUsage)
document.CertReq.PublicKey.value = pkcs10data

If Len(pkcs10data) > 0 Then
document.CertReq.submit()
return True
Else
return MsgBox ("PKCS10 Creation Failed",48,"Certificate request")
    End If

End Sub
// -->
</SCRIPT>

<p> Select the following key information
<p> Cryptographic Service Provider
```

```
<select name="CSP">
<script language="VBScript">
 On Error Resume Next
   Dim i, csp, sv


 certmgr.providerType = 1
 i = 0
 csp = ""
 csp = certmgr.enumProviders(i,0)
 sv = "SELECTED"
 While Len(csp) <> 0
  document.write("<OPTION VALUE=1 " & sv & ">" & csp & "</OPTION>")
  i = i + 1
  csp = ""
  csp = certmgr.enumProviders(i,0)
  selvalue = ""
 Wend
</script>
</select>

<p> Enable strong private key protection?
<select name="KeyProt">
 <option value="1">Yes</option>
 <option value="0" selected>No</option>
</select>
<input type="hidden" name="PublicKey" value="">
<p>
<INPUT TYPE="Button" VALUE="Submit certificate request" ,
  ONCLICK="if (MissingRequiredFAlert()) return false; SendReq()">
</INSERT>

<INSERT NAME=PublicKey2IE>
<SCRIPT LANGUAGE="VBScript">
<!--
Sub SendReq

On Error Resume Next
Dim pkcs10data,DN,i,Message

        DN= ""
CommonName= "Unspecified Distinguished Name"
DN= "CN=" + CommonName + ";"
certmgr.KeySpec = 1
KeyUsage = "1.3.6.1.5.5.7.3.2"
i = document.all.CSP.options.selectedIndex
certmgr.providerName = document.all.CSP.options(i).text
certmgr.providerType = document.all.CSP.options(i).value
If document.CertReq.KeyProt.value = 1 Then
certmgr.GenKeyFlags = 3
Else
certmgr.GenKeyFlags = 1
End If
pkcs10data = ""
pkcs10data = certmgr.CreatePKCS10(DN, KeyUsage)
document.CertReq.PublicKey.value = pkcs10data

If Len(pkcs10data) > 0 Then
document.CertReq.submit()
return True
Else
return MsgBox ("PKCS10 Creation Failed",48,"Certificate request")
    End If

End Sub
// -->
</SCRIPT>
```

```
<p> Select the following key information
<p> Cryptographic Service Provider
<select name="CSP">
<script language="VBScript">
 On Error Resume Next
   Dim i, csp, sv


 certmgr.providerType = 1
 i = 0
 csp = ""
 csp = certmgr.enumProviders(i,0)
 sv = "SELECTED"
 While Len(csp) <> 0
  document.write("<OPTION VALUE=1 " & sv & ">" & csp & "</OPTION>")
  i = i + 1
  csp = ""
  csp = certmgr.enumProviders(i,0)
  selvalue = ""
 Wend
</script>
</select>

<p> Enable strong private key protection?
<select name="KeyProt">
 <option value="1">Yes</option>
 <option value="0" selected>No</option>
</select>
<input type="hidden" name="PublicKey" value="">
<p>
<INPUT TYPE="Button" VALUE="Submit certificate request" ,
  ONCLICK="if (ValidateEntry()) return false; SendReq()">
</INSERT>
#
# ========================================================================
#
# non-X.509 certificate request fields (INSERTs)
#
# ========================================================================
#
<INSERT NAME=UserId>
<p> Owning SAF User ID [optfield] <BR>
<INPUT NAME="UserId" TYPE="text" SIZE=8 maxlength="8">
</INSERT>

<INSERT NAME=Label>
<p> Label assigned to certificate being requested [optfield] <BR>
<INPUT NAME="Label" TYPE="text" SIZE=32 maxlength="32">
</INSERT>

<INSERT NAME=Requestor>
<p> Your name for tracking this request [optfield] <BR>
<INPUT NAME="Requestor" TYPE="text" SIZE=32 maxlength="32">
</INSERT>

<INSERT NAME=PassPhrase>
<p> Pass phrase for securing this request. You will need to supply
this value when retrieving your certificate [optfield] <BR>
<INPUT NAME="PassPhrase" TYPE="password" SIZE=32 maxlength="32"> <BR>
<p> Reenter your pass phrase to confirm <BR>
<INPUT NAME="ConfirmPassPhrase" TYPE="password" SIZE=32
 maxlength="32">
</INSERT>

<INSERT NAME=ChallengePassPhrase>
<p> If you specified a pass phrase when submitting the certificate
```

```
request, type it here, exactly as you typed it on the
request form <BR>
<INPUT NAME="ChallengePassPhrase" TYPE="password" SIZE=32
maxlength="32">
</INSERT>

<INSERT NAME=HostIdMap>
<p> HostIdMapping Extension value in subject-id@host-name form
    [optfield] <BR>
<INPUT NAME="HostIdMap" TYPE="text" SIZE=100 maxlength="100">
</INSERT>

<INSERT NAME=TransactionId>
<p> Enter the assigned transaction ID [optfield] <BR>
<INPUT NAME="TransactionId" TYPE="text" SIZE=56 maxlength="56"
VALUE="[transactionid]">
</INSERT>

#####################################################################
#                                                                   #
#                   Additional section                              #
#                                                                   #
#####################################################################

<INSERT NAME=-copyright>
<!--
/*******************************************************************/
/*                                                                 */
/* Licensed Materials - Property of IBM                            */
/* 5694-A01                                                        */
/* (C) Copyright IBM Corp. 2001                                    */
/*                                                                 */
/*******************************************************************/
-->
<META HTTP-EQUIV="Content Type" content="text/html; charset=ISO-8859-1">
</INSERT>

<INSERT NAME=-pagefooter>
<A HREF="mailto:webmaster@your-company">
email: webmaster@your-company.com</A>
</INSERT>
```

The pkiserv.tmpl certificate templates file

Chapter 23. Environment variables

This chapter describes the environment variables that PKI Services uses and their possible values. It also includes a code sample of the environment variables file, pkiserv.envars (see “The pkiserv.envars environment variables file” on page 267). For information about PKISERVD proc, which specifies the pathname of the environment variables file, see “PKISERVD sample procedure to start PKI Services daemon” on page 292.

Environment variables in the environment variables file

The environment variables contained in pkiserv.envars and their values are:

_PKISERV_MSG_LOGGING

Values include:

STDOUT_LOGGING Indicates writing all messages (verbose, debug, informational, warning, error, and severe) to STDOUT and *additionally* writing the error and severe messages to STDERR. This is the default if the environment variable is not set.

STDERR_LOGGING Indicates writing verbose, debug, informational, and warning messages to STDOUT and writing error and severe messages to STDERR.

_PKISERV_MSG_LEVEL

Specifies the subcomponent and message level to log. Messages for a particular subcomponent are logged only if the message level is greater than or equal to the specified level for that subcomponent. You can use an asterisk (*) to indicate all subcomponents. The subcomponent list consists of a subcomponent name and a message level separated by a period (.).

For example, the following sets the message level for all subcomponents to log warning messages or higher. (This is the default setting.)

Example:

```
_PKISERV_MSG_LEVEL=*.W
```

You can specify multiple subcomponents by separating the entries with commas (,). For example, the following indicates that all subcomponents are set to message level W (Warning) and that the PKID subcomponent is set to message level D (Debug).

Example:

```
_PKISERV_MSG_LEVEL=*.W,PKID.D
```

The subcomponents are:

Table 58. Subcomponents for message level

Subcomponent	Meaning
*	This is the wildcard character, which represents all subcomponents.
CORE	The core functions of PKI Services that are not specific to the other subcomponents.
DB	Activity related to the request or issued certificate VSAM data stores.
LDAP	LDAP posting operations.
PKID	The PKI Services daemon address setup and infrastructure.
POLICY	Certificate creation and revocation policy processing.
SAF	SAF key ring, OCEP, and `R_datalib` calls.

The message levels are:

Table 59. Message levels

Debug level (hierarchically listed)	Meaning
S	This indicates logging only Severe messages.
E	This indicates logging Severe and Error messages.
W	This indicates logging Severe, Error, and Warning messages. This is the default message level for all subcomponents if you do not set the environment variable.
I	This indicates logging Severe, Error, Warning, and Informational messages.
D	This indicates logging Severe, Error, Warning, Informational and Diagnostic.
V	This indicates logging ALL messages, including Verbose Diagnostic messages. This is very verbose. **Recommendation:** Do not use this level unless IBM support personnel request you to do so.

_PKISERV_CONFIG_PATH

Specifies the pathname for the directory containing the configuration file, pkiserv.conf, and the certificate template file, pkiserv.tmpl. The default value (if you do not set the environment variable) is /etc/pkiserv.

Recommendation: Copy both of these files from the install directory, /usr/lpp/pkiserv/samples, before making any changes.

Note: Because the PKISERV CGIs run in a z/OS HTTP Server address space, if the pkiserv.tmpl is not in its default location of /etc/pkiserv/pkiserv.tmpl, you need to add the _PKISERV_CONFIG_PATH variable to the z/OS HTTP Server's environment variable file. The HTTP servers environment variables file is usually in /etc/httpd.envvars. PKI Services uses two instances of the z/OS HTTP Server.

Therefore, if the two servers are using different environment variables files, you need to update both files.

_PKISERV_EXIT

Specifies the full pathname for the installation-provided PKI exit program that the PKI Services Web page interface calls. (This exit is a UNIX-executable program or shell script.) If you do not define this variable or if it contains a null value, the PKI exit processing is disabled.

Note: The PKI Services CGI scripts run in a z/OS HTTP Server address space, so you must specify the _PKISERV_EXIT environment variable in the z/OS HTTP Server's environment variables file. The z/OS HTTP Server environment variables file is usually /etc/httpd.envvars. PKI Services uses two instances of the z/OS HTTP Server. Therefore, if the two servers are using different environment variables files, you need to update both files.

The pkiserv.envars environment variables file

The following code sample is for the pkiserv.envars environment variables file. (For information about updating the environment variables file, see "Optionally updating PKI Services environment variables" on page 40.) The code sample that follows might not be identical to the code shipped with the product. To see the most current code, look at the pkiserv.envars file in the source directory /usr/lpp/pkiserv/samples/.

```
#-------------------------------------------------------------------#
#                                                                   #
# PKI Services sample environment variable file                     #
#                                                                   #
# Licensed Materials - Property of IBM                              #
# 5694-A01                                                          #
# (C) Copyright IBM Corp.  2001                                     #
# Status = HKY7706                                                  #
#                                                                   #
#-------------------------------------------------------------------#
#
# Language and Path configurations
#
LANG=En_US.IBM-1047
LIBPATH=/usr/lpp/pkiserv/lib:/usr/lib
NLSPATH=/usr/lib/nls/msg/%L/%N:/usr/lpp/pkiserv/lib/nls/msg/%L/%N
#
# Configuration File location and Message configuration Options
#
_PKISERV_CONFIG_PATH=/etc/pkiserv
_PKISERV_MSG_LOGGING=stdout_logging
_PKISERV_MSG_LEVEL=*.w
#
# Location of the OCSF Registry (/var/ocsf is the default location)
#
OCSFREGDIR=/var/ocsf
```

Chapter 24. The IKYSETUP REXX exec

IKYSETUP is a REXX exec that issues RACF commands to perform RACF administration. This chapter describes the actions IKYSETUP performs and provides a code sample of IKYSETUP.

Actions IKYSETUP performs by issuing RACF commands

In broad terms, the actions that IKYSETUP performs are as follows:

- Sets up the PKI Services daemon user ID
- Sets up the access control to protect PKI Services
  - Protects end-user functions
  - Protects administrative functions
- Creates the CA certificate, private key, and key ring
- Creates the z/OS HTTP Server certificate, private key, and key ring
- Enables surrogate operation for the z/OS HTTP Server
- Enables the PKI Services daemon to call OCSF functions

Setting up the PKI Services daemon user ID

Create the daemon user ID (by default, PKISRVD) using the RACF ADDUSER TSO command. Give it an OMVS segment because it needs access to UNIX System Services. This user ID also needs update access to the VSAM data sets identified in the [ObjectStore] section of the pkiserv.conf file. If necessary, use the RACF ADDSD and PERMIT TSO commands to give this user ID UPDATE access to the VSAM data sets.

Recommendation: Define the daemon user ID with the NOPASSWORD attribute.

To associate this user ID to the PKI Services started procedure, use the following RACF TSO commands:

```
RDEFINE STARTED PKISERVD.* STDATA(USER(PKISRVD))
SETROPTS CLASSACT(STARTED) RACLIST(STARTED)
SETROPTS RACLIST(STARTED) REFRESH
```

Setting up access control to protect PKI Services

This task can be divided into two steps:

1. Protecting end-user functions
2. Protecting administrative functions.

Protecting end-user functions

You must first determine who your end-users are and how they will be using their certificates. In general there are two categories of end-users:

- Internal clients, such as employees who have SAF user IDs on the host system and who may be using their certificates to access resources on the host
- External clients, who have no access to the host system.

When PKI Services is called, the unit of work has some identity (user ID) associated with it. For external customers, a surrogate user ID is necessary.

Recommendation: Although under certain circumstances it may be beneficial for internal clients to access PKI Services under their own identities, your implementation will be simpler if you use surrogate user IDs for internal clients as well.

Use the RACF ADDUSER TSO command to create the surrogate user ID (PKISERV). Give it an OMVS segment because it needs access to z/OS UNIX.

Recommendation: Define the surrogate user ID with the PROTECTED and RESTRICTED attributes.

The R_PKIServ SAF callable service is protected by FACILITY class resources of the form IRR.RPKISERV.*function*, where function is one of the following:

- GENCERT
- EXPORT
- REQCERT
- VERIFY
- REVOKE
- GENRENEW
- REQRENEW.

Create these resources and give the PKISERV user ID either READ or CONTROL access to them. CONTROL bypasses subsequent resource checks.

Additional FACILITY class resources of the form IRR.DIGTCERT.*function* protect the actual certificate generation and retrieval functions. If subsequent resource checks are not being bypassed, define these resources and their access.

There are two ways to handle certificate approval:

- An administrator can review certificate requests
- Requests can be auto-approved without administrator action (this should probably be reserved for internal clients only).

If you plan to have an administrator approve certificate requests before issuing certificates, PKISERV needs the following access:

Table 60. Access required if you plan to use an administrator

Resource	Access
IRR.DIGTCERT.REQCERT	READ
IRR.DIGTCERT.VERIFY	READ
IRR.DIGTCERT.REVOKE	READ
IRR.DIGTCERT.REQRENEW	READ
IRR.DIGTCERT.EXPORT	(If your end-users will always provide a passphrase) READ (Otherwise) UPDATE

If your clients request certificates that are auto-approved without action by an administrator, PKISERV needs the following access:

Table 61. Access required if you plan to use auto-approval

Resource	Access
IRR.DIGTCERT.GENCERT	CONTROL

Table 61. Access required if you plan to use auto-approval (continued)

Resource	Access
IRR.DIGTCERT.ADD	UPDATE
IRR.DIGTCERT.VERIFY	READ
IRR.DIGTCERT.REVOKE	READ
IRR.DIGTCERT.GENRENEW	READ
IRR.DIGTCERT.EXPORT	(If your end-users will always provide a passphrase), READ (Otherwise) UPDATE

Finally, because the Web server will be switching identities to PKISERV, you must give it surrogate permission. This is done by creating another resource in the SURROGAT class (BPX.SRV.PKISERV) and giving the Web server daemon user ID READ access to it.

Protecting administrative functions

This is much easier to set up than protecting the end-user functions. Your PKI Services administrators must have SAF user IDs on the host system. When PKI Services is called for administrative functions, the unit of work is always tagged with the identity of the authenticated administrator. Each administrator needs the following FACILITY class resource access to:

Table 62. FACILITY class access needed for protecting administrative functions

Resource	Access	(Purpose)
IRR.RPKISERV.PKIADMIN	READ	(For list and query operations)
	UPDATE	(To act on certificate requests and issued certificates)

To grant user ID ADMINID authority to administer PKI Services, use the following RACF TSO commands:

```
RDEFINE FACILITY (IRR.RPKISERV.PKIADMIN) UACC(NONE)
PERMIT IRR.RPKISERV.PKIADMIN CLASS(FACILITY) ACCESS(UPDATE) ID(ADMINID)
SETROPTS RACLIST (FACILITY) REFRESH
```

Creating the CA certificate, private key, and key ring

To create and sign digital certificates for others, you need to define a CA certificate and associated private key. The RACF RACDCERT GENCERT TSO command does this.

Before issuing the command, you need to know what the CA's distinguished name will be and where it will be located (under CERTAUTH or under the PKI Services daemon user ID). Typically, CAs have distinguished names in the following form:

OU=*your-CA's-friendly-name*.O=*your-organization*.C=*your-two-letter-country-abbreviation*

Example:

The RACDCERT GENCERT TSO command to create a 20–year CERTAUTH certificate with a distinguished name of OU=Human Resources Certificate Authority.O=Your Company, Inc.C=US is:

```
RACDCERT CERTAUTH GENCERT SUBJECTSDN(OU('Human Resources Certificate Authority')
   O('Your Company, Inc')    C('US')) WITHLABEL('Local PKI CA')
   NOTBEFORE(DATE(2001/05/07)) NOTAFTER(DATE(2021/05/06))
```

To back up the certificate and private key to a password-protected data set and migrate the private key to ICSF, issue:

```
RACDCERT CERTAUTH EXPORT(LABEL('Local PKI CA')) DSN('PKISRVD.PRIVATE.KEY.P12BIN')
   FORMAT(PKCS12DER)  PASSWORD('your-passphrase')

RACDCERT CERTAUTH ADD('PKISRVD.PRIVATE.KEY.P12BIN')
 PASSWORD('your-passphrase') ICSF
```

Note: The preceding example assumes you want to use ICSF for private key protection and signing. For this to succeed, ICSF must be running and configured for RSA operations. (For additional information, see *z/OS ICSF Administrator's Guide*.) If you do not want to use ICSF, omit the RACDCERT ADD command.

After your CA certificate is created, you must place it in a key ring so that PKI Services can access it. This is also done using the RACF RACDCERT TSO command with sub keywords ADDRING and CONNECT. For example, the RACDCERT TSO commands to create a key ring called CAring for User ID PKISRVD and connect the preceding certificate to it are:

Example:

```
RACDCERT ADDRING(CAring) ID(PKISRVD)
RACDCERT ID(PKISRVD) CONNECT(CERTAUTH LABEL('Local PKI CA') RING(CAring)
   USAGE(PERSONAL) DEFAULT)
```

Note: Make sure your CA certificate is marked TRUSTed in RACF. (Otherwise PKI Services will not be able to use the certificate.) Use the RACDCERT LIST command to check this and the RACDCERT ALTER command to change it if needed.

To use RACF's certificate services, the PKISRVD user ID needs access to the following FACILITY Class resources:

Table 63. Access PKISERVD needs to use RACF's certificate services

Resource	Access
IRR.DIGTCERT.GENCERT	(If the CA certificate was created under CERTAUTH) CONTROL (Otherwise) READ
IRR.DIGTCERT.LISTRING	READ

Configuring the z/OS HTTP Server for SSL mode

The PKISERV application requires the z/OS HTTP Server to operate in three modes. That is why PKI Services requires two z/OS HTTP Servers. The modes are:

- Normal
- SSL without client authentication
- SSL with client authentication.

For SSL, your server needs to obtain a digital certificate. You can:

- Purchase one from an external source
- Create one using RACF

Note: If your server is already operating in SSL mode, you can skip the following section, “Using RACF to obtain a certificate for the Web server”.

Using RACF to obtain a certificate for the Web server

The z/OS HTTP Server supports using either gskkyman key databases (.kdb files) or RACF (SAF) key rings for the server’s certificate store. You are expected to use SAF key rings if setting up their Web server for the first time.

Note: If you have already set up your Web server using gskkyman, you can continue to use it.

Use RACDCERT to generate the server certificate signed by the new Certificate Authority.

Example:

```
RACDCERT GENCERT ID(WEBSRV) SIGNWITH(CERTAUTH LABEL('Local PKI CA'))
 WITHLABEL('SSL Cert') SUBJECTSDN(CN('www.yourcompany.com') O('Your Company Inc')
 L('Millbrook') SP('New York') C('US'))
```

The Web server needs a key ring containing its new certificate and any trusted CA certificate. The RACDCERT command with operands ADDRING and CONNECT also sets this up. For example, the RACDCERT commands to create a key ring called SSLring for user ID WEBSRV and to connect the Web server and CA certificates to it are:

Example:

```
RACDCERT ADDRING(SSLring) ID(websrv)
RACDCERT ID(websrv) CONNECT(CERTAUTH LABEL('Local PKI CA')) RING(SSLring)
     USAGE(PERSONAL) DEFAULT)
RACDCERT ID(websrv) CONNECT(ID(websrv) LABEL('SSL Cert') RING(SSLring)
     USAGE(PERSONAL) DEFAULT)
```

Export the CA certificate to an MVS data set. Then OPUT it to an HFS file so that it can be made available to your clients.

Example:

```
RACDCERT EXPORT(LABEL(''Local PKI CA'))
     CERTAUTH DSN('pkisrvd.private.cacert.derbin') FORMAT(CERTDER)
```

Enabling the z/OS HTTP Server for surrogate operation

Your server must be able to act as a surrogate for clients requesting certificates. To enable this, create:

- Profile BPX.SERVER in the FACILITY class
- Profile BPX.SRV.PKISERV in the SURROGAT class.

Give the z/OS HTTP Server daemon user ID READ access to both of these profiles.

Enabling the PKI Services daemon to call OCSF functions

For access to OCSF, the PKI Services daemon needs READ access to BPX.SERVER. If program control is in effect (for example, z/OS UNIX level security), then the daemon needs READ access to BPX.DAEMON as well.

Code Sample: IKYSETUP

IKYSETUP contains the commands to perform the RACF administrator tasks of adding groups and user IDs, setting up access control, creating CA and SSL certificates, and setting up daemon security. The example that follows might not be identical to the code shipped with the product. To see the exact code, look at SYS1.SAMPLIB member IKYSETUP.

```
/* REXX */
/*********************************************************************/
/*                                                                   */
/* DESCRIPTIVE NAME:  PKI Services RACF setup CLIST                  */
/*                                                                   */
/* Licensed Materials - Property of IBM                              */
/* 5694-A01                                                          */
/* (C) Copyright IBM Corp. 2001                                      */
/* Status = HKY7706                                                  */
/*                                                                   */
/*01* EXTERNAL CLASSIFICATION: OTHER                                 */
/*01* END OF EXTERNAL CLASSIFICATION:                               */
/*                                                                   */
/* FUNCTION:                                                         */
/*                                                                   */
/*   This CLIST will issue the RACF TSO commands necessary to set up*/
/*   security for PKI Services. It must be run from TSO by a user ID*/
/*   that is RACF SPECIAL.                                           */
/*                                                                   */
/* USAGE:                                                            */
/*                                                                   */
/*   1) Read accompanying PKI Services post installation             */
/*      instructions.                                                */
/*   2) Perform necessary prerequisite product installation for      */
/*      the webserver (websphere), LDAP, etc.                        */
/*   3) Make note of any predetermined values such as the LDAP       */
/*      suffix, webserver fully qualified domain name, and the       */
/*      settings contained in the pkiserv.conf file.                 */
/*   4) Copy the CLIST to a data set where you can edit it.          */
/*   5) Examine the entire CLIST, in particular, the configurable    */
/*      section.                                                     */
/*   6) Modify the values in the configurable section as needed for */
/*      your installation.                                           */
/*   7) Run the CLIST. Syntax:                                       */
/*                                                                   */
/*  EX 'data-set-name(IKYSETUP)' 'RUN(YES | NO | PROMPT)'            */
/*                                                                   */
/*  where:  YES - indicates to run CLIST as is                       */
/*          NO - indicates to display the commands only              */
/*          PROMPT - indicates to prompt the user prior              */
/*           to invoking each command                                */
/*                                                                   */
/* DISCLAIMER:                                                       */
/*                                                                   */
/*   This CLIST is not intended to cover every possible customer     */
/*   scenario. Modification of the actual commands to be issued      */
/*   may be required                                                 */
/*                                                                   */
/*********************************************************************/

trace value('O')

/*-------------------------------------------------------------*/
/* configurable section                                        */
/*-------------------------------------------------------------*/

/*---------------------------------*/
/* Part 1 - Things you must change */
```

```
/*--------------------------------*/


/*****************************************************************/
/* This exec will create the certificate, private key,  and      */
/* keyring needed for your certificate authority.                */
/*                                                               */
/* You must update the distinguished name of your certificate    */
/* authority defined below. The suffix of this DN must match     */
/* the suffix set up for your LDAP directory (suffix value from  */
/* your slapd.conf file).                                        */
/*                                                               */
/* Typically, Certificate Authorities have distinguished names   */
/* in the following form:                                        */
/*                                                               */
/*  OU=<your-CA's-friendly-name>,O=<your-organization>,          */
/*        C=<your-2-letter-country-abbreviation>                 */
/*                                                               */
/* e.g., OU=Human Resources Certificate Authority.O=IBM,C=US     */
/*                                                               */
/* If you already have your CA certificate and private key set   */
/* up in RACF, set ca_dn="" and update the ca_label variable to  */
/* equal your CA certificate's label. Note, it must reside       */
/* under CERTAUTH                                                */
/*****************************************************************/

ca_dn=,
 "OU('Human Resources Certificate Authority')",
 "O('Your Company')",
 "C('Your Country 2 Letter Abbreviation')"
ca_label = "Local PKI CA"         /* Label for CA certificate   */

/*****************************************************************/
/* This exec will create the certificate, private key,  and      */
/* keyring needed for your webserver. (Required for SSL.)        */
/*                                                               */
/* You must update the distinguished name of your                */
/* webserver. The Common Name (CN) must match your webserver's   */
/* fully qualified domain name.                                  */
/*                                                               */
/* e.g., CN=www.ibm.com,O=IBM,C=US                               */
/*                                                               */
/* If you already have your webserver configured for SSL, set    */
/* web_dn="".                                                    */
/*****************************************************************/
web_dn=,
 "CN('www.YourCompany.com')",
 "O('Your Company')",
 "L('Your City')",
 "SP('Your Full State or Province Name')",
 "C('Your Country 2 Letter Abbreviation')"

/*****************************************************************/
/* The sample web server protection directives supplied by PKI   */
/* use SSLring for the web server's SAF key ring. If you change  */
/* the value below, you will need to modify the "KeyFile"        */
/* directive in the samples/httpd.conf and samples/httpd2.conf   */
/* files when configuring the web server.                        */
/*                                                               */
/* If you already have your webserver configured for SSL and     */
/* are using a SAF key ring (vs a gskkyman keyfile), then set    */
/* web_ring equal to your webserver's SAF key ring name. If you  */
/* are using a gskkyman keyfile, then set web_ring="". Note,     */
/* you will have to add the CA's certificate to the webserver's  */
/* keyfile manually                                              */
/*****************************************************************/
web_ring = "SSLring"              /* SAF keyring for web server */
```

```
/*****************************************************************/
/* You must provide UID and GID values for the user IDs and      */
/* groups being created below                                    */
/*****************************************************************/
daemon="PKISRVD"                 /* user ID for PKI daemon     */
daemon_uid="554"                 /* uid for PKI daemon         */
surrog="PKISERV"                 /* user ID for the surrogate  */
surrog_uid="555"                 /* uid for the surrogate id   */

/*****************************************************************/
/* pkigroup members are authorized to administer PKI Services    */
/* certificates and certificate requests.  If you know the user */
/* IDs that should be connected to this group, update the        */
/* pkigroup_mem stem variable. If not, you can always connect    */
/* users later.                                                  */
/*                                                               */
/* If you do not wish to have this exec create this group,       */
/* set the group name to ""                                      */
/*                                                               */
/*****************************************************************/
pkigroup="PKIGRP"             /* PKI Services Admin group name  */
pki_gid="655"                 /* PKI Services Admin group id    */
pkigroup_mem.0=0       /* Number of pkigroup members to connect */
pkigroup_mem.1=""


/*-----------------------------------*/
/* Part 2 - Questions you must answer */
/*-----------------------------------*/

/*****************************************************************/
/* Question 1 -  Restrict the surrogate user ID?                 */
/*                                                               */
/* The surrogate user ID is the identity assigned to client      */
/* processes when requesting certificate services. The           */
/* RESTRICTED attribute can be assigned to this ID to limit the */
/* resources available to this user should the user ID be        */
/* hijacked by an unfriendly client (hacker). We recommend       */
/* that you run the surrogate this way. However, this probably   */
/* will cause additional setup work. If you want the RESTRICTED */
/* attribute assigned now, set restrict_surrog=1. Note, you      */
/* can always do this at some later time.                        */
/*****************************************************************/
restrict_surrog=0

/*****************************************************************/
/* Question 2 - Use ICSF?                                        */
/*                                                               */
/* if ICSF key protection is desired for your CA's private key, */
/* set use_icsf=1. ICSF must be configured for PKA support and   */
/* running for this to be successful. Note, you can defer this   */
/* until later if you wish. Read the next paragraph before       */
/* making this decision                                          */
/*****************************************************************/
use_icsf=0

/*****************************************************************/
/* If you set use_icsf=1 above, you will need to restrict access*/
/* to the CA's private key. Unless you indicate otherwise, this */
/* exec will activate the CSFKEYS class, create a profile in the*/
/* CSFKEYS class to protect the CA's private key, and permit     */
/* the PKI Services daemon to use it.                            */
/*                                                               */
/* If you are already using ICSF, then you may have profiles in */
/* the CSFSERV class protecting ICSF services. The PKI Services */
/* daemon would need access to the profile that covers the       */
```

```
/* CSFDSV and CSFDSG services. Also, the PKI Services surrogate */
/* ID would need access to the profile that covers the          */
/* CSFENC and CSFDEC services. You may also have a RACF group   */
/* for authorized ICSF users. Both of these user IDs            */
/* would need to be added to this group.                        */
/*                                                              */
/* Set the following variables as needed:                       */
/*                                                              */
/* csfkeys_profile - Profile to be created in the CSFKEYS class */
/*     Set the value to '' if you don't want the profile        */
/* csfserv_profile - Profile to be created in the CSFSERV class */
/*     e.g., 'CSF*'                                             */
/* csfusers_grp - Group name for authorized ICSF users          */
/*     e.g., 'ICSFUGRP'                                         */
/**************************************************************/
csfkeys_profile='IRR.DIGTCERT.CERTIFAUTH.*'
csfserv_profile='CSF*'
csfusers_grp=''

/**************************************************************/
/* Question 3 - Back up your private key?                     */
/*                                                            */
/* The exec will prompt you to enter a pass phrase to encrypt a */
/* backup copy of your CA's certificate and private key.      */
/* Caution, the text you enter at the prompt WILL be displayed  */
/* at the terminal. Backup is highly recommended.  If you do not*/
/* wish to back up your CA's certificate and private key to a   */
/* pass phrase encrypted data set, set key_backup=0. The back up*/
/* may be done later if the key is not stored in ICSF.        */
/*                                                            */
/* Note, back up is not performed if the CA certificate was not */
/* created by this exec                                       */
/* ***********************************************************/
key_backup=1

/**************************************************************/
/* Question 4 - Set up z/OS UNIX level security?              */
/*                                                            */
/* z/OS UNIX may be set up to operate with a higher level of    */
/* security than traditional UNIX. While we recommend this, it  */
/* difficult to set up. You may want to defer this until later. */
/*                                                            */
/* If you don't want to set up UNIX security now, leave       */
/* unix_sec=0.                                                */
/*                                                            */
/* If you already have UNIX level security established and wish */
/* to continue it, set unix_sec=1.                            */
/*                                                            */
/* If you don't have UNIX level security established and wish   */
/* to establish it now, set unix_sec=2. Note additional manual  */
/* configuration probably will be required. This can be done    */
/* by adding, removing, updating members of the two stem        */
/* variables below. The pgmcntl_dsn stem contains the data set  */
/* names of load libraries that need program control. The       */
/* bpx_userid stem contains the user IDs of your server daemons.*/
/* (These need access to BPX.SERVER and BPX.DAEMON in the       */
/* FACILITY class.) Again, you can defer this until later by    */
/* leaving unix_sec=0                                         */
/**************************************************************/
unix_sec=0
pgmcntl_dsn.0=9  /* Number of program controlled data sets below */
pgmcntl_dsn.1="'CEE.SCEERUN'"
pgmcntl_dsn.2="'CBC.SCLBDLL'"
pgmcntl_dsn.3="'GLD.SGLDLNK'"
pgmcntl_dsn.4="'GSK.SGSKLOAD'"
pgmcntl_dsn.5="'SYS1.CSSLIB'"
pgmcntl_dsn.6="'TCPIP.SEZALINK'"
```

```
pgmcntl_dsn.7="'SYS1.LINKLIB'"
pgmcntl_dsn.8="'CSF.SCSFMOD0'"
pgmcntl_dsn.9="'CSF.SCSFMOD1'"
bpx_userid.0=1 /* Number of additional bpx server ids below */
bpx_userid.1="OMVSKERN"

/*--------------------------------*/
/* Part 3 - Things you can change  */
/*--------------------------------*/

/*****************************************************************/
/* This exec will record results to a log data set if desired.  */
/* the name of the data set is specified below. If you do not   */
/* want log data set recording, set log_dsn="" (Not recommended)*/
/*****************************************************************/
log_dsn="PRIVATE.IKYSETUP.LOG"       /* Under your ID */

/*****************************************************************/
/* Note IKYCVSAM, the sample JCL to create VSAM datasets and    */
/* pkiserv.conf expect the object store and ICL datasets to     */
/* have PKISRVD as their high level qualifier.                  */
/* Changing either "daemon" or "vsamhlq" will                   */
/* require making the same change to IKYCVSAM and pkiserv.conf  */
/*****************************************************************/
vsamhlq=daemon     /* HLQ for VSAM data sets. Same as daemon ID */

web_label = "SSL Cert"             /* Label for web server cert */

ca_expires ="2020/01/01"          /* date the CA certificate for
                                     certificate authority should
                                     expire                    */

web_expires ="2020/01/01"         /* date the certificate for
                                     web server SSL should
                                     expire                    */
ca_ring="CAring"                  /* keyring name for PKI Srvs  */


/*****************************************************************/
/* Data set to contain the backup copy of the CA certificate    */
/* and private key. (pass phrase encrypted PKCS#12 format)      */
/*****************************************************************/
backup_dsn = "'" || daemon || ".PRIVATE.KEY.BACKUP.P12BIN'"

/*****************************************************************/
/* Data set to contain the exported copy of the CA certificate  */
/* (DER encoded). This is to be OPUT to an HFS file later to    */
/* enable easy downloading by clients.                          */
/*****************************************************************/
export_dsn = "'" || daemon || ".PRIVATE.CACERT.DERBIN'"

/*****************************************************************/
/* This EXEC expects the web server to be set up.  If this is   */
/* not the case, please refer to:                               */
/* z/OS HTTP Server Planning, Installing and Using.             */
/* If the user ID assigned to the IBM HTTP Server Daemon is not */
/* WEBSRV, please update the assignment below.                  */
/*****************************************************************/
webserver="WEBSRV"

/*---------------------------------------------------------------*/
/* End of configurable section                                   */
/*---------------------------------------------------------------*/

parse upper arg "RUN(" runopt ")"

if runopt = '' then
```

```
  runopt="NO"
if runopt ^= "YES" & runopt ^= "PROMPT" & runopt ^= "NO" then do
  say "syntax ex 'data-set-name(IKYSETUP)' 'run(yes | no | prompt)'"
  return 8
end
if runopt ^= "YES" & runopt ^= "PROMPT" then
  runopt="NO"

say 'IKYSETUP EXEC invoked ...'
return_code= '0'
max_return_code= '0'
logdata.0=0

if log_dsn ^= "" then do
  say "Allocating log data set" log_dsn "..."
  x = OUTTRAP(MSGS.)
  "FREE FI(IKYLOGDD)"
  "FREE DA("||log_dsn||")"
  "DELETE" log_dsn
  x = OUTTRAP('OFF')
  "ALLOCATE DA("||log_dsn||") FILE(IKYLOGDD) RECFM(V B)" ,
  " LRECL(256) DSORG(PS) BLKSIZE(2560) SP(1,1) TRACKS "
  al_rc= rc
  IF al_rc ^= 0 THEN
    do
      say 'Allocation of log data set failed.'
      return 8
    end
end

call logsay "RUN("runopt") requested"
if runopt="NO" then
  call logsay "Running in test mode. Commands are not being invoked"
call logsay " "
/*********************************************************************/
/* Create the daemon and surrogate user IDs using RACF ADDUSER TSO*/
/* command. Give them an OMVS segment since they will need access */
/* to UNIX System Services.                                          */
/*********************************************************************/


call logsay "Creating users and groups ..."
call tsoserv "ADDUSER " daemon "name('PKI Srvs Daemon')",
  " nopassword",
  " omvs(uid("daemon_uid")",
  " assize(256000000)",
  " threads(512))"

if restricted_surrog=1 then
  resattr="restricted"
else
  resattr=""
call tsoserv "ADDUSER " surrog "nopassword",
  resattr,
  " omvs(uid("surrog_uid"))",
  " name('PKI Srvs Surrogate')"

call tsoserv "SETROPTS EGN GENERIC(DATASET)"

call tsoserv "ADDSD '"vsamhlq".**' UACC(NONE)"

if pkigroup ^= "" then do
  call tsoserv "ADDGROUP " pkigroup "OMVS(GID("pki_gid"))"
  do i = 1 to pkigroup_mem.0
    call tsoserv "CONNECT" pkigroup_mem.i "GROUP("pkigroup")"
  end
end
```

```
/*************************************************************
 * Give the administrators access to the VSAM data sets
 * identified in the [ObjectStore] section of
 * the pkiserv.conf file.
 *************************************************************/
 call logsay "Allowing administrators to access PKI databases ..."
 call tsoserv "PERMIT '"vsamhlq".**' ID("pkigroup") ACCESS(CONTROL)"
 call tsoserv "SETROPTS GENERIC(DATASET) REFRESH"

/*****************************************************************/
/* In order to create and sign digital certificates for others  */
/* you need to define or import in RACF a Certificate Authority */
/* certificate and associated private key.                       */
/* This is done using the RACF RACDCERT GENCERT command.         */
/*****************************************************************/
if ca_dn ^= "" then do
  call logsay "Creating the CA certificate ..."
  certcmd = "RACDCERT GENCERT CERTAUTH SUBJECTSDN("ca_dn")",
   " WITHLABEL('"ca_label"') NOTAFTER(DATE("ca_expires"))"
  if use_icsf=1 & key_backup=0 then
    certcmd= certcmd || " ICSF"

  call tsoserv certcmd

  if key_backup=1 then do
/*****************************************************************/
/* Export certificate and key to PKCS#12 dataset                 */
/*****************************************************************/
    say ""
    say "Enter a passphrase to protect the key. You will need"
    say " this value later if you need to restore the key."
    say ""
    say "Attention, the value will be displayed in the screen:"
    parse pull pp
    call logsay "Backing up the CA certificate ..."
    certcmd = "RACDCERT CERTAUTH EXPORT(LABEL('"ca_label"'))",
     " DSN("backup_dsn") FORMAT(PKCS12DER)",
     " PASSWORD('"pp"')"

    call tsoserv certcmd
  end

  if use_icsf=1 & key_backup=1 then do
/*****************************************************************/
/* If ICSF was requested and key backup, reload the certificate */
/* to get the key migrated to ICSF                               */
/*****************************************************************/
    call logsay "Migrating the CA's private key to ICSF ..."
    certcmd = "RACDCERT CERTAUTH ADD("backup_dsn")",
     " PASSWORD('"pp"') ICSF"

    call tsoserv certcmd
  end

end /* ca_dn ^= "" */
/*****************************************************************/
/* Mark the CA certificate as HIGHTRUST so HostIdMappings       */
/* are honored                                                   */
/*****************************************************************/
call logsay "Marking CA certificate as HIGHTRUST ..."
certcmd = "RACDCERT CERTAUTH ALTER(LABEL('"ca_label"')) HIGHTRUST"
call tsoserv certcmd

/*****************************************************************/
/* The CA certificate must be saved to a data set so that it may */
```

```
/* be OPUT to an HFS file.                                          */
/*******************************************************************/
call logsay "Saving the CA certificate to a data set for OPUT ..."
certcmd = "RACDCERT CERTAUTH EXPORT(LABEL('"ca_label"'))",
  " DSN("export_dsn") FORMAT(CERTDER)"
call tsoserv certcmd

/*******************************************************************/
/* The CA certificate must be placed in a key ring so that         */
/* PKI Services can access it.                                     */
/*******************************************************************/
call logsay "Creating the PKI Services keyring ..."
call tsoserv "RACDCERT ADDRING("ca_ring") ID("daemon")"
call tsoserv "RACDCERT ID("daemon") CONNECT(CERTAUTH",
 " LABEL('"ca_label"')",
 " RING("ca_ring") USAGE(PERSONAL) DEFAULT) "

/*******************************************************************/
/* Create the certificate for the webserver signed by your new CA  */
/*******************************************************************/

if web_dn ^= "" then do
  call logsay "Creating the Webserver SSL certificate and keyring ..."
  call tsoserv "RACDCERT GENCERT ID("webserver") SIGNWITH(CERTAUTH",
   " LABEL('"ca_label"'))",
   " WITHLABEL('"web_label"') SUBJECTSDN("web_dn")",
   " NOTAFTER(DATE("web_expires"))"

/*******************************************************************/
/* Add the certificate to the webserver's RACF (SAF) key ring   */
/*******************************************************************/
  call tsoserv "RACDCERT ADDRING("web_ring") ID("webserver")"
  call tsoserv "RACDCERT  ID("webserver") CONNECT(ID("webserver")",
    " LABEL('"web_label"') RING("web_ring") USAGE(PERSONAL) DEFAULT)"
end /* web_dn ^= "" */

/*******************************************************************/
/* Add the CA certificate to the webserver's RACF (SAF) key ring*/
/*******************************************************************/
if web_ring ^= "" then
  call tsoserv "RACDCERT  ID("webserver") CONNECT(CERTAUTH",
     " LABEL('"ca_label"') RING("web_ring"))"

if unix_sec = 0 then do
/*******************************************************************/
/* Not setting up z/OS UNIX higher security. However, the       */
/* daemon does need access to one server service. So, if the    */
/* daemon user ID is not uid 0, then it must be given read      */
/* access to FACILITY class profile BPX.SERVER                  */
/*******************************************************************/
  if strip(daemon_uid,L,'0') ^= "" then do /* if daemon not uid 0 */
    call logsay "Giving" daemon "access to BPX.SERVER ..."
    call tsoserv "RDEFINE FACILITY BPX.SERVER"
    call tsoserv "PERMIT BPX.SERVER CLASS(FACILITY)",
        " ID("daemon") ACCESS(READ)"
  end
end
else do
  call logsay "Setting up or modifying z/OS UNIX security ..."
  if unix_sec = 2 then do
/*******************************************************************/
/* Set up z/OS UNIX to operate with a higher level of           */
/* security than traditional UNIX, by defining BPX.SERVER and   */
/* BPX.DAEMON classes.                                          */
/*******************************************************************/
    call tsoserv "RDEFINE FACILITY BPX.SERVER"
    call tsoserv "RDEFINE FACILITY BPX.DAEMON"
```

```
    do i = 1 to bpx_userid.0
      call tsoserv "PERMIT BPX.SERVER CLASS(FACILITY)",
        " ID("bpx_userid.i") ACCESS(READ)"
      call tsoserv "PERMIT BPX.DAEMON CLASS(FACILITY)",
        " ID("bpx_userid.i") ACCESS(READ)"
    end
  end


/*****************************************************************/
/* To use the higher level of security, you need to establish   */
/* RACF program control and enable the PKI Services daemon      */
/* user ID and webserver daemon user ID to access protected     */
/* UNIX daemon services.                                        */
/*****************************************************************/
  call tsoserv "PERMIT BPX.SERVER CLASS(FACILITY) ID("daemon")",
   " ACCESS(READ)"
  call tsoserv "PERMIT BPX.DAEMON CLASS(FACILITY) ID("daemon")",
   " ACCESS(READ)"
  call tsoserv "PERMIT BPX.SERVER CLASS(FACILITY) ID("webserver")",
   " ACCESS(UPDATE)"
  call tsoserv "PERMIT BPX.DAEMON CLASS(FACILITY) ID("webserver")",
   " ACCESS(READ)"

  if unix_sec = 2 then do
/*****************************************************************/
/* Set the PKI Services daemon and DLLs up for program control  */
/*****************************************************************/
    call tsoserv "RDEFINE PROGRAM * UACC(NONE)"
    do i = 1 to pgmcntl_dsn.0
     call tsoserv "RALTER PROGRAM * ADDMEM("pgmcntl_dsn.i"//NOPADCHK)",
      " UACC(READ)"
    end
    call tsoserv "SETROPTS WHEN(PROGRAM)"
  end
  call tsoserv "PERMIT * CLASS(PROGRAM)",
   " ID("surrog") ACCESS(READ)"
  call tsoserv "SETROPTS WHEN(PROGRAM) REFRESH"
end /* unix_sec ^= 0 */

/************************************************/
/* Allow the daemon to be a certificate authority */
/************************************************/
call logsay "Allowing the PKI Services daemon to act as a CA ..."
call tsoserv "RDEFINE FACILITY IRR.DIGTCERT.GENCERT"
call tsoserv "RDEFINE FACILITY IRR.DIGTCERT.LISTRING"
call tsoserv "RDEFINE FACILITY IRR.DIGTCERT.LIST"
call tsoserv "PERMIT IRR.DIGTCERT.GENCERT CLASS(FACILITY)",
 " ID("daemon") ACCESS(CONTROL)"
call tsoserv "PERMIT IRR.DIGTCERT.LISTRING CLASS(FACILITY)",
 " ID("daemon") ACCESS(READ)"
call tsoserv "PERMIT IRR.DIGTCERT.LIST CLASS(FACILITY)",
 " ID("daemon") ACCESS(READ)"

/************************************************/
/* Allow the webserver to access its keyring      */
/************************************************/
call logsay "Allowing the Webserver to access its keyring ..."
call tsoserv "PERMIT IRR.DIGTCERT.LISTRING CLASS(FACILITY)",
 " ID("webserver") ACCESS(READ)"
call tsoserv "PERMIT IRR.DIGTCERT.LIST CLASS(FACILITY)",
 " ID("webserver") ACCESS(READ)"

/*****************************************************************/
/* Permit the webserver daemon User ID to switch identity to the */
/* surrogate Id                                                  */
/*****************************************************************/
```

```
call logsay "Allowing the Webserver to switch identity to "surrog" ..."
call tsoserv "SETROPTS CLASSACT(SURROGAT)"
call tsoserv "RDEFINE SURROGAT BPX.SRV."surrog
call tsoserv "PERMIT BPX.SRV."surrog" CLASS(SURROGAT)",
 " ID("webserver") ACCESS(READ)"
call tsoserv "SETROPTS RACLIST(SURROGAT) REFRESH"

if use_icsf then do
/*************************************************/
/* Allow the daemon authorization to use ICSF      */
/*************************************************/
  call logsay "Allowing the PKI Services daemon to use ICSF ..."
  if csfkeys_profile ^= '' | csfserv_profile ^= '' then do
    call tsoserv "SETROPTS  GENERIC(CSFKEYS CSFSERV)"
    call tsoserv "SETROPTS  GENERIC(CSFKEYS CSFSERV) REFRESH"
  end
  if csfkeys_profile ^= '' then do
    call tsoserv "RDEFINE CSFKEYS" csfkeys_profile "UACC(NONE)"
    call tsoserv "PERMIT" csfkeys_profile "CLASS(CSFKEYS)",
     " ID("daemon") ACCESS(READ)"
    call tsoserv "SETROPTS CLASSACT(CSFKEYS) RACLIST(CSFKEYS)"
    call tsoserv "SETROPTS RACLIST(CSFKEYS) REFRESH"
  end
  if csfserv_profile ^= '' then do
    call tsoserv "RDEFINE CSFSERV" csfserv_profile "UACC(NONE)"
    call tsoserv "PERMIT" csfserv_profile "CLASS(CSFSERV)",
     " ID("daemon") ACCESS(READ)"
    call tsoserv "PERMIT" csfserv_profile "CLASS(CSFSERV)",
     " ID("surrog") ACCESS(READ)"
    call tsoserv "SETROPTS CLASSACT(CSFSERV) RACLIST(CSFSERV)"
    call tsoserv "SETROPTS RACLIST(CSFSERV) REFRESH"
  end
  if csfusers_grp ^= '' then do
    call tsoserv "CONNECT" daemon "GROUP(" csfusers_grp ")"
    call tsoserv "CONNECT" surrog "GROUP(" csfusers_grp ")"
  end
end


/*************************************************************
 * Tie the daemon user ID to PKI Services started procedure
 *************************************************************/
 call logsay "Creating the STARTED class profile for the daemon ..."
 call tsoserv "RDEFINE STARTED PKISERVD.* STDATA(USER("daemon"))"
 call tsoserv "SETROPTS CLASSACT(STARTED) RACLIST(STARTED)"
 call tsoserv "SETROPTS RACLIST(STARTED) REFRESH"


/*****************************************************************/
/* Give the surrogate user ID authority to request certificate  */
/* generation functions.                                         */
/*****************************************************************/
call logsay "Allowing "surrog" to request certificate functions ..."
call tsoserv "SETR GENERIC(FACILITY)"
call tsoserv "RDEFINE FACILITY IRR.RPKISERV.**"
call tsoserv "PERMIT IRR.RPKISERV.** CLASS(FACILITY) ID("surrog")",
 " ACCESS(CONTROL)"

/*****************************************************************/
/* The administrative functions of PKI Services are protected   */
/* by the IRR.RPKISERV.PKIADMIN FACILITY class resource.        */
/* The following commands give UPDATE access to the PKI         */
/* services group to allow them to act on certificate           */
/* requests and issued certificates.                            */
/*****************************************************************/
call logsay "Creating the profile to protect PKI Admin functions ..."
```

```
call tsoserv "RDEFINE FACILITY IRR.RPKISERV.PKIADMIN"
call tsoserv "PERMIT IRR.RPKISERV.PKIADMIN CLASS(FACILITY)",
 " ID("pkigroup") ACCESS(UPDATE)"
call tsoserv "PERMIT IRR.RPKISERV.PKIADMIN CLASS(FACILITY)",
 " ID("surrog") ACCESS(NONE)"
call tsoserv "SETROPTS RACLIST(FACILITY) REFRESH"


/*****************************/
/* Done. Now write to the log */
/*****************************/
upper daemon vsamhql export_dsn
call logsay " "
call logsay "-------------------------------------------------"
call logsay "Information needed for PKI Services UNIX set up:"
call logsay "-------------------------------------------------"
call logsay " "
call logsay "The daemon user ID is:"
call logsay " " daemon
call logsay " "
call logsay "The VSAM high level qualifier is:"
call logsay " " vsamhlq
call logsay,
"This is needed for the [ObjectStore] section in pkiserv.conf"
call logsay " "
call logsay "The PKI Services' DER encoded certificate is in data set:"
call logsay " " export_dsn
call logsay,
"This must be OPUT to /var/pkiserv/cacert.der with the BINARY option"
call logsay " "
call logsay "The fully qualified PKI Services' SAF keyring is:"
call logsay " " daemon"/"ca_ring
call logsay,
"This is needed for the [SAF] section in pkiserv.conf"
call logsay " "
if ca_dn ^= "" then do
  call logsay "The PKI Services CA DN is:"
  call norm_dn ca_dn
  call logsay " " dn
  call logsay "The suffix must match the LDAP suffix in slapd.conf"
end
else
  call logsay "CA certificate not created by this exec"
call logsay " "
if web_dn ^= "" then do
  call logsay "The webserver's SAF keyring is:"
  call logsay " " web_ring
  call logsay,
  "This is needed for the KeyFile directive in httpd*.conf files"
  call logsay " "
  call logsay "The Webserver's DN is:"
  call norm_dn web_dn
  call logsay " " dn
  call logsay "The left most RDN must be the webserver's fully",
              "qualified domain name"
end
else
  call logsay,
"Webserver certificate and keyring not created. You must add the CA",
"certificate as a 'trusted root' manually"
call logsay " "
if log_dsn ^= "" then do
  x = OUTTRAP(MSGS.)
  'EXECIO' logdata.0 'DISKW IKYLOGDD (FINIS STEM LOGDATA.'
  'FREE FI(IKYLOGDD)'
  x=OUTTRAP('OFF')
  say "Commands complete. Results written to log data set" log_dsn
```

```
end
/**********************/
/* Exit               */
/**********************/
say 'The IKYSETUP EXEC has completed.'
Exit max_return_code

/************************************************************/
/* tsoserv - echo rc and commands and track highest rc      */
/************************************************************/
tsoserv:
Parse arg cmd
return_code = 0
skipit= 0
if runopt = "NO" | runopt = "PROMPT" then
  call logsay cmd
if runopt = "PROMPT" then do
  say "Run command (y/n) ?"
  parse pull ans
  if substr(ans,1,1) ^= 'Y' & substr(ans,1,1) ^= 'y' then
    skipit= 1
end
if skipit = 0 then
  if runopt = "YES" | runopt = "PROMPT" then do
    msg_status= MSG('ON')
    x=OUTTRAP('rac_ret.')
    Address TSO cmd
    return_code=rc
    y=OUTTRAP('OFF')
    call logsay 'Return code' return_code 'from->' cmd
    If return_code\=0 then do
      Do j=1 to rac_ret.0
      call logsay rac_ret.j
      end
    end
  end
max_return_code= max(max_return_code,return_code)
return return_code
return 0

/************************************************************/
/* logsay - echo messages to the terminal and logdata stem  */
/************************************************************/
logsay:
Parse arg cmd
parse var cmd leftpart " PASSWORD('" pw "') " rightpart
if pw ^= "" then
  cmd= leftpart "PASSWORD('******')" rightpart
say cmd
k= logdata.0 + 1
logdata.k= cmd
logdata.0= k

return 0

/************************************************************/
/* norm_dn - transform the RACF dn keywords to an LDAP dn   */
/************************************************************/
norm_dn:
parse arg in_dn
parse var in_dn q.1 "('" v.1 "')",
                q.2 "('" v.2 "')",
                q.3 "('" v.3 "')",
                q.4 "('" v.4 "')",
                q.5 "('" v.5 "')",
                q.6 "('" v.6 "')",
                q.7 "('" v.7 "')" rest
```

```
dns.= ""
do i = 1 to 7
  q= strip(q.i)
  upper q
  if q = "" then
    leave
  if q = "CN" then
    dns.1= "CN=" ||  v.i
  else
  if q = "T" then
    dns.2= "T=" ||  v.i
  else
  if q = "OU" then
    dns.3= "OU=" ||  v.i
  else
  if q = "O" then
    dns.4= "O=" ||  v.i
  else
  if q = "L" then
    dns.5= "L=" ||  v.i
  else
  if q = "SP" then
    dns.6= "ST=" ||  v.i
  else
    dns.7= "C=" ||  v.i
end
dn= ""
do i = 1 to 7
  if dns.i ^= "" then
    if dn = "" then
      dn= dns.i
    else
      dn= dn || "," || dns.i
end
return 0
```

Chapter 25. Other code samples

This chapter provides code samples for the following files:

- httpd.conf and httpd2.conf, which contain z/OS HTTP Server directives. (See "z/OS HTTP Server configuration directives".)
- IKYCVSAM, which is a sample IDCAMS JCL to create VSAM data sets. (See "IKYCVSAM" on page 289.)
- PKISERVD, which is a sample procedure to start PKI Services daemon. (See "PKISERVD sample procedure to start PKI Services daemon" on page 292.)

Note: Other important programs are contained in other chapters:

- IKYSETUP, a REXX exec to set up RACF profiles — see Chapter 24, "The IKYSETUP REXX exec" on page 269
- pkiserv.envars, the PKI Services environment variables file — see "The pkiserv.envars environment variables file" on page 267
- pkiserv.conf, the PKI Services configuration file — see Chapter 21, "The pkiserv.conf configuration file" on page 221
- pkiserv.tmpl, the PKI Services certificate template file — see Chapter 22, "The pkiserv.tmpl certificate templates file" on page 223.

z/OS HTTP Server configuration directives

The example that follows might not be identical to the code shipped with the product. If you want to see the exact code, look at the httpd.conf sample z/OS HTTP Server configuration directives in the source directory /usr/lpp/pkiserv/samples/.

```
# Licensed Materials - Property of IBM
# 5694-A01
# (C) Copyright IBM Corp. 2001
# Status = HKY7706

# For a secure system, set the default User ID to %%CLIENT%%
UserId     %%CLIENT%%

# SSL support using a SAF keyring
keyfile SSLring SAF
#         OR
# May use a gskkyman key database instead of SAF keyring
#keyfile  /etc/key.kdb

sslmode on
sslport 443
Normalmode on
Protection PublicUser {
        ServerId        PublicUser
        UserID          PKISERV
        Mask            Anyone
}
Protect /PKIServ/public-cgi/*  PublicUser
Protect /PKIServ/ssl-cgi-bin/*  PublicUser
Protect /PKIServ/*  PublicUser

Protection AuthenticatedUser {
        ServerId        AuthenticatedUser
        AuthType        Basic
        PasswdFile      %%SAF%%
        UserID          %%CLIENT%%
        Mask            All
```

Other code samples

```
}
Protect /PKIServ/ssl-cgi-bin/auth/* AuthenticatedUser

Protection SurrogateUser {
        ServerId        SurrogateUser
        AuthType        Basic
        PasswdFile      %%SAF%%
        UserID          PKISERV
        Mask            All
}

Protect /PKIServ/ssl-cgi-bin/surrogateauth/* SurrogateUser

Redirect  /PKIServ/ssl-cgi/*  https://<server-domain-name>/PKIServ/ssl-cgi-bin/*
Redirect  /PKIServ/ssl-cgi/auth/*  \
https://<server-domain-name>/PKIServ/ssl-cgi-bin/auth/*
Redirect  /PKIServ/ssl-cgi/surrogateauth/*  \
https://<server-domain-name>/PKIServ/ssl-cgi-bin/surrogateauth/*


Redirect  /PKIServ/clientauth-cgi/* \ https://<server-domain-name>:1443/PKIServ/clientauth-cgi/*

Exec      /PKIServ/public-cgi/*    <application-root>/PKIServ/public-cgi/*
Exec      /PKIServ/ssl-cgi-bin/*   <application-root>/PKIServ/ssl-cgi-bin/*
Pass      /PKIServ/cacerts/*       /var/pkiserv/*

AddType  .cer  application/x-x509-user-cert      ebcdic  0.5 # Browser Certificate
AddType  .der  application/x-x509-ca-cert        binary  1.0 # CA Certificate
```

The source of the following sample z/OS HTTP Server configuration directives for your /etc/httpd1443.conf file is /usr/lpp/pkiserv/samples/httpd2.conf.

```
# Licensed Materials - Property of IBM
# 5694-A01
# (C) Copyright IBM Corp. 2001
# Status = HKY7706

# For a secure system, set the default User ID to %%CLIENT%%
UserId     %%CLIENT%%


# SSL support using a SAF keyring
keyfile SSLring SAF
#         OR
# May use a gskkyman key database instead of SAF keyring
#keyfile  /etc/key.kdb

sslmode on
sslport 1443
Normalmode off
SSLClientAuth   strong
SSLX500CARoots local_and_x500
SSLX500Host <ldap-server-name>
SSLX500Port <ldap-port-number>
SSLX500UserID <ldap-distinguished-name>
SSLX500Password <ldap-password>

Protection RenewRevokeUser {
        ServerId        RenewRevokeUser
        AuthType        Basic
        UserID          PKISERV
        SSL_CLIENTAUTH  Client
        Mask            Anyone
}

Protect /PKIServ/clientauth-cgi/*  RenewRevokeUser

Protection AuthenticatedAdmin {
        ServerId        AuthenticatedAdmin
        AuthType        Basic

        UserID          %%CERTIF%%
        SSL_CLIENTAUTH  Client
        Mask            Anyone
}
Protect /PKIServ/clientauth-cgi/auth/* AuthenticatedAdmin

Redirect  /PKIServ/public-cgi/*         http://<server-domain-name>/PKIServ/public-cgi/*
Redirect  /PKIServ/ssl-cgi/*            https://<server-domain-name>/PKIServ/ssl-cgi-bin/*

Exec      /PKIServ/clientauth-cgi/* <application-root>/PKIServ/clientauth-cgi-bin/*
```

IKYCVSAM

IKYCVSAM is sample IDCAMS JCL to create VSAM data sets. This is installed as a member of SYS1.SAMPLIB.

Other code samples

Note: The example that follows might not be identical to the code shipped with the product. To view the most current code, see SYS1.SAMPLIB member IKYCVSAM.

```
//IKYCVSAM JOB <job card parameters>
//*********************************************************************
//*  SAMP:      IKYCVSAM                                               *
//*                                                                    *
//*    Licensed Materials - Property of IBM                            *
//*    5694-A01                                                        *
//*    (C) Copyright IBM Corp. 2001                                    *
//*    Status = HKY7706                                                *
//*                                                                    *
//*********************************************************************
//*                                                                    *
//*  This sample JCL may be used to create the VSAM data sets          *
//*  PKI Services utilizes to store certificate requests and           *
//*  issued certificates.                                              *
//*                                                                    *
//*********************************************************************
//*                                                                    *
//*  Caution: This is neither a JCL procedure nor a complete job.      *
//*  Before using this job step, you will have to make the following   *
//*  modifications:                                                    *
//*                                                                    *
//*  1) Change the job card to meet your system requirements.          *
//*                                                                    *
//*  2) If you wish to change the data set qualifiers from the         *
//*     default value change all occurrences of "PKISRVD.VSAM"         *
//*     to a preferred value. If you choose to modify this value, be   *
//*     be sure to also modify the sample configuration file           *
//*     appropriately(/etc/pkiserv/pkiserv.conf).                      *
//*                                                                    *
//*  3) Change vvvvvv to the VOL=SER= value appropriate for the        *
//*     system this job is to be run on.                               *
//*                                                                    *
//*  4) If you wish to change the default userid to own the VSAM       *
//*     data set, change the OWNER(PKISRVD) operand to the userid you  *
//*     want to own the data sets. If you choose to modify this value  *
//*     ensure you have modified the sample setup REXX exec (IKYSETUP) *
//*     to account for this change.                                    *
//*                                                                    *
//*  5) If you wish to change either the primary or secondary record   *
//*     allocation sizes for either the OST or ICL datasets from the   *
//*     default value, update the RECORDS(50 50) operands on the       *
//*     DEFINE CLUSTER or DEFINE ALTERNATE INDEX commands.             *
//*                                                                    *
//*  **Note, do not change any of the numeric values such as           *
//*        CISZ(512).                                                  *
//*--------------------------------------------------------------------*
//*--------------------------------------------------------------------*
//* Delete cluster, AIX, and PATH if they already exist                *
//*--------------------------------------------------------------------*
//DELCLUST EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
    DELETE -
        PKISRVD.VSAM.OST -
        CLUSTER -
        PURGE -
        ERASE
    DELETE -
        PKISRVD.VSAM.ICL -
        CLUSTER -
        PURGE -
        ERASE
    IF LASTCC LT 9 THEN SET MAXCC = 0
```

```
/*
//*-------------------------------------------------------------------*
//* Define KSDS                                                       *
//*-------------------------------------------------------------------*
//DEFKSDS  EXEC PGM=IDCAMS
//VOLUME   DD   UNIT=SYSDA,DISP=SHR,VOL=SER=(vvvvvv)
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
    DEFINE CLUSTER -
        (NAME(PKISRVD.VSAM.OST) -
        FILE(VOLUME) -
        VOL(vvvvvv) -
        RECSZ(1024 32756) -
        INDEXED -
        NOREUSE -
        KEYS(4 0) -
        SHR(2) -
        SPANNED -
        CISZ(512) -
        RECORDS(50 50) -
        OWNER(PKISRVD) ) -
      DATA -
        (NAME(PKISRVD.VSAM.OST.DA)) -
      INDEX -
        (NAME(PKISRVD.VSAM.OST.IX))

    DEFINE CLUSTER -
        (NAME(PKISRVD.VSAM.ICL) -
        FILE(VOLUME) -
        VOL(vvvvvv) -
        RECSZ(1024 32756) -
        INDEXED -
        NOREUSE -
        KEYS(4 0) -
        SHR(2) -
        SPANNED -
        CISZ(512) -
        RECORDS(50 50) -
        OWNER(PKISRVD) ) -
      DATA -
        (NAME(PKISRVD.VSAM.ICL.DA)) -
      INDEX -
        (NAME(PKISRVD.VSAM.ICL.IX))
/*
//*-------------------------------------------------------------------*
//* Repro record of all binary zeros into KSDS                        *
//*-------------------------------------------------------------------*
//REPROKSD EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=*
//SYSDATA  DD   *
. . . . . 80 bytes of all binary zeros . . . . .
//SYSIN    DD   *
   REPRO INFILE(SYSDATA) -
      OUTDATASET(PKISRVD.VSAM.OST)
   REPRO INFILE(SYSDATA) -
      OUTDATASET(PKISRVD.VSAM.ICL)
/*
//*-------------------------------------------------------------------*
//* Define ALTERNATE INDEX AND PATH                                   *
//*-------------------------------------------------------------------*
//DEFALTDX EXEC PGM=IDCAMS
//VOLUME   DD   UNIT=SYSDA,DISP=SHR,VOL=SER=(vvvvvv)
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
    DEFINE ALTERNATEINDEX -
       (NAME(PKISRVD.VSAM.OST.AIX) -
        RELATE(PKISRVD.VSAM.OST)-
```

```
        VOL(vvvvvv) -
        RECORDS(50 50) -
        KEYS(24 44) ) -
      DATA -
        (NAME(PKISRVD.VSAM.OST.AIX.DA)) -
      INDEX -
        (NAME(PKISRVD.VSAM.AIX.IX))
    DEFINE PATH -
        (NAME(PKISRVD.VSAM.OST.PATH) -
         PATHENTRY(PKISRVD.VSAM.OST.AIX))
/*
//*-------------------------------------------------------------------*
//* BUILD ALTERNATE INDEX                                             *
//*-------------------------------------------------------------------*
//BLDINDEX EXEC PGM=IDCAMS
//BASEDD   DD   DSNAME=PKISRVD.VSAM.OST,DISP=OLD
//AIXDD    DD   DSNAME=PKISRVD.VSAM.OST.AIX,DISP=OLD
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
    BLDINDEX INFILE(BASEDD) -
        OUTFILE(AIXDD)
/*
//*-------------------------------------------------------------------*
//* Print out the cluster                                             *
//*-------------------------------------------------------------------*
//PRTCLUST EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
   PRINT -
       INDATASET(PKISRVD.VSAM.OST) CHAR
/*
```

PKISERVD sample procedure to start PKI Services daemon

PKISERVD is the sample procedure to start PKI Services daemon. The PKI Services daemon runs as a started task. The procedure for this can be found in 'SYS1.PROCLIB' member PKISERVD. (PKISERVD is an alias for IKYSPROC.)

PKISERVD contains the TZ (timezone) environment variable, which is the environment variable most likely to change. You need to specify any other environment variables that PKI Services needs in an environment variables file, by default pkiserv.envars. PKISERVD contains FN (file name) and DIR (directory) parameters, to specify the pathname of the environment variables file. You can make any needed changes in PKISERVD, such as updating this pathname.

Recommendation: By default, the pathname for the pkiserv.envars environment variables file is /usr/lpp/pkiserv/samples/pkiserv.envars. If you need to make changes in the environment variables file, you need to copy it from the samples directory to another directory. IBM recommends that you specify your environment variables using an environment variables file under the /etc directory, for example /etc/pkiserv/pkiserv.envars.

The code sample that follows might not be identical to the code shipped with the product. To see the most current code, see 'SYS1.PROCLIB' member PKISERVD.

```
//*******************************************************************
//*                                                                 *
//*             Licensed Materials - Property of IBM                *
//*             5694-A01                                            *
//*             (C) Copyright IBM Corp.  2001                       *
//*             Status=HKY7706                                      *
//*                                                                 *
//*******************************************************************
```

```
//*********************************************************************
//*                                                                   *
//* Procedure for starting the PKI Services Daemon                    *
//*                                                                   *
//*********************************************************************
//PKISERVD PROC REGSIZE=256M,                                         X
//             OUTCLASS='A',                                          X
//             TZ='EST5EDT',                                          X
//             FN='pkiserv.envars',                                   X
//             DIR='/usr/lpp/pkiserv/samples',                        X
//             STDO='1>DD:STDOUT',                                    X
//             STDE='2>DD:STDERR'
//*--------------------------------------------------------------------
//GO       EXEC  PGM=IKYPKID,REGION=&REGSIZE,TIME=1440,
//  PARM=('ENVAR("_CEE_ENVFILE=&DIR/&FN","TZ=&TZ") / &STDO &STDE')
//STDOUT   DD  SYSOUT=&OUTCLASS
//STDERR   DD  SYSOUT=&OUTCLASS
//SYSOUT   DD  SYSOUT=&OUTCLASS
//CEEDUMP  DD  SYSOUT=&OUTCLASS
```

Other code samples

Chapter 26. The certificate validation service

This chapter:

- Provides an overview of PKITP, the PKI Services Trust Policy plug-in for OCSF
- Describes certificate policies and extensions
- Explains how to perform additional OCEP configuration needed for PKITP
- Describes the Trust Policy API, CSSM_TP_PassThrough

Overview

The PKI Services Trust Policy (PKITP) is an OCSF plug-in to perform certificate validation. It supports the following two functions through the implementation of CSSM_TP_PassThrough:

- CertGroupVerify
- FreeEvidence

Server applications running on z/OS can use this function to verify certificates that other network entities (users, other servers, and so forth) present. PKI Services or other certificate authorities may have issued these certificates.

Before using this plug-in, the server administrator must create a SAF key ring containing the certificates of trusted CAs (anchor certificates). (See *z/OS Security Server RACF Command Language Reference* for how to create a SAF key ring.) This key ring can also contain trusted site certificates if appropriate.

The server application must attach to and open this key ring using the OCEP DL plug-in. (See *z/OS SecureWay Security Server Open Cryptographic Enhanced Plug-ins Application Programming* for more information on OCEP and the use of SAF key rings.) The server application must also bind to any needed LDAP directories by attaching to and opening these directories using the OCSF LDAPDL plug-in. These LDAP directories can be internal corporate directories, directories of extranet business partners, directories of public certificate authorities, or combinations of these.

The following figure illustrates this diversity. The uppercase letter boxes are certificate authorities, and the lowercase letter boxes are end-entity certificates.

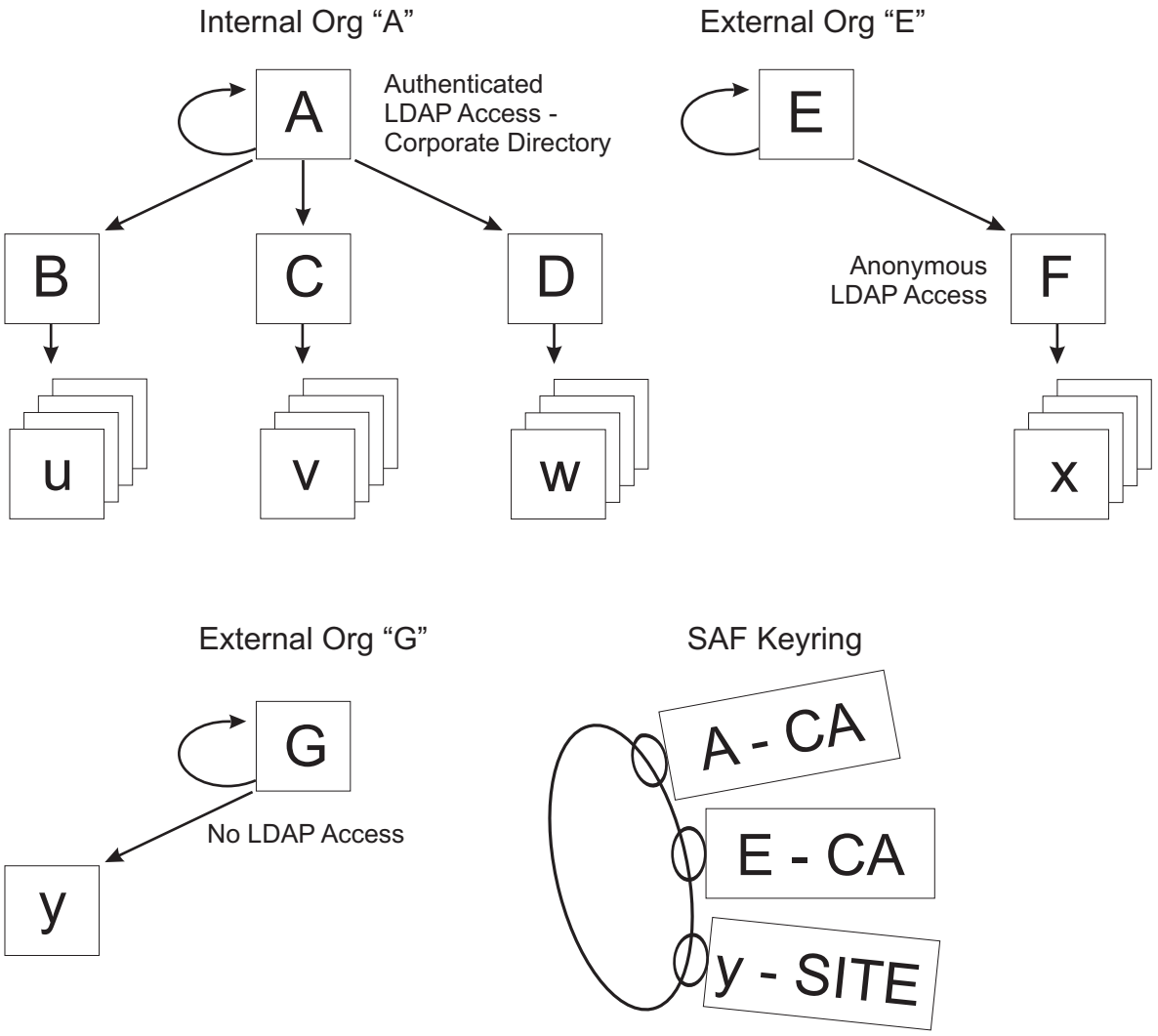


Figure 35. Examples of organizations, certificates, and chains

Organization A represents the local (corporate) certificate hierarchy. It contains one self-signed root certificate 'A.' Perhaps RACF or the Tivoli PKI created this. 'B', 'C', and 'D' are intermediate CAs. They could be separate instances of PKI Services. Certificates issued within this hierarchy are stored in an LDAP directory accessible to corporate server applications.

Organization E represents a public or business partner's certificate hierarchy with an LDAP directory that allows anonymous access. Organization G represents some other certificate hierarchy, in which either the directory does not exist or it is not accessible. The key ring contains three anchor certificates. Certificates 'A' and 'E' are trusted CAs, and there is a business need to trust end-entity certificate y, even though it cannot be verified. If each of these CAs has posted current CRLs to their default locations (PKI Services has no support for CRL distribution points) and all certificate chains to be verified are genuine, the PKITP CertGroupVerify function can validate the following input chains:

- Single certificates x, u, v, or w (PKITP can extract the missing links from the directories.)
- Chains u–B, v–C, w–D, u–B–A, v–C–A, w–D–A, x–F, or x–F–E (These chains have no missing links.)

- Any chain beginning with certificate y (As Figure 35 on page 296 shows, y is in the key ring as a ″SITE.″ Site certificates are trusted regardless.)

Note that, as with the OCEP Trust Policy, non-self-signed (intermediate) CA certificates can be connected to the key ring to shorten the validation path. Doing so has the following consequences:

- Certificate revocation list (CRL) checking is not performed for the anchor certificate in the chain, even if this happens to be an intermediate CA certificate. If the intermediate CA certificate is revoked, PKITP does not detect it.
- A chain containing the parent chain of the intermediate CA cannot be verified.

  Recommendation: When an intermediate CA certificate is connected to the key ring, the certificates that make up its parent chain should be connected as well. This ensures that all chains originating from the intermediate CA or higher can be verified.

Certificate policies

PKITP supports CA and server application-defined certificate policies. CAs can and, in most cases, do establish their own policies for issuing certificates. These policies are declared within issued certificates through the CertificatePolicies extension. When this extension exists and is *not* marked critical, the extension is for informational purposes only (for example, specifying the URL for locating the CA's certificate practice statement (CPS)). When this extension exists and *is* marked critical, the policies identified in the extension restrict the use of the certificate. These restrictions apply to subordinate CA certificates and to end-entity certificates.

Note: For certificates that PKI Services generates, the PKI Services configuration file parameters PolicyRequired and PolicyCritical define whether the extension exists and whether it is marked critical, respectively. By default the certificate policies extension is *not* included in a certificate and the critical flag is *not* turned on. (For more information, see Table 19 on page 43.)

Likewise, a server application can be a general application that wishes to verify certificates for no specific policy or can be an application that was written for a specific purpose and wishes to verify certificates issued for that purpose (policy).

If the server application specifies an explicit set of policies, then at least one of these policies must be present in each certificate of the certification path (chain). Additionally, PKITP extracts the certificate polices marked critical from each certificate in the chain to determine the intersection (that is, only policies listed in every critically marked CertificatePolicies extension are retained.) The server application must indicate that it supports at least one of these polices. If any of these tests is unsuccessful, certificate validation fails.

Certificate extensions

PKITP supports the following certificate extensions:

- SubjectKeyIdentifier — Checked for form only.
- KeyUsage — For CA certificates, the key CertSign flag must be on.
- SubjectAltName — Checked for form only. Must be marked critical if the Subject DN is empty.
- IssuerAltName — Checked for form only. Must be marked critical if the Issuer DN is empty.

- BasicConstraints — For CA certificate, cA flag must be on. Also checked for certification path length.
- CertificatePolicies — See preceding description.
- AuthorityKeyIdentifier — Checked for form only.
- HostIdMappings — Checked for form only.

All other extensions are ignored if they are not marked critical. Unsupported critical extensions prevent certificate validation.

CRL extensions and CRL entry extensions

PKITP supports the following CRL and CRL entry extensions, which are checked for form only:

CRL extensions:

- AuthorityKeyIdentifier
- CRLNumber
- IssuerAltName
- IssuingDistributionPoint

CRL entry extensions:

- CertificateIssuer
- CRLReason
- HoldInstructionCode
- InvalidityDate

All other extensions are ignored if they are not marked critical. Unsupported critical extensions prevent certificate validation.

Files for PKITP

The following table lists files for PKITP:

Table 64. Summary of information about important files for PKITP

File	Description	Source location (default)
Makefile.pkitpsamp	Makefile for pkitpsamp.c.	/usr/lpp/pkiserv/samples/
install_pkitp	Program that registers the PKI Services Trust Policy plug-in with OCSF.	/usr/lpp/pkiserv/bin
pkitp_ivp	This program verifies that the plug-in installed successfully.	/usr/lpp/pkiserv/bin
pkitp.h	Contains #defines for applications calling the PKI Services OCSF Trust Policy.	/usr/lpp/pkiserv/include/
pkitp.so	This is the OCSF Trust Policy plug-in for PKI Services.	/usr/lpp/pkiserv/lib
pkitpsamp.c	Sample application program (in the C language) to call the PKI Trust Policy plug-in.	/usr/lpp/pkiserv/samples

Configuring and getting started with PKITP

If you have not already installed and configured OCEP, you need to do so now. See “Installing and configuring OCSF and OCEP” on page 17 for more information. To install PKITP, you need to follow all of the configuration instructions in *z/OS SecureWay Security Server Open Cryptographic Enhanced Plug-ins Application Programming* and then perform the following post-installation instructions. The PKITP must be registered with OCSF before being used.

Steps for configuring PKITP

Before you begin: If you have not already done so, run the OCSF and OCEP install and verification scripts.

Perform the following steps to install and configure PKITP:

1. Run the PKITP post installation script by entering the following command:

   ```
   /usr/lpp/pkiserv/bin/install_pkitp
   ```

   The program prompts you for certain information. Assuming PKI Services has been installed in its default location, answer the prompts as follows:

Prompt	Response
addin directory?	/usr/lpp/pkiserv/lib
addin filename?	pkitp.so
action? [install|uninstall]	install

   You know you are done and that the installation was successful when you see the following:

   ```
   Installing IBMPKITP...
   Addin successfully installed.
   ```

2. To verify that the installation was successful, run the verification program (/usr/lpp/pkiserv/bin/pkitp_ivp).

   You know you are done and that the verification program ran successfully when you see the following:

   ```
   Starting pkitp IVP
   Initializing CSSM
   CSSM Initialized
   Attaching pkitp
   Attach successful, Detaching pkitp
   Detach of pkitp successful
   Completed pkitp IVP
   ```

Trust Policy API

Programming Interface information

PKITP supports only one API, CSSM_TP_PassThrough. The Globally Unique Identifier (GUID) for this plug-in is: `{01EBC8AC-CC6F-450c-83B4-F0BE0FBE78F9}`. (Before an application can use a module, an installation application must register the module’s name, location, and description with OCSF. The name given to a

module includes both a logical name and a GUID. The logical name is a string the module developer chooses to describe the module. The GUID is a structure used to differentiate between service provider modules in the OCSF registry.)

CSSM_TP_PassThrough

Purpose

This function lets applications call TP module-specific operations that have been exported. For PKITP, the module-specific operations support certificate chain validation, based on the CA and SITE certificates that are contained within a key ring.

Format

```
void * CSSMAPI CSSM_TP_PassThrough
      (CSSM_TP_HANDLE TPHandle,
      CSSM_CL_HANDLE CLHandle,
      CSSM_DL_HANDLE DLHandle,
      CSSM_DB_HANDLE DBHandle,
      CSSM_CC_HANDLE CCHandle,
      uint32 PassThroughId,
      const void *InputParams)
```

Parameters

TPHandle
Handle to this Trust Policy module (PKITP)

CLHandle
Not used. PKITP ignores this.

DLHandle
Not used. PKITP ignores this.

DBHandle
Not used. PKITP ignores this.

CCHandle
Not used. PKITP ignores this.

PassThroughId
Used to indicate the pass-through service requested. Two services are provided:

- Service 1 CertGroupVerify (TP_VERIFY_PASSTHROUGH)
- Service 2 FreeEvidence (TP_FREE_EVIDENCE_PASSTHROUGH)

InputParams
Pointer to the API-caller-provided input parameter structure. The same structure is used for both pass-through functions. It is declared in pkitp.h as follows:

CSSM_TP_PassThrough

```
typedef struct tp_verify_extra {
   /* similar parameters as TP_CertGroupVerify */
   CSSM_CL_HANDLE CLHandle; - Not used. Set to 0. Ignored by PKITP
   CSSM_DL_DB_LIST_PTR DBList; - List of
   CSSM_DL_DB_HANDLE, see below
   CSSM_CSP_HANDLE CSPHandle;           - Handle to IBMSWCSP module
   CSSM_FIELD_PTR PolicyIdentifiers;    - Not used. Set to 0. Ignored by PKITP
   uint32 NumberofPolicyIdentifiers;    - Not used. Set to 0. Ignored by PKITP
   CSSM_TP_STOP_ON VerificationAbortOn; - Must be set to CSSM_TP_STOP_ON_POLICY
   CSSM_CERTGROUP_PTR CertToBeVerified; - Address of cert group struct to verify
   CSSM_DATA_PTR AnchorCerts;           - Not used. Set to 0. Ignored by PKITP
   uint32 NumberofAnchorCerts;          - Not used. Set to 0. Ignored by PKITP

   /* extra parameters: input */
   TP_INITIALPOLICY_PTR InitialPolicy; - Address of policy struct or 0 , see below
   time_t CurrentTime;                  - Not used. Set to 0. Ignored by PKITP
   time_t ValidationTime;               - Time to use for validation, e.g., time(0)

   /* extra parameters: output */
   CSSM_BOOL result;                    - Success indicator
   uint32 DLStatusCode                  - Status code from DL failures
   uint32 DLIndex                       - Index (from 0) into DBList of failing DL
   TP_EVIDENCE_PTR Evidence;            - Address of evidence struct or 0, see below
} TP_VERIFY_EXTRA, *TP_VERIFY_EXTRA_PTR;
```

The DB list

This DBList contains one or more handles to open DB stores. The last entry in this list must be a handle to an OCEPDL DB (a SAF key ring). The key ring is used to declare the list of trusted CA and SITE certificates. Like the OCEP Trust Policy, certificate chains to verify must originate from one of these trusted CAs (anchors) or the end-entity certificate must be one of the SITE certificates. Also like the OCEP Trust Policy, if the security product (SAF) marks any certificate in the candidate chain NOTRUST, the certificate chain fails validation.

The other entries in the list are used for LDAPDL DB stores. PKITP runs through these to locate CRLs and intermediate CA certificates. For each item PKITP requests, the LDAPDLs are queried in the order in which they appear in the list. The search stops the first time an LDAPDL returns an item or when the OCEPDL is reached. No query is made to the OCEPDL to locate CRLs or intermediate CA certificates.

The initial policy

The following optional, caller-provided and initialized structure defines InitialPolicy. PKITP uses the default values if the structure is not provided:

```
 typedef struct tp_initialpolicy {
  /* initial-policy-set - To be used if your application is policy specific */
  uint32 NumberofPolicyIdentifiers;                  - Number of application specific policy OIDs
                                                     (defaults to 0)
  CSSM_OID_PTR PolicyIdentifiers;                    - Address of array of policy OIDs or 0

  /* check certificates against CRLs */
  uint32 useCRLs;                                    - 0 - no CRL processing, 1 -Search for CRLs
                                                     but, continue if search fails, 2 - Strong CRL
                                                     checking (defaults to 2). Must be set to 0 if
                                                     DBList contains no LDAPDL DB stores.

  /* initial-explicit-policy indicator */
  CSSM_BOOL initialExplicitPolicy;                   - Indicates that PKITP should consider the
                                                     policy set critical (defaults to false)

  /* initial-policy-mapping-inhibit indicator *
  CSSM_BOOL initialPolicyMappingInhibit;             - Not used. Ignored by PKITP
} TP_INITIALPOLICY, *TP_INITIALPOLICY_PTR;
```

The evidence

The following optional, caller-provided structure defines the evidence. This structure is used to return information relative to the validation decision PKITP makes. The caller must free the data areas returned. (The FreeEvidence pass-through function is provided for this.)

```
typedef struct tp_evidence {
  /* valid certification path if validation succeeds */
  CSSM_CERTGROUP_PTR CompleteCertGroup;                    - Cert group from EE to anchor CA

  /* relevant CRL if validation fails */
  CSSM_DATA_PTR CRL;                                       - CRL for revoked cert or incorrect CRL

  /* relevant certificate if validation fails */
  CSSM_DATA_PTR Cert;                                      - Certificate causing the failure

  /* authority-constrained-policy */
  CSSM_BOOL authAnyPolicy;                                 - false - critical certificatePolicies found

  uint32 NumberofAuthCertPolicyIdentifiers;                - Nonzero if authAnyPolicy is false
  CSSM_OID_PTR AuthCertPolicyIdentifiers;                  - Array of policy OIDs

  /* list of policy mappings that occurred */
  uint32 NumberOfMappedPolicies;                           - Not used. Ignored by PKITP
  TP_CSSM_OID_PAIR_PTR mappedPolicies;                     - Not used. Ignored by PKITP
} TP_EVIDENCE, *TP_EVIDENCE_PTR;
```

Error codes

Table 65 lists the error codes that are unique to PKI Services OCSF Trust Policy (PKITP).

Table 65. PKI Services OCSF Trust Policy (PKITP) error codes

Decimal Value	Error Description
8001	Certificate encoding error. Incorrect CertificatePolicies extension.
8002	Certificate policies violation.
8003	Incorrect certificate distinguished name chaining.
8004	Certificate encoding error. Subject name missing.
8006	Incorrect certificate BasicConstraints extension - cA flag off in signing certificate.

Table 65. PKI Services OCSF Trust Policy (PKITP) error codes (continued)

Decimal Value	Error Description
8008	Incorrect certificate keyUsage extension - keyCertSign flag off in signing certificate.
8010	Unsupported AltName form in certificate.
8013	Certificate or CRL encoding error. Signature algorithm mismatch.
8014	Certificate encoding error. Incorrect version.
8015	CRL encoding error. Incorrect version.
8016	Unsupported critical extension in certificate.
8017	Unsupported critical extension in CRL.
8018	Unsupported critical entry extension in CRL.
8019	Certificate encoding error. Duplicate extension.
8020	CRL encoding error. Duplicate extension.
8021	Certificate signature failed verification.
8022	CRL signature failed verification.
8023	Incorrect date range in certificate or CRL. NotAfter earlier than NotBefore.
8024	Certificate's date range is in the future.
8025	Certificate has expired.
8026	CRL's date range is in the future.
8027	CRL has expired.
8028	DBList incorrect, no LDAPDL DBs or non-LDAPDL specified.
8029	CRL not found.
8030	Certificate is revoked.
8031	Unable to build certificate chain.
8033	Certificate not trusted.
8501	Unexpected status code returned from accessing LDAPDL.
8502	Unexpected status code returned from accessing OCEPDL.
8503	DBList incorrect, no OCEPDL DB or DB empty.

Providing the certificate validation service

To perform certificate validation, your server application calls the CSSM_TP_PassThrough API (see CSSM_TP_PassThrough on page 301), passing it the certificate chain to verify. The API returns a boolean value indicating success or failure, along with additional information about the certificate chain. The pkitpsamp.c code sample that follows is provided as an aid for developing your own server application. By default, you can find this file in the /usr/lpp/pkiserv/samples directory.

Steps for building the sample application

Perform the following steps to build the sample application:

1. Copy the pkitpsamp.c program and Makefile.pkitpsamp to the current directory by entering the following commands:

   ```
   cp /usr/lpp/pkiserv/samples/pkitpsamp.c pkitpsamp.c
   cp /usr/lpp/pkiserv/samples/Makefile.pkitpsamp Makefile
   ```

---

2. Before compiling pkitpsamp.c, you need to edit some data (for example, information about how you want the Trust Policy to operate and where your LDAP is located). In the pkitpsamp.c code (see “Example of using PKITP program” on page 306), find the section that begins with a block comment that says `// Start of application specific options`. Update the code as necessary up to the block comment that says `// End of application specific options`:

   a. If the number of LDAP servers is not `1`, change `NUM_LDAPS`.

   b. Update `ldap_info` by specifying your LDAP server and port (*`myldap.mycompany`*`.com:389` in the sample program).

      Note: If you have more than one LDAP server, you need to provide this information for each LDAP server.

   c. Specify the user ID and keyname for the SAF key ring containing trusted CA or site certificates (in the sample, this is G9VEMER/*myring*).

   d. If necessary, change the value of the CSSM_GUID:
      `IBMSWCSP_GUID` This is the value in the sample.
      `IBMWKCSP_GUID` This GUID is for use when the IBMSWCSP plug-in is not available (for example, US export-controlled locations).

   e. If necessary, change the value of USECRLS:
      `0` This means using no CRL processing. (You must specify 0 if you have no LDAP servers.)
      `1` This means querying LDAP for CRLs and processing those found. This is the value in the sample.
      `2` This means using strong CRL checking. (With strong CRL checking, a valid CRL must be found for each CA certificate in the chain.)

   f. If necessary, change `NUM_POLICIES`, the policies that the application calling PKITP uses. In the sample, this is `2`. For each policy, specify the DER-encoded policy information.

   g. If necessary, change `INITIALExplicitPolicy` from the default of `FALSE` to `TRUE` if you want PKITP to require all certificates in the chain to have at least one `policydata` in the preceding list.

---

3. Compile and link to produce the executable, pkitpsamp, by entering the following command:

```
make
```

4. Run the pkitpsamp.c in your own directory by entering the following command:

```
pkitpsamp
```

Example of using PKITP program

Note: The example that follows might not be identical to the code shipped with the product. If you want to see the most current code, look in the /usr/lpp/pkiserv/samples directory.

```
/*********************************************************************/
/* This file contains sample code. IBM provides this code on an      */
/* 'as is' basis without warranty of any kind, either express or     */
/* implied, including but not limited to, the implied warranties     */
/* of merchantability or fitness for a particular purpose.           */
/*********************************************************************/
/*********************************************************************/
/*                                                                   */
/*    Licensed Materials - Property of IBM                           */
/*    5694-A01                                                       */
/*    (C) Copyright IBM Corp. 2001                                   */
/*    Status = HKY7706                                               */
/*                                                                   */
/*********************************************************************/
/*    Sample use of IBM PKITP program                                */
/*                                                                   */
/*    Purpose: Program attaches needed CSSM modules, then prompts    */
/*             the user for filename(s) containing DER encoded       */
/*             certificates.  The certificate(s) are read from the   */
/*             file, then passed to PKITP for verification.          */
/*             A summary of the results are printed to stdout.       */
/*                                                                   */
/* Caution: In order to run this sample program, modification MUST   */
/*          BE MADE to several values assigned to the following      */
/*          variables that are defined between the block comment     */
/*          containing the text "Start of application specific       */
/*          options" and the block comment containing the text       */
/*          "End of application specific options"(without the        */
/*          quotation marks):                                        */
/*                                                                   */
/*   #define NUM_LDAPS 1                                             */
/*           Define the number of LDAP servers that PKITP should     */
/*           query for certificates, CRLs and ARLs.  This can be 0,  */
/*           if entire certificate chain will be passed as input to  */
/*           PKITP AND caller requests to NOT process CRLs/ARLs (see */
/*           useCRLs option below).                                  */
/*                                                                   */
/*   struct ldap_info ldapserver[NUM_LDAPS] =                        */
/*                     { "@LDAPSERVERNAME:PORTNUMBER@",              */
/*                       "@LDAPUSER@",                               */
/*                       "@LDAPUSERPASSWORD@"};                      */
/*           If NUM_LDAPS > 0, then ldapserver array should define   */
/*           the LDAP server:port, user and password for each LDAP   */
/*           server. Replace @LDAPSERVERNAME:PORTNUMBER@ with the    */
/*           appropriate ldap server name and port number (e.g       */
/*           myldap.mycompany.com:389 ). Replace @LDAPUSER@ with the */
/*           appropriate ldap admin user name (e.g cn=root) and      */
/*           @LDAPUSERPASSWORD@ with the password for the specified  */
/*           ldap user name (e.g rootpw)                             */
/*                                                                   */
/*   char keyring[] = "@USERID@/@KEYRINGNAME@";                      */
```

```
/*          Define the SAF keyring containing trusted CA and/or     */
/*          site certificates. Format is "USERID/keyname". Replace  */
/*          @USERID@ with the userid of the keyring owner and       */
/*          @KEYRINGNAME@ with the name of the keyring. (e.g        */
/*          IBMUSER/CAring) Note that the userid and the keyring    */
/*          names are case sensitive so the userid is all           */
/*          uppercase and the keyring name is mixed case in this    */
/*          example.                                                */
/*                                                                  */
/*   CSSM_GUID csp_guid = IBMSWCSP_GUID;                            */
/*          The CSSM GUID (globally unique id) for Cryptographic    */
/*          Service Provider. PKITP will call the specified CSP to  */
/*          verify signatures in the certificate chain. The         */
/*          csp_guid variable is set to use the software CSP,       */
/*          IBMSWCSP_GUID, but may to set to either IBMWKCSP_GUID   */
/*          or IBMCCA_GUID.                                         */
/*                                                                  */
/*   #define USECRLS 1                                              */
/*          Define how the useCRLs option should be set.            */
/*            Set to 0 if no CRL processing is to be performed      */
/*            Set to 1, if LDAP is to be queried for CRLs and       */
/*            process the CRLs found.                               */
/*            Set to 2, for strong CRL checking (With strong CRL    */
/*            cheching, a valid CRL must be found for each CA       */
/*            certificate in the chain.)                            */
/*                                                                  */
/*   #define NUM_POLICIES 2                                         */
/*   static unsigned char my_policy1[5] =                           */
/*         {0x06,0x03,0x2a,0x03,0x04};       // DER encoded 1.2.3.4 */
/*   static unsigned char my_policy2[7] =                           */
/*         {0x06,0x05,0x2a,0x03,0x03,0x02,0x01}; // DER 1.2.3.3.2.1 */
/*   CSSM_DATA policydata[NUM_POLICIES] =                           */
/*         {{sizeof(my_policy1),(unsigned char *)my_policy1},       */
/*          {sizeof(my_policy2),(unsigned char *)my_policy2}};      */
/*          Define the policies that the application calling PKITP  */
/*          uses. These become important if a certificate in the    */
/*          certificate chain has a critically marked policy        */
/*          extension.  At least one policy that is listed in such  */
/*          a critically marked policy extension, must appear in    */
/*          the list defined here or PKITP will return certicate    */
/*          policy error.                                           */
/*                                                                  */
/*   #define INITIALExplicitPolicy FALSE                            */
/*          Set to true if you want PKITP to require that all       */
/*          certificates in chain to have at least one policy       */
/*          listed by the policydata defined above.                 */
/*                                                                  */
/********************************************************************/
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <cssm.h>
#include <ibmocepdl.h>
#include <ibmswcsp.h>
#include <cssmapi.h>
#include <cssmtype.h>
#include <pkitp.h>
#include <ldapdl.h>

struct ldap_info
 {
 char * ldapserver;
 char * ldapauthuser;
 char * ldapauthpass;
 };

//-------------------------------------------------------------
```

CSSM_TP_PassThrough

```
// storage function definitions needed to talk to CSSM
//------------------------------------------------------------

#ifdef __cplusplus
extern "C"
#endif
void * OurMalloc(size_t size, void * allocRef)
{
   return malloc(size);
}

#ifdef __cplusplus
extern "C"
#endif
void OurFree(void* memPtr, void * allocRef)
{
   free(memPtr);
}

#ifdef __cplusplus
extern "C"
#endif
void * OurRealloc(void * memPtr,
                      size_t size,
                      void * allocRef)
{
   return realloc(memPtr, size);
}

#ifdef __cplusplus
extern "C"
#endif
void * OurCalloc(size_t num,
              size_t size,
              void* allocRef)
{
   return calloc(num, size);
}

static CSSM_API_MEMORY_FUNCS memoryFuncs; // used to pass function addresses to CSSM
static CSSM_CSP_HANDLE       ibm_csp_handle;

//------------------------------------------------------------
// internal function declarations
//------------------------------------------------------------
int connectTP(char * ringname,
              int number_ldap,
              struct ldap_info *,
              CSSM_DL_DB_LIST *,
              CSSM_TP_HANDLE *);

void disconnectTP(CSSM_DL_DB_LIST *, CSSM_TP_HANDLE);

int buildCertGroup(CSSM_CERTGROUP *, char * [], uint32);
void verifyCertGroup(CSSM_CERTGROUP certgroup,
                     CSSM_DL_DB_LIST * datasources_ptr,
                     CSSM_TP_HANDLE tphandle);

void
reportCertGroupVerify
   (TP_VERIFY_EXTRA extraVerifyInfo);

void printEvidence(TP_EVIDENCE_PTR evidence_ptr);

void freeCertGroup(CSSM_CERTGROUP * certGroupPtr);
////////////////////////////////////////////////////////////////////////////
//
```

```
// Start of application specific options
//
// The defines and declarations that follow should be altered to fit the
// particular application calling PKITP.
//
////////////////////////////////////////////////////////////////////////////

////////////////////////////////////////////////////////////////////////////
// Define the number of LDAP servers that PKITP should query for certificates,
// CRLs and ARLs.  This can be 0, if entire certificate chain will be passed as
// input to PKITP AND caller requests to NOT process CRLs/ARLs (see useCRLs
// option below).
//
// If NUM_LDAPS > 0, then ldapserver array should define the LDAP server:port,
// user and password for each LDAP server, as this example shows.
////////////////////////////////////////////////////////////////////////////
#define NUM_LDAPS 1

struct ldap_info ldapserver[NUM_LDAPS] =
                                  { "@LDAPSERVERNAME:PORTNUMBER@",  // LDAP server:port
                                    "@LDAPUSER@",                   // user
                                    "@LDAPUSERPASSWORD@"};          // password

////////////////////////////////////////////////////////////////////////////
// Define the SAF keyring containing trusted CA and/or site certificates.
// Format is "USERID/keyname"
////////////////////////////////////////////////////////////////////////////
char keyring[] = "@USERID@/@KEYRINGNAME@";

////////////////////////////////////////////////////////////////////////////
// The CSSM GUID (globally unique id) for Cryptographic Service Provider.
// PKITP will call the specified CSP to verify signatures in the certificate
// chain.  Must be either:   IBMSWCSP_GUID, IBMWKCSP_GUID or IBMCCA_GUID
////////////////////////////////////////////////////////////////////////////

CSSM_GUID csp_guid = IBMSWCSP_GUID;

////////////////////////////////////////////////////////////////////////////
// Define how the useCRLs option should be set.
// Set to 0 if no CRL processing to be done
// Set to 1, if we are to query LDAP for CRLs and process those found
// Set to 2, for strong CRL checking -- must find CRLs in LDAP.
////////////////////////////////////////////////////////////////////////////

#define USECRLS 1

////////////////////////////////////////////////////////////////////////////
// Define the policies that the application calling PKITP uses.
//
// These become important if a certificate in the certificate chain has a
// critically marked policy extension.  At least one policy
// that is listed in such a critically marked policy extension, must appear
// in the list defined here or PKITP will return certicate policy error.
////////////////////////////////////////////////////////////////////////////

#define NUM_POLICIES 2

static unsigned char my_policy1[5] = {0x06,0x03,0x2a,0x03,0x04};  // DER encoded 1.2.3.4
static unsigned char my_policy2[7] = {0x06,0x05,0x2a,0x03,0x03,0x02,0x01};  // DER 1.2.3.3.2.1

CSSM_DATA policydata[NUM_POLICIES] = {{sizeof(my_policy1),(unsigned char *)my_policy1},
                                      {sizeof(my_policy2),(unsigned char *)my_policy2}};


#define INITIALExplicitPolicy FALSE  // Set to true if you want PKITP to require that all
                                     // certificates in chain have at least one policy
                                     // listed by our policydata defined above
```

```
////////////////////////////////////////////////////////////////////////
//
// End of application specific options
//
////////////////////// ////////////////////////////////////////////////////

//------------------------------------------------------------
// main
//------------------------------------------------------------
int
main(int argc, char* argv[])
{

 CSSM_DL_DB_LIST datasources;
 CSSM_TP_HANDLE  tphandle = 0;
 CSSM_CERTGROUP certGroup;
 int repeating = 1;
 char buffer[1024];
 int num_certs = 0;
 char * cert_files[25];
 char * next_file;
 char * input;

 int rc;

 rc = connectTP(keyring,NUM_LDAPS,ldapserver, &datasources, &tphandle);
 if (rc == 0)
  {
  ////////////////////////////////////////////////////////////////////////
  // prompt for certificates to verify
  ////////////////////////////////////////////////////////////////////////
  do
    {
    num_certs = 0;
    printf("Enter filename(s) of certificate(s).  (List EE first).  ");
    printf("Blank line to quit.\n");

    if ((input = gets(buffer)) != NULL)      // get input line
      {
      next_file = strtok(input," ");
      while ((next_file != NULL) && (num_certs < 25)) // tokenize it
        {
        cert_files[num_certs] = next_file;
        num_certs++;
        next_file = strtok(NULL," ");
        }
      }

    ////////////////////////////////////////////////////////////////////
    // If we were given a list of files containing certificates, input them to TP
    ////////////////////////////////////////////////////////////////////
    if (num_certs > 0)
      {
      rc = buildCertGroup(&certGroup, cert_files, num_certs);
      if (rc == 0)
        {
        verifyCertGroup(certGroup, &datasources, tphandle);
        freeCertGroup(&certGroup);
        }
      }
    } while (num_certs > 0);
  }
 disconnectTP(&datasources, tphandle);
}

//------------------------------------------------------------
```

```
// connectTP
//
// Purpose:  connect to the datasources PKITP needs
// then connect to the PKITP
//
// Input:  ringname - string containing "USERID/ringname" of SAF
//                   keyring containing trusted CA and/or SITE certificates
//         number_ldap - number of ldap servers
//         ldaps - array of ldap_info structures
//
// Output: The CSSM_DL_DB_LIST structure addressed by datasources will have
//         been initialized with the various handles that CSSM_ModuleAttach
//         and CSSM_DL_DbOpen calls have returned
//         The CSSM_TP_HANDLE addressed by tphandle_ptr will have been initialied.
//         int returned will be 0 if successful, -1 if not successful.
//-------------------------------------------------------------

int connectTP(char * ringname,
              int number_ldap,
              struct ldap_info * ldaps,
              CSSM_DL_DB_LIST * datasources_ptr,
              CSSM_TP_HANDLE * tphandle_ptr)
  {
   uint32 status = 0;
   int z;
   CSSM_VERSION cssm_version = {CSSM_MAJOR, CSSM_MINOR};
   CSSM_VERSION  CSP_version = {IBMSWCSP_MAJOR_VERSION, IBMSWCSP_MINOR_VERSION};
   CSSM_DB_ACCESS_TYPE access = { CSSM_TRUE,
                                  CSSM_FALSE,
                                  CSSM_FALSE,
                                  CSSM_FALSE};
   CSSM_VERSION DL_version;
   CSSM_DL_HANDLE LDAP_dlhandle;
   CSSM_MODULE_INFO* moduleInfoPtr;
   void * voidptr;
   CSSM_DB_ACCESS_TYPE accessRequest = { CSSM_TRUE,     // ReadAccess
                                        CSSM_TRUE,     // WriteAccess
                                        CSSM_FALSE,    // PrivilegedMode
                                        CSSM_FALSE  }; // Asynchronous


   memoryFuncs.malloc_func = OurMalloc;
   memoryFuncs.free_func = OurFree;
   memoryFuncs.realloc_func = OurRealloc;
   memoryFuncs.calloc_func = OurCalloc;
   memoryFuncs.AllocRef = NULL;

   DL_version.Major = IBMOCEPDL_MAJOR_VERSION;
   DL_version.Minor = IBMOCEPDL_MINOR_VERSION;

   datasources_ptr->NumHandles = number_ldap + 1;
   voidptr = malloc(sizeof(CSSM_DL_DB_HANDLE)*(number_ldap +1));   // get storage for DBlist
   memset(voidptr,0,(sizeof(CSSM_DL_DB_HANDLE)*(number_ldap +1))); // zero it
   datasources_ptr->DLDBHandle = (CSSM_DL_DB_HANDLE *)voidptr;

   if (CSSM_Init(&cssm_version, &memoryFuncs, NULL) != CSSM_OK)
        {
        printf("Failed CSSM_Init: %d, line %d\n",CSSM_GetError()->error,__LINE__);
        return -1;
        }

   ////////////////////////////////////////////////////////////////////////////
   // attach to LDAP and open each LDAP DB
   ////////////////////////////////////////////////////////////////////////////
   if (number_ldap > 0)              // if we have any LDAP sources
     {
     moduleInfoPtr = CSSM_GetModuleInfo((CSSM_GUID*)&LDAPDL_GUID,
```

```
                                    CSSM_SERVICE_DL,
                                    CSSM_ALL_SUBSERVICES,
                                    CSSM_INFO_LEVEL_ALL_ATTR);
  if (!moduleInfoPtr)
    {
    printf("Failed CSSM_GetModduleInfo: %d, line %d\n",CSSM_GetError()->error,__LINE__);
    return -1;
    }

  LDAP_dlhandle = CSSM_ModuleAttach((CSSM_GUID*)&LDAPDL_GUID,
                                &moduleInfoPtr->Version,
                                &memoryFuncs,
                                0,
                                0,
                                0,
                                NULL,
                                NULL);
  if (!LDAP_dlhandle)
    {
    printf("Failed CSSM_ModuleAttach: %d, line %d\n",CSSM_GetError()->error,__LINE__);
    return -1;
    }

  // connect to multiple database instances

  //-------------------------------------------------------------
  // fill in LDAP DL authentication information:
  // necessary only if user is supplying a name and password
  //-------------------------------------------------------------
  for (z = 0; z < number_ldap; z++)             // for each LDAP source
    {
    LDAP_BIND_PARMS bindParms;
    CSSM_USER_AUTHENTICATION userAuthentication = {0,0};
    CSSM_DATA userCredential = {0,0};
    CSSM_USER_AUTHENTICATION_PTR userAuthenticationPtr = 0;

    datasources_ptr->DLDBHandle[z].DLHandle = LDAP_dlhandle;
    if (ldaps[z].ldapauthuser && ldaps[z].ldapauthpass)
      {
      //-------------------------------------------------------------
      // fill in LDAP DL specific data structure: LDAP_BIND_PARMS
      //-------------------------------------------------------------
      bindParms.DN = ldaps[z].ldapauthuser;
      bindParms.SASL = 0;
      bindParms.credentials.Data = (uint8 *)ldaps[z].ldapauthpass;
      bindParms.credentials.Length = strlen(ldaps[z].ldapauthpass)+1;
      userCredential.Length = sizeof(LDAP_BIND_PARMS);
      userCredential.Data   = (unsigned char*)&bindParms;
      userAuthentication.Credential = &userCredential;
      userAuthenticationPtr = &userAuthentication;
      }

   //-------------------------------------------------------------
   // Open LDAP DL Database
   //-------------------------------------------------------------
   datasources_ptr->DLDBHandle[z].DBHandle = CSSM_DL_DbOpen(LDAP_dlhandle,
                                    ldaps[z].ldapserver,
                                    &accessRequest,
                                    userAuthenticationPtr,
                                    (void *)0);
   if (!datasources_ptr->DLDBHandle[z].DBHandle)
     {
     printf("Failed CSSM_DL_DbOpen %d, line %d\n", CSSM_GetError()->error,__LINE__);
     return -1;
     }

  }                             // end of for each each LDAP source
```

```
  if (CSSM_FreeModuleInfo(moduleInfoPtr) == CSSM_FAIL)
  {
  printf("Failed CSSM_FreeModuleInfo, line %d, error %d\n",__LINE__,
         CSSM_GetError()->error);
  // This is not a catastrophic error, we'll continue
  }
}                               // end if we have any LDAP sources

//////////////////////////////////////////////////////////////////////
// Attach to OCEP DL (to access RACF keyring)
//////////////////////////////////////////////////////////////////////

datasources_ptr->DLDBHandle[number_ldap].DLHandle =
                                CSSM_ModuleAttach(&IBMOCEPDL_GUID,
                                &DL_version,
                                &memoryFuncs,
                                0,
                                0,
                                0,
                                NULL,
                                NULL);


if (!(datasources_ptr->DLDBHandle[number_ldap].DLHandle))
  {
  printf("Failed CSSM_ModuleAttach: %d, line %d\n",CSSM_GetError()->error,__LINE__);
  return -1;
  }

datasources_ptr->DLDBHandle[number_ldap].DBHandle =
             CSSM_DL_DbOpen(datasources_ptr->DLDBHandle[number_ldap].DLHandle,
                            ringname,
                            &access,
                            NULL,
                            NULL);

if (!(datasources_ptr->DLDBHandle[number_ldap].DBHandle))
  {
  printf("Failed CSSM_DL_DbOpen %d, line %d\n", CSSM_GetError()->error,__LINE__);
  return -1;
  }

//////////////////////////////////////////////////////////////////////
// Attach to cryptographic service provider - PKITP uses for signature checking
//////////////////////////////////////////////////////////////////////
ibm_csp_handle = CSSM_ModuleAttach(&csp_guid, &CSP_version,
                                   &memoryFuncs, 0, 0, 0, NULL, NULL);
if (!ibm_csp_handle)
  {
  printf("Failed CSSM_ModuleAttach %d, line %d\n", CSSM_GetError()->error,__LINE__);
  return -1;
  }


//////////////////////////////////////////////////////////////////////
// Attach to PKITP
//////////////////////////////////////////////////////////////////////
moduleInfoPtr = CSSM_GetModuleInfo((CSSM_GUID*)&PKITP_GUID,
                                   CSSM_SERVICE_TP,
                                   CSSM_ALL_SUBSERVICES,
                                   CSSM_INFO_LEVEL_ALL_ATTR);
if (!moduleInfoPtr)
  {
  printf("Failed CSSM_GetModduleInfo: %d, line %d\n",CSSM_GetError()->error,__LINE__);
  return -1;
  }
```

```
   *(tphandle_ptr) = CSSM_ModuleAttach((CSSM_GUID*)&PKITP_GUID,
                                &moduleInfoPtr->Version,
                                &memoryFuncs,
                                0,
                                0,
                                0,
                                NULL,
                                NULL);
   if (!(*tphandle_ptr))
     {
     printf("Failed CSSM_ModuleAttach: %d, line %d\n",CSSM_GetError()->error,__LINE__);
     return -1;
     }

   if (CSSM_FreeModuleInfo(moduleInfoPtr) == CSSM_FAIL)
      {
      printf("Failed CSSM_FreeModuleInfo, line %d, error %d\n",__LINE__,
            CSSM_GetError()->error);
      // This is not a catastrophic error, we'll continue
     }

  return 0;
}

//-------------------------------------------------------------
// disconnectTP
//
// Purpose:  to close any open databases and detach any CSSM modules
//           that connectTP attached
//
// Input:  The CSSM_DL_DB_LIST structure, CSSM_TP_HANDLE,
//         ibm_csp_handle (static variable referenced both places)
//         that were initialized by connectTP.
//
// Output: None
//-------------------------------------------------------------

void disconnectTP(CSSM_DL_DB_LIST * datasources_ptr, CSSM_TP_HANDLE tphandle)
  {
  int x;
  int status;

  ////////////////////////////////////////////////////////////////
  // Sever ties to LDAP
  // For each LDAP database opened -- call CSSM_DL_DbClose
  ////////////////////////////////////////////////////////////////

  for (x = 0; x < datasources_ptr->NumHandles - 1; x++)
     {
      // we close each ldap database separately
      if (datasources_ptr->DLDBHandle[x].DBHandle)        // if we opened database
        {
        status = CSSM_DL_DbClose(datasources_ptr->DLDBHandle[x]);
        if (status != 0)
          {
          printf("Failed CSSM_DL_DbClose %d, line %d\n", CSSM_GetError()->error,__LINE__);
          // we continue trying to close other stuff
          }
        }
     }

   ////////////////////////////////////////////////////////////////
   // Now detach the LDAP module
   ////////////////////////////////////////////////////////////////
   if (datasources_ptr->DLDBHandle[0].DLHandle)
     {
```

```
    if ((status = CSSM_ModuleDetach(datasources_ptr->DLDBHandle[0].DLHandle)) != 0)
      {
      printf("Failed CSSM_ModuleDetach: %d, line %d\n", CSSM_GetError()->error,__LINE__);
      // we continue trying to close other stuff
      }
    datasources_ptr->DLDBHandle[0].DLHandle = 0;   // clear handle
    }

  ////////////////////////////////////////////////////////////////
  // Say goodbye to OCEP
  ////////////////////////////////////////////////////////////////
   status = CSSM_DL_DbClose(datasources_ptr->DLDBHandle[datasources_ptr->NumHandles - 1]);
   if (status != 0)
      {
      printf("Failed CSSM_DL_DbClose %d, line %d\n", CSSM_GetError()->error,__LINE__);
      // we continue trying to close other stuff
      }

   if (datasources_ptr->DLDBHandle[datasources_ptr->NumHandles - 1].DLHandle)
     {
     if ((status = CSSM_ModuleDetach(
        datasources_ptr->DLDBHandle[datasources_ptr->NumHandles - 1].DLHandle)) != 0)
      {
      printf("Failed CSSM_ModuleDetach: %d, line %d\n", CSSM_GetError()->error,__LINE__);
      // we continue trying to close other stuff
      }
    datasources_ptr->DLDBHandle[datasources_ptr->NumHandles - 1].DLHandle = 0;
    }

  ////////////////////////////////////////////////////////////////
  // Say goodbye to cryptographic service provider (CSP)
  ////////////////////////////////////////////////////////////////
   if (ibm_csp_handle)
     {
      if (status = CSSM_ModuleDetach(ibm_csp_handle) != 0)
       {
       printf("Failed CSSM_ModuleDetach: %d, line %d\n", CSSM_GetError()->error,__LINE__);
       // we continue trying to close other stuff
       }
     }

  ////////////////////////////////////////////////////////////////
  // Farewell PKITP
  ////////////////////////////////////////////////////////////////
  if (tphandle)
    {
    if (status = CSSM_ModuleDetach(tphandle) != 0)
      {
      printf("Failed CSSM_ModuleDetach: %d, line %d\n", CSSM_GetError()->error,__LINE__);
      // we continue trying to close other stuff
      }
    }

 return;
}

 /*****************************************************************
  *  name: buildCertGroup - read certificates from files, set up
  *        CSSM_CERTGROUP to reference input certificates
  *
  *  input: CSSM_CERTGROUP * -- addresses unintialized CSSM_CERTGROUP
  *         certFile - array of strings containing names of files that
  *             have DER encoded certificates to be verified by PKITP
  *         certCount - number of elements (strings) in certFile
  *
  *  output: returns CSSM_OK if all certificates read
  *          -  CSSM_CERTGROUP will have NumCerts set and CertList
```

```
 *             will be the address of array of certificates
 *          returns CSSM_FALSE if error reading a file
 ********************************************************************/

int buildCertGroup(CSSM_CERTGROUP * certGroupPtr,
                   char * certFile[]], uint32 certCount)
  {
  FILE      * inFile;
  CSSM_DATA * certArray = (CSSM_DATA *) calloc(certCount,sizeof(CSSM_DATA));
  uint32    i, certSize;

  certGroupPtr->NumCerts = certCount;
  certGroupPtr->CertList = certArray;

  for (i=0; i < certCount; i++)
   {
   inFile = fopen(certFile[i],"rb");
   if (!inFile)
     {
     printf("File %s could not be opened\n",certFile[i]);
     if (i > 1)          // if we've read any certs before this
       {
       certGroupPtr->NumCerts = i - 1;   // indicate how many read
       freeCertGroup(certGroupPtr);  // free alloc'd storage
       }
     return(CSSM_FAIL);
     }
  /* Find size of certificate file */
    fseek(inFile,0L,SEEK_END);
    certSize = ftell(inFile);
    rewind(inFile);

 /* Read in certificate data*/
    certArray[i].Length = certSize;
    certArray[i].Data   = (uint8 *)calloc(certSize, sizeof(char));
    fread(certArray[i].Data, 1, certSize, inFile);
    fclose(inFile);
    }
  return(CSSM_OK);
  }


 /*********************************************************************
  *  name: verifyCertGroup - call the Trust Policy (FINALLY)
  *
  *  purpose:  call CSSM_TP_PassThrough (PKITP) to verify certificate(s)
  *            call reportCertGroupVerify (internal routine to display
  *               results to stdout
  *            call CSSM_TP_PassThrough (PKITP) to free storage related to
  *               results
  *
  *  input:    CSSM_CERTGROUP containing number of and array of certificates
  *            CSSM_DL_DB_LIST containing CSSM handles for LDAP and OCEP
  *            CSSM_TP_HANDLE  CSSM handle for PKITP
  *
  *  output:   none
  *
  ********************************************************************/

  void verifyCertGroup(CSSM_CERTGROUP certgroup,
                       CSSM_DL_DB_LIST * datasources_ptr,
                       CSSM_TP_HANDLE tphandle)
  {
  ////////////////////////////////////////////////////////////////////
  //
  // While there are only 3 parameters on CSSM_TP_PassThrough call to PKITP:
  //  - the CSSM_TP_HANDLE,
```

```
   //  - the function code "TP_VERIFY_PASSTHROUGH" and
   //  - a pointer to the TP_VERIFY_EXTRA structure.
   // TP_VERIFY_EXTRA structure contains many parameters, including the address of
   // TP_INITIALPOLICY structure that can be used to override the default
   // policy settings and the address of TP_VERIFY_EXTRA which PKITP can use
   // to pass back more detailed results.
   ////////////////////////////////////////////////////////////////////
   TP_INITIALPOLICY initialPolicyPreferences;
   TP_EVIDENCE pkixEvidence;
   TP_VERIFY_EXTRA extraVerifyInfo;

   ////////////////////////////////////////////////////////////////////
   // The field initialPolicyMappingInhibit in TP_INITIALPOLICY is not used
   // by PKITP, therefore we do not set it.
   ////////////////////////////////////////////////////////////////////
   initialPolicyPreferences.NumberofPolicyIdentifiers = NUM_POLICIES;
   initialPolicyPreferences.PolicyIdentifiers = policydata;
   initialPolicyPreferences.initialExplicitPolicy = INITIALExplicitPolicy;
   initialPolicyPreferences.initialPolicyMappingInhibit = CSSM_FALSE;
   initialPolicyPreferences.useCRLs = USECRLS;

   ////////////////////////////////////////////////////////////////////
   // The following fields in TP_VERIFY_EXTRA are not used by PKITP.
   // CLHandle, PolicyIdentifiers and NumberofPolicyIdentifiers
   // (not to be confused with fields of same name in TP_INITIALPOLICY structure),
   // AnchorCerts and NumberofAnchorCerts.
   // Therefore we do not set these fields below.
   ////////////////////////////////////////////////////////////////////
   extraVerifyInfo.DBList                    = datasources_ptr;
   extraVerifyInfo.CSPHandle                 = ibm_csp_handle;
   extraVerifyInfo.VerificationAbortOn       = CSSM_TP_STOP_ON_POLICY;
   extraVerifyInfo.CertToBeVerified          = &certgroup;
   extraVerifyInfo.InitialPolicy             = &initialPolicyPreferences;
   extraVerifyInfo.Evidence                  = &pkixEvidence;
   extraVerifyInfo.ValidationTime            = time(0);


   (void*)CSSM_TP_PassThrough(tphandle,
                              0,
                              0,
                              0,
                              0,
                              TP_VERIFY_PASSTHROUGH,
                              (void *)&extraVerifyInfo);

   reportCertGroupVerify(extraVerifyInfo);
   (void*)CSSM_TP_PassThrough(tphandle,
                              0,
                              0,
                              0,
                              0,
                              TP_FREE_EVIDENCE,
                              (void *)&extraVerifyInfo);
   }

 //============================================================================
// function: reportCertGroupVerify
//============================================================================

void
reportCertGroupVerify
   (TP_VERIFY_EXTRA extraVerifyInfo)
{
   //-------------------------------------
   // report success or failure
   //-------------------------------------
   unsigned int reported_err = CSSM_GetError()->error;
```

```
   printf("TP_VERIFY_PASSTHROUGH : ");
   if (CSSM_FALSE == extraVerifyInfo.result)
   {
    printf("FAILED.  Error code: %d\n",reported_err);
   }
   else
   {
    printf("PASSED\n");
   }

   //--------------------------------------
   // report evidence
   //--------------------------------------
   printEvidence(extraVerifyInfo.Evidence);


}

void printEvidence(TP_EVIDENCE_PTR evidence_ptr)
{
 if (evidence_ptr == NULL) return;
 if (evidence_ptr->CompleteCertGroup)
   {
   printf("CompleteCertGroup was returned containing %d certificates at address %x\n",
     evidence_ptr->CompleteCertGroup->NumCerts,
     evidence_ptr->CompleteCertGroup->CertList);
   }
 else printf("CompleteCertGroup was NULL.\n");

 if (evidence_ptr->CRL)
   {
   printf("CRL was returned of %d bytes (decimal) at address %x\n",
     evidence_ptr->CRL->Length,
     evidence_ptr->CRL->Data);
   }
 else printf("CRL was NULL.\n");

  if (evidence_ptr->Cert)
   {
   printf("Cert (failed certificate) was returned of %d bytes (decimal) at address %x\n",
     evidence_ptr->Cert->Length,
     evidence_ptr->Cert->Data);
   }
 else printf("Cert was NULL.\n");
}

  /*********************************************************************
   *  name: freeCertGroup - Free certificate data storage              *
   *********************************************************************/

   void freeCertGroup(CSSM_CERTGROUP * certGroupPtr)
   {
   CSSM_DATA      * certArray = certGroupPtr->CertList;
   uint32           i;
   uint32           certCount = certGroupPtr->NumCerts;

   for (i=0; i <= certCount-1; i++)
    {
    free(certArray[i].Data);
    }
   free(certArray);
   return;
   }
```

End of Programming Interface information

Part 8. Appendixes

Appendix A. LDAP directory server requirements

PKI Services typically requires access to an LDAP directory server to store issued certificates and certificate revocation lists. The z/OS LDAP server is recommended but not required. You can use a non-Z/OS LDAP server if it can support the objectclasses and attributes PKI Services uses. These are listed in the following table:

Table 66. Table of LDAP objectclasses and attributes that PKI Services sets

End Entity or branch node?	Visible RDN attribute	Objectclasses used	Additional attributes set (other than visible RDN attribute)
Creating a branch node	C=	country	none
Creating a branch node	L=	locality	none
Creating a branch node	O=	organization	none
Creating a branch node	OU=	organizationalUnit	none
Creating a branch node	Any supported value other than the preceding	organizationalUnit and extensibleObject	ou — ou value from CreateOUValue in LDAP section of pkiserv.conf file
Creating a user End Entity	Any supported value	account, pkiUser, and extensibleObject	userCertificate and uid — hardcoded to ″NoUid″
Creating a CA End Entity	O=	organization and pkiCA	cACertificate
Creating a CA End Entity	OU=	organizationalUnit and pkiCA	cACertificate
Creating a CA End Entity	Any supported value other than O or OU	account, pkiCA, and extensibleObject	cACettificate and uid — hardcoded to ″NoUid″
User End Entity that already exists	Any supported value	pkiUser	userCertificate
CA End Entity that already exists	Any supported value	pkiCA	cACertificate

The R_PKIServ SAF callable service supports specifying the subject's DN through named fields in the CertPlist. The CGIs invoke the R_PKIServ SAF callable service. For more information, see *z/OS Security Server RACF Callable Services*. PKI Services supports the subject's DN fields, plus some additional ones: postal code, street, and mail. They are mapped to LDAP attributes as the following table indicates:

Table 67. Relationship of named fields to LDAP attributes and object identifiers

Named field	Visible RDN attribute	OID
CommonName	CN	2.5.4.3
Title	TITLE	2.5.4.12
OrgUnit	OU	2.5.4.11
Org	O	2.5.4.10
Locality	L	2.5.4.7
StateProv	ST	2.5.4.8
Country	C	2.5.4.6
n/a - (PKCS#10 only)	POSTALCODE	2.5.4.17

Table 67. Relationship of named fields to LDAP attributes and object identifiers (continued)

Named field	Visible RDN attribute	OID
n/a - (PKCS#10 only)	STREET	2.5.4.9
n/a - (PKCS#10 only)	MAIL	0.9.2342.19200300.100.1.3

Appendix B. Using a gskkyman key database for your certificate store

This appendix lists the steps the RACF programmer performs to use a gskkyman key database.

Steps for using a gskkyman key database for your certificate store

Perform the following steps to use a gskkyman key database for your server's certificate store:

Note: If the z/OS HTTP Server is installed and configured for SSL using gskkyman, you need to perform only steps 9, 10, 11, and 15.

1. From the UNIX shell, cd to /etc and enter /usr/lpp/gskssl/bin/gskkyman.

   _____

2. Choose option 1 to create a key database. Type in a name or let it default to key .kdb and enter a password you want to use. When asked ″work with the database now?″ enter 1 for yes.

   _____

3. Choose option 3 — Create new key pair and certificate request. Answer the prompts for file name, label, key size (1024 recommended), and subject name fields.

   Note: Common Name should be your server's symbolic IP address (for example, www.*yourcompany*.com).

   _____

4. Exit gskkyman when you are done.

   _____

5. From TSO, use the OGET command to put the certificate request in an MVS data set.

   Example:

   ```
   OGET '/etc/certreq.arm' certreq.arm
   ```

   _____

6. Use RACDCERT to read the request and generate the server certificate.

   Example:

   ```
   RACDCERT GENCERT(certreq.arm) ID(WEBSRV) SIGNWITH(CERTAUTH
   LABEL('Local PKI CA')) WITHLABEL('SSL Cert')
   ```

   _____

7. Export both the new server certificate and the CA certificate to MVS data sets, and OPUT these to HFS files.

   Example:

   ```
   RACDCERT EXPORT(LABEL('SSL Cert')) ID(WEBSRV) DSN(cert.arm)
   FORMAT(CERTB64)
   OPUT cacert.der '/var/pkiserv/cacert.der' BINARY
   ```

   _____

8. You can optionally delete both certificate TSO data sets (but not the HFS files).

9. In the UNIX shell, cd to /etc and invoke /usr/lpp/gskssl/bin/gskkyman.

10. Choose option 2 to open the key database (created earlier). Reply to the name and password prompts.

11. Choose option 6 to store a CA certificate and specify the '/var/pkiserv/cacert.der' file.

12. When asked to ″exit gskkyman?″ Enter 0 for No.

13. Choose option 4 to receive a certificate issued for your request and specify the '/etc/cert.arm' file. Again enter 0 when asked to ″exit gskkyman?″

14. Choose option 11 to store encrypted database password.

15. Exit gskkyman.

16. You can optionally remove the /etc/cert.arm file.

Appendix C. Accessibility

Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use software products successfully. The major accessibility features in z/OS enable users to:

- Use assistive technologies such as screen-readers and screen magnifier software
- Operate specific or equivalent features using only the keyboard
- Customize display attributes such as color, contrast, and font size

Using assistive technologies

Assistive technology products, such as screen-readers, function with the user interfaces found in z/OS. Consult the assistive technology documentation for specific information when using it to access z/OS interfaces.

Keyboard navigation of the user interface

Users can access z/OS user interfaces using TSO/E or ISPF. Refer to *z/OS TSO/E Primer*, *z/OS TSO/E User's Guide*, and *z/OS ISPF User's Guide Volume I* for information about accessing TSO/E and ISPF interfaces. These guides describe how to use TSO/E and ISPF, including the use of keyboard shortcuts or function keys (PF keys). Each guide includes the default settings for the PF keys and explains how to modify their functions.

Notices

This information was developed for products and services offered in the USA.

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 described 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
USA

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 transactions, therefore, this statement may not apply to you.

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.

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
2455 South Road
Poughkeepsie, NY 12601-5400
USA

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 information 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.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Programming interface information

This book primarily documents information that is NOT intended to be used as Programming Interfaces of PKI Services.

This book also documents intended Programming Interfaces that allow the customer to write programs to obtain the services of PKI Services. This information is identified where it occurs, either by an introductory statement to a chapter or section or by the following marking:

Programming Interface information

End of Programming Interface information

Trademarks

The following terms are trademarks of the IBM Corporation in the United States, or other countries, or both:

AIX
BookManager
DB2
DFS
IBM
IBMLink
Library Reader
MVS
OS/390
RACF
Redbooks
Resource Link
S/390
SecureWay

TalkLink
z/OS
zSeries

Intel is a trademark of Intel Corporation in the United States, other countries, or both.

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.

Tivoli is a trademark of International Business Machines Corporation or Tivoli Systems, Inc. in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, and service names may be trademarks or service marks of others.

Bibliography

The following lists titles and book numbers of books referenced in this publication.

- *z/OS DFSMS Access Method Services for Catalogs*, SC26-7394
- *z/OS Distributed File Service DFS Administration*, SC24-5915
- *z/OS HTTP Server Planning, Installing, and Using*, SC34-4826
- *z/OS ICSF Administrator's Guide*, SA22-7521
- *z/OS ICSF Application Programmer's Guide*, SA22-7522
- *z/OS ICSF System Programmer's Guide*, SA22-7520
- *z/OS SecureWay Security Server LDAP Client Programming*, SC24-5924
- *z/OS Security Server LDAP Server Administration and Use*, SC24-5923
- *z/OS MVS Programming: Authorized Assembler Services Reference SET-WTO*, SA22-7612
- *z/OS MVS Programming: Sysplex Services Reference*, SA22-7618
- *z/OS SecureWay Security Server Open Cryptographic Enhanced Plug-ins Application Programming*, SC24-5925
- *z/OS Open Cryptographic Services Facility Application Programming*, SC24-5899
- *z/OS Security Server RACF Callable Services*, SA22-7691
- *z/OS Security Server RACF Command Language Reference*, SA22-7687
- *z/OS Security Server RACF Security Administrator's Guide*, SA22-7683
- *z/OS TSO/E REXX Reference*, SA22-7790
- *z/OS UNIX System Services Command Reference*, SA22-7802
- *z/OS UNIX System Services Planning*, GA22-7800

Index

Special Characters

Numerics

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Z

Readers' Comments — We'd Like to Hear from You

z/OS
Security Server PKI Services
Guide and Reference

Publication No. SA22-7693-00

Overall, how satisfied are you with the information in this book?

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Overall satisfaction	☐	☐	☐	☐	☐

How satisfied are you that the information in this book is:

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Accurate	☐	☐	☐	☐	☐
Complete	☐	☐	☐	☐	☐
Easy to find	☐	☐	☐	☐	☐
Easy to understand	☐	☐	☐	☐	☐
Well organized	☐	☐	☐	☐	☐
Applicable to your tasks	☐	☐	☐	☐	☐

Please tell us how we can improve this book:

Thank you for your responses. May we contact you? ☐ Yes ☐ No

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.

Name _____ Address _____

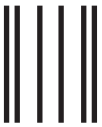
Company or Organization _____

Phone No. _____

Readers' Comments — We'd Like to Hear from You
SA22-7693-00

Cut or Fold
Along Line

Fold and Tape | **Please do not staple** | Fold and Tape



NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST-CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

IBM Corporation
Department 55JA, Mail Station P384
2455 South Road
Poughkeepsie, NY
12601-5400

Fold and Tape | **Please do not staple** | Fold and Tape

Cut or Fold
Along Line

Program Number: 5694-A01

Printed in the United States of America
on recycled paper containing 10%
recovered post-consumer fiber.

SA22-7693-00