

RACF for PKI Services

Theo Antoff, antoff IT inc, April 2005

RACF FOR PKI SERVICES	1
1 SETTING UP RACF ENVIRONMENT FOR PKI PREREQS	2
1.1. <i>z/OS UNIX level security</i>	2
1.2. <i>RACF for IBM HTTP server</i>	2
Started Task Userid	2
Profile in class STARTED	3
Access to profiles in class CSFSERV	3
1.3. <i>RACF for OCSF and OCEP</i>	3
1.4 <i>RACF for LDAP</i>	3
Started Task Userid	3
Profile in class STARTED	3
Access to DB2 from TSO and BATCH	4
Daemon and Server control	4
Access to OCSF	4
Access to profiles in class CSFSERV	4
1.5. <i>RACF for ICSF</i>	4
Started task userid	4
ICSF datasets	4
Profile in class STARTED	4
Profiles in class CSFKEYS	4
Access to profiles in class CSFSERV	5
2 SETTING UP RACF ENVIRONMENT FOR PKI SERVICES	5
2.1. <i>RACF groups for PKI Services</i>	5
PKI administrator group	5
PKI Started Task Userid default group	5
2.2. <i>RACF userids for PKI Services</i>	6
PKI started task userid	6
Surrogate userid	6
2.3. <i>Dataset profiles for PKI</i>	6
2.4. <i>RACF RACDCERT command</i>	7
2.5. <i>Certificate Authority (CA) Certificate</i>	9
Create the CA certificate	9
Make the CA Certificate HIGHTRUST	9
Backup to a dataset	10
Save the CA certificate to a dataset for import to a UNIX file	10
Create PKI Services keyring	10
2.6. <i>Web server SSL certificate</i>	11
Create the Web server certificate	11
Create the Web server key ring	11
2.7. <i>Display certificates</i>	12
CHECKCERT keyword	12
LIST keyword	12
RLIST command	14
2.8. <i>Display Key rings</i>	15
LISTRING keyword	15
RLIST command	16
2.9 <i>Daemon and server control for PKI userid and Surrogate userid</i>	17
2.10 <i>Allow PKI userid to act as CA</i>	17
2.11 <i>Allow Web server to access its own key ring</i>	18
2.12 <i>Allow Web server userid to switch identity to Surrogate userid</i>	18
2.13 <i>Profile for PKI Services procedure in class STARTED</i>	18
2.14 <i>Allow access for SUPKI to OCSF</i>	18
2.15 <i>ICSF</i>	18

2.16 Protect certificate functions	19
2.17.Run RACF commands	20
3 RACF ADMINISTRATION FOR PKI SERVICES	23
3.1 Creating a helpdesk function.....	23
3.2 Administering certificates with HostIdMapping extension	24
3.3 Display certificates using utilities iclview and vosview	24
3.4 Establishing PKI Services as an intermediate certificate authority.....	26
3.5 Renewing your PKI Services CA certificate.....	27
3.6 Controlling applications that call R_PKIServ	27
R_PKIServ end-user functions.....	27
R_PKIServ administrative functions.....	29
3.7. Using encrypted passwords for LDAP servers.....	29
Profiles in class LDAPBIND	30
Profile in Class FACILITY	30
List PROXY Segment	31

This article intends to help RACF administrators or RACF systems programmers to define all necessary RACF controls to enable installation of PKI Services for z/OS V1.4.

Almost all of the RACF work necessary to be performed to have PKI Services up and running is in the REXX exec IKYSETUP residing in SYS1.SAMPLIB and in chapter 25 of **Security Server PKI Services Guide and Reference, SA22-7693**.

Running this exec will allow RACF people or people assigned the task to install PKI Services not familiar even with the RACDCERT command, to run it, create a RACF environment and then proceed with the httpd.conf and pkiserv.conf customizations.

Although IKYSETUP is a valuable tool, we undertook a different approach in this article - first explain how to create RACF environment for PKI Services when undoubtedly a lot of naming standards, procedures and RACF design style are already in place, and then run a job through which the new product seamlessly fits into your RACF database.

1 Setting up RACF environment for PKI prereqs

The implementation of PKI Services requires that z/OS UNIX System Services (USS) is active. Therefore we expect that most of the USS RACF controls are in place as per the content of member BPXISEC1 in SYS1.SAMPLIB.

The prerequisite products for PKI Services are IBM HTTP Server or Web server, Lightweight Directory Access Protocol (LDAP) server, Open Cryptographic Services Facility (OCSF) and Open Cryptographic Enhanced Plug-ins (OCEP).

Integrated Cryptographic Services Facility (ICSF) and z/OS Communications Server's sendmail utility are optional.

1.1. z/OS UNIX level security

We recommend you have z/OS level of UNIX security on your system. More details about the two possible levels of security (UNIX level of security and z/OS UNIX level of security) can be found at <http://www.antoff-it.com/unix-security.pdf>.

1.2. RACF for IBM HTTP server

Everywhere in this article we will use the term Web server instead of z/OS HTTP server.

Started Task Userid

The command to create a started userid for your web server is:

```
ADDUSER SUWEB DFLTGRP(STC) OMVS(UID(0) HOME('/usr/lpp/internet') +
```

PROGRAM('/bin/sh')

Profile in class STARTED

The command to create a profile in class STARTED for your web server procedure is:

RDEF STARTED WEB STDATA(USER(SUWEB) GROUP(STC))**

The commands to control your web server started task userid are:

**PERMIT BPX.DAEMON CLASS(FACILITY) ID(SUWEB) ACCESS(READ)
PERMIT BPX.SERVER CLASS(FACILITY) ID(SUWEB) ACCESS(READ)**

Note: If you wish to define your web server userid with a nonzero UID and according to your naming standard for STUs, do not forget to give this userid appropriate permissions to directories and files:

```
SETFACL -m u:SUWEB:r-x /usr/lpp/internet/sbin/httpd_V5R3M0
SETFACL -m u:SUWEB:r-x /web/pki1/httpd.conf
SETFACL -m u:SUWEB:rw- /web/pki1/logs
SETFACL -m d:u:SUWEB:rw- /web/pki1/logs
SETFACL -m f:u:SUWEB:rw- /web/pki1/logs
SETFACL -m u:SUWEB:rw- /web/pki1/httpd-pid
```

The Web server's nonzero user ID must be given read/execute access to the security DLLs, read/write access to the key database file, and read access to the stash file.

Access to profiles in class CSFSERV

This access is needed, only if ICSF is active.

PE CSF* CL(CSFSERV) ID(SUWEB) ACC(R)

1.3. RACF for OCSF and OCEP

The use of Open Cryptographic Services Facility (OCSF) and Open Cryptographic Enhanced Plug-ins (OCEP) is controlled by fixed-name profiles in class FACILITY:

CDS.CSSM Authorizes access to OCSF
CDS.CSSM.CRYPTO Authorizes access to Cryptographic Service Provider
CDS.CSSM.DATALIB Authorizes access to Data Library (DL) Service Provider

It is likely that all these resources will have the same access list, so only one profile may be defined:

RDEF FACILITY CDS. UACC(NONE) OWNER(SECADM)**

1.4 RACF for LDAP

The RACF setup for the LDAP server consists of the following commands.

Started Task Userid

Create started task userid with:

**ADDUSER SULDAP DFLTGRP(STC) NAME(LDAP Started Task User') +
OMVS(AUTOUID HOME(') PROG('/bin/sh'))**

Profile in class STARTED

Create profile with:

RDEF STARTED LDAP.** STDATA(USER(SULDAP) GROUP(STC))

Access to DB2 from TSO and BATCH

Permit to appropriate subsystem in class DSNR with:

PE DB2subsystem.BATCH CL(DSNR) ID(SULDAP) ACC(R)

Daemon and Server control

Permit to these profiles in class FACILITY:

PE BPX.DAEMON CL(FACILITY) ID(SULDAP) ACC(R)
PE BPX.SERVER CL(FACILITY) ID(SULDAP) ACC(U)

Access to OCSF

Permit to profile in class FACILITY:

PE CDS.** CL(FACILITY) ID(SULDAP) ACC(R)

Access to profiles in class CSFSERV

This access is needed only if ICSF is active:.

PE CSF* CL(CSFSERV) ID(SULDAP) ACC(R)

1.5.RACF for ICSF

Using Integrated Cryptographic Services Facility (ICSF) is **recommended**, but not a prerequisite for PKI Services. You can install and configure ICSF before setting up PKI Services or later.

Started task userid

Define a started task userid to run the ICSF procedure with:

ADDUSER SUICSF DFLTGRP(STC) NAME('ICSF Started task user') NOPASSWORD

ICSF datasets

The ICSF datasets (CKDS and PKDS) contain Cryptographic Keys (CKDS) and Public Keys (PKDS):

AG ICSF SUPGROUP(DATA) OWNER(DATA)
AD 'ICSF.** OWNER(SECADM) UACC(NONE)

Profile in class STARTED

Define a profile for the ICSF procedure:

RDEF STARTED ICSF.** STDATA(USER(SUICSF) GROUP(STC))

Profiles in class CSFKEYS

Profiles in this class protect labels for keys in the format:

RDEF CSFKEYS label UACC(NONE) AUDIT(ALL)
PE label CL(CSFKEYS) ID(SECADM) ACC(R)

You may need to create one or more jobrole groups in addition to SECADM for key management. Also you may decide to create a common generic profile for all key labels:

```
RDEF CSFKEYS ** UACC(NONE) OWNER(SECADM) AUDIT(ALL)
```

and permit one or several jobrole groups to it.

Access to profiles in class CSFSERV

Profiles in this class protect cryptographic services in fixed-name format:

```
RDEF CSFSERV service-name UACC(NONE) OWNER(SECADM) AUDIT(ALL)
PE service-name CL(CSFSERV) ID(SECADM) ACC(R)
```

Similarly to class CSFKEYS you may define a common generic profile

```
RDEF CSFSERV ** UACC(NONE) OWNER(SECADM) AUDIT(ALL)
or
RDEF CSFSERV CSF* UACC(NONE) OWNER(SECADM) AUDIT(ALL)
```

For all profile names in class CSFSERV refer to chapter 3 of the *ICSF Administrator's Guide SA22-7521*.

If you have not activated CSFKEYS and CSFSERV, issue the commands:

```
SETR CLASSACT(CSFKEYS CSFSERV)
SETR RACLIST(CSFKEYS CSFSERV)
SETR RACLIST(CSFKEYS CSFSERV) REFR
```

2 Setting up RACF environment for PKI services

2.1. RACF groups for PKI Services

PKI administrator group

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. We recommend appointing as PKI administrators one or more of your RACF administrators after some training mostly on the usage of the RACDCERT command.

The obvious candidate for a group to administer PKI Services is the existing group of RACF administrators. However, you may have already a unit managing certificates outside the RACF security team. In this case it is worth considering training this group to use PKI Services on z/OS.

Also, all RACF userids and groups to be used in PKI Services must have OMVS segments. Here is the command to add a specific job role group for PKI administrators:

```
AG PKIADM SUPGROUP(JOBROLE) OWNER(JOBROLE) OMVS(AUTOGID)
CO (userid1 userid2 etc) GROUP(PKIADM)
```

If you need to assign an OMVS segment to your existing RACF admin group, issue the following command:

```
ALG SECADM GID(AUTOGID)
```

PKI Started Task Userid default group

If you need to assign an OMVS segment to your existing RACF default group for started task userids, issue the following command:

```
ALG STC GID(AUTOGID)
```

2.2. RACF userids for PKI Services

PKI started task userid

The command to create the userid who runs the PKI procedure (see the PKI procedure in SYS1.PROCLIB) is:

```
AU SUPKI NAME('PKI SRVS DAEMON') DFLTGRP(STC) OWNER(STC) NOPASSWORD +  
OMVS(AUTOUID HOME('/web/pki/') PROG('/bin/sh') ASSIZE(256000000) THREADS(512))
```

The ASSIZEMAX and THREADSMAX values are recommended, but you have to increase them if they are not enough for your workload. The full list of keywords specifying various limits in the OMVS segment follows:

ASSIZEMAX	Maximum address space size
CPUTIMEMAX	Maximum CPU time
FILEPROCMA	Maximum number of files per process
MMAPAREAMAX	Maximum memory map size
PROCUSERMAX	Maximum number of processes per UID
THREADSMAX	Maximum number of threads per process

Once you have set individual user limits for users who require higher resource limits, you should consider removing their superuser authority. You should also reevaluate your installation's BPXPRMxx limits and consider reducing these limits.

For ranges and defaults for the above items, see chapter 9 in the ***z/OS MVS Initialization and Tuning Reference SA22-7592-03***.

Surrogate userid

A surrogate userid is the identity assigned to client processes when they are requesting PKI services. A surrogate userid is required for external clients. For simplicity we recommend that you use surrogate userid's for internal clients as well, rather than allowing them to access PKI Services under their own identities.

Our chosen name for surrogate userid is PKISRV and the command to create it, is

```
AU PKISRV NAME ('PKI SURROGATE') DFLTGRP(MISC) OWNER(MISC) +  
OMVS(AUTOUID HOME('/') PROG('/bin/sh') ASSIZEMAX(256000000) +  
THREADSMAX(512)) NOPASSWORD RESTRICTED
```

We recommend this userid to be RESTRICTED with the consequences that this user must be explicitly permitted with READ to several resources having UACC(READ) or ID(*) with access READ.

The most obvious are: profile ** in class PROGRAM, a few profiles starting with IRR.DIGTCERT in class FACILITY, profile ** in class APPL.

2.3. Dataset profiles for PKI

We recommend that datasets needed for PKI data be named

yourprefix.PKI

Alternatively, if you do not use prefixing, use HLQ of PKI. To define a RACF profile for your PKI datasets use:

```
AD 'PKI.**' OWNER (PKIADM) AUDIT(ALL)
```

Note: In the command we assume that you have turned EGN on and also we omitted UACC, because we recommend UACC(NONE) which is the default. We believe that by using

AUDIT(ALL) your auditors will be happy if SMF records for SUCCESS READ and higher are made available.

The recommended permissions are:

PE 'PKI.' ID(MVSMNT) ACC(ALTER),**

because usually group MVSMNT is required to create datasets (VSAM in the case of PKI Services). Do not forget to place your group with ALTER if you plan to create these datasets and you do not have the OPERATIONS attribute.

To allow your PKI administrators access to PKI datasets, issue:

PE 'PKI.' ID(PKIADM) ACC(CONTROL)**

Also, the started task userid for PKI Services (SUPKI) must be able to write into the PKI datasets:

PE 'PKI.' ID(SUPKI) ACC(CONTROL)**

2.4. RACF RACDCERT command

The Certificate Authority, commonly known as CA, acts as a trusted 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. You can use RACF to create, register, store, and administer digital certificates and their associated private keys, and build certificate requests that can be sent to a certificate authority for signing. You can also use RACF to manage key rings of stored digital certificates. Digital certificates and key rings are managed in RACF primarily by using the RACDCERT command.

RACF allows you to manage three types of digital certificates:

CA certificate A certificate that is associated with a certificate authority and is used to verify

signatures in other certificates.

User certificate A certificate that is associated with a RACF user ID and is used to authenticate the user's identity.

Site certificate A certificate that is associated with a server, or network entity other than a user or certificate authority.

Your organization may already have rules in place for the creation of certificates. If not, then such rules should be discussed and established. An essential part of any certificate is the so-called Distinguished Name. It consists of the components listed in Table 1 below:

Table 1. Components of Distinguished Name

Common name	(CN)*
Title	(T)*
Organization Unit	(OU)
Organization	(O)
Locality	(L)
Street	(STREET)*
State/Province	(SP)
Mail	(MAIL)*
Postal Code	(POSTALCODE)*
Country	(C)

The abbreviations above are used as subkeywords for defining certificates. The fields with * are not typical for CA certificates.

Note: The components MAIL, POSTALCODE and STREET are valid for PKI services scripts only. They cannot be used as subkeywords in the RACDCERT command.

The RACDCERT command allows a RACF-defined userid to use a digital certificate as identification. The certificate must be one of the supported formats contained in an MVS data set. For more information on these formats refer to **z/OS Security Server RACF Command Language Reference**.

RACDCERT is used to create, store and maintain digital certificates in RACF, and should be used for all maintenance of the DIGTCERT class profiles and related USER profile fields. For those curious to find out if the command LISTUSER would provide information on digital certificates associated with the listed userid, we would like to advise that LISTUSER will not provide such information. However, USER-related record type 0207 (User Certificate Name Record) provides userid, certificate name and certificate label. For more details see Chapter 10. RACF Database Unload Utility (IRRDBU00) from **z/OS V1R4.0 Security Server RACF Macros and Interfaces**.

The RACDCERT command is your primary administrative tool for managing digital certificates using RACF. Granular authorities for use of the RACDCERT command by users not having SYSTEM SPECIAL are controlled through profiles in the FACILITY class of the type IRR.DIGTCERT.function. It is outside the scope of this article to show the best set of profiles, UACCs and access lists for these profiles. One possible approach is suggested in section "Certificate Administration" as part of our CLIST to install PKI Services on page 21.

The RACDCERT command is used to manage resources in the following classes:

- DIGTCERT** Profiles in the DIGTCERT class contain information about digital certificates, as well as the certificate itself. The userid associated with the certificate can be found in the APPLDATA field of the profile.
- DIGTRING** Profiles in the DIGTRING class contain information about key rings and certificates that are part of each key ring. Key rings are named collections of the personal, site and CA certificates associated with a specific user.
- DIGTNMAP** Profiles in the DIGTNMAP class contain information about certificate name filters.

Note: Profiles in the DIGTCERT, DIGTRING and DIGTNMAP classes are maintained automatically through RACDCERT command processing. You cannot administer profiles in these classes using the RDEFINE, RALTER, and RDELETE commands.

Profile names in the DIGTCERT class are in the form:

serial-number.issuer's-distinguished-name

For example, if: **41D87A3B05D** is the serial number and, **OU=VeriSign Class1.O=VeriSign.L=Internet**, is the distinguished name, then the profile name for this DIGTCERT profile would be:

41D87A3B05.OU=VeriSignClass1.O=VeriSign.L=Internet

As can be seen, any characters that would not be valid in a profile name, such as a blank, will be replaced with X'4A' (¢).

Profile names in the DIGTRING class are in the form

owning-userid.key-ring-name

You may use the SEARCH command or the RLIST command to find all entries defined in the classes, for example:

```
SR CLASS(DIGTCERT)
SR CLASS(DIGTRING)
RLIST DIGTCERT *
RLIST DIGTRING *
```

The output from RLIST has almost 40 times more lines due to the usual information about OWNER, UACC, installation data, application data, your access, date of creation, access list (not applicable here) etc.

2.5. Certificate Authority (CA) Certificate

Create the CA certificate

The command to create this most important certificate is:

```
RACDCERT GENCERT CERTAUTH SUBJECTSDN(OU('PKI Dept') +
O('antoff IT') C('US')) WITHLABEL('PKIServicesCA') NOTAFTER(DATE(2020/01/01))
```

where:

GENCERT creates a digital certificate and potentially a public or private key pair.

CERTAUTH relates to certificate of a Certificate Authority (CA).

SUBJECTSDN specifies the subject's distinguished name, which consists of the components shown in Table 1. In our case organization unit is OU=PKI Dept, organization is O=antoff IT, country is C=US.

WITHLABEL specifies the label assigned to this certificate. If specified, this must be unique to the user ID with which the certificate is associated.

Note: The label-name is stripped of leading and trailing blanks. If a single quotation mark is intended to be part of the label-name, you must use two single quotation marks together for each single quotation mark within the string, and the entire string must then be enclosed within single quotation marks. We associated this CA certificate with webserver PKI1.

NOTAFTER specifies the local date and time after which the certificate is no longer valid.

Note: It is recommended that you set up a job (with CAPS OFF) to use the RACDCERT command when a long string of keywords and their values are needed.

Example 1. Job to create CA certificate

```
//your job card
//STEP1 EXEC PGM=IKJEFT01
//SYSLBS DD DSN=SYS1.BROADCAST,DISP=SHR
//SYSTSPRT DD SYSOUT=*
//SYSTSIN DD *
RACDCERT GENCERT CERTAUTH SUBJECTSDN(OU('PKI Dept') +
O('antoff IT') C('US')) WITHLABEL('PKIServicesCA') +
NOTAFTER(DATE(2020/01/01))
```

Make the CA Certificate HIGHTRUST

The subkeywords TRUST | NOTRUST | HIGHTRUST indicate whether the status of the certificate is trusted, not trusted, or highly trusted. Whether the certificate is not trusted or trusted depends on whether or not the certificate is valid and whether the private key has been compromised or not. Since highly trusted certificates are by definition trusted certificates, any certificate usage that was enabled by marking the certificate trusted will also

be enabled by marking the certificate highly trusted. However, only CA certificates can be highly trusted. The keyword ALTER is used to change the status to HIGHTRUST:

```
RACDCERT CERTAUTH ALTER(LABEL('PKIServicesCA')) HIGHTRUST
```

Note: The value of the LABEL keyword is case-sensitive (ours is in upper case).

Backup to a dataset

It is absolutely critical to backup the certificate into a dataset with the PKCS12DER Format. The keyword for it is EXPORT:

```
RACDCERT CERTAUTH EXPORT(LABEL('PKIServicesCA')) +  
DSN('PKI.CAPKI.BACKUP.P12BIN') +  
FORMAT(PKCS12DER) PASSWORD('xxxxxxx')
```

Note: Be careful when you issue this command – the password is written in clear, so you have to find a secure place to store/remember the password for future use. If lost, this backup becomes useless. Also the password value is case-sensitive.

Save the CA certificate to a dataset for import to a UNIX file

The CA certificate should be placed into an MVS dataset in the DER format and then copied to an HFS file:

```
RACDCERT CERTAUTH EXPORT(LABEL('PKIServicesCA')) +  
DSN('PKI.CAPKI.DERBIN') FORMAT(CERTDER)  
chown SUPKI /var/pkiserv  
oput 'pki.capki.derbin' '/var/pkiserv/pkiserv/cacert.der' binary  
chmod 750 /var/pkiserv/cacert.der  
chown supki /var/pkiserv/*
```

Create PKI Services keyring

The RACDCERT command to create a key ring for the started task userid (daemon) running PKI Services is:

```
RACDCERT ADDRING(PKIRING) ID(SUPKI),
```

where

ADDRING creates a key ring. Only users can have a key ring. Key ring names become names of RACF profiles in the DIGTRING class, and can contain only characters that are allowed in RACF profile names. Although asterisks are allowed in ring names, a single asterisk is not allowed. For a CA certificate the user who owns the ring is the PKI daemon. Lower case characters are permitted. A key ring name can be up to 237 characters in length. Since only user IDs can have key rings, neither CERTAUTH nor SITE can be specified with ADDRING.

After creating the key ring, the certificate should be connected to the key ring. The command is:

```
RACDCERT ID(SUPKI) +  
CONNECT(CERTAUTH LABEL('PKIServicesCA')) +  
RING(PKIRING) USAGE(PERSONAL) DEFAULT),
```

where,

ID(userid) indicates that the certificate being added to the key ring is a user certificate, and userid is the user ID that is associated with this certificate. If the ID keyword is not specified, it defaults to the value specified or the default value on the RACDCERT command.

- CONNECT** specifies that a digital certificate is being added to a key ring.
- CERTAUTH** indicates that the certificate being added to the key ring is a CA certificate.
- USAGE** allows the altering of the trust policy within the confines of a specific key ring. For example, a CERTAUTH certificate connected with USAGE(PERSONAL) can be used to demote a certificate-authority certificate in order to insure that it is not used as a certificate authority in this ring. It can be used as a user certificate if a private key is present. However, typically one would not be present. Consequently, connecting a CERTAUTH certificate as USAGE(PERSONAL) is a way of marking it NOTRUST for this key ring only. Also, a user certificate connected with USAGE(CERTAUTH) can be used to promote an ordinary user certificate to a CA certificate. It can then be used to authenticate user certificates for this key ring only.
- DEFAULT** specifies that the certificate is the default certificate for the ring. Only one certificate within the key ring can be the default certificate. If a default certificate already exists, its DEFAULT status is removed, and the specified certificate becomes the default certificate. If you want the specified certificate to be the default, DEFAULT must be explicitly specified.

Note: All subkeywords belong to CONNECT, i.e. in the RACDCERT command the left bracket is immediately after CONNECT and the right bracket is after DEFAULT.

2.6. Web server SSL certificate

Once we have created our CA certificate we can start issuing other types of certificates, e.g. an SSL certificate for our web server started task userid needed to process handshakes with upcoming client certificates belonging to external or internal users.

Create the Web server certificate

The command to create the Web server certificate is:

```
RACDCERT GENCERT ID(SUWEB) +
SIGNWITH(CERTAUTH LABEL('PKIServicesCA')) +
SUBJECTSDN(CN('antoff.it.com')) +
O('antoff IT') L('DOVER') SP('DELAWARE) C('US')) WITHLABEL('SSL PKI') +
NOTAFTER(2020/01/01))
```

where,

SIGNWITH(CERTAUTH(LABEL('label-name'))) specifies the certificate with a private key that is signing the certificate. In our case it is our CA certificate. If SIGNWITH is not specified, the default is to sign the certificate with the private key of the certificate that is being generated.

Create the Web server key ring

Similarly to the CA, we have to create a key ring for the certificates of our webserver with the command:

```
RACDCERT ADDRING(SSLRING) ID(SUWEB)
```

We now connect the CA certificate to the ring belonging to the web server:

```
RACDCERT ID(SUWEB) CONNECT(CERTAUTH LABEL('PKIServicesCA') +
RING(SSLRING))
```

Note: We did not specify USAGE, because the default value of USAGE is same as in the added certificate, i.e. we preserved USAGE(CERTAUTH). We also omitted DEFAULT, because we did not want to make the CA certificate to be the DEFAULT in this key ring.

We connect the web server certificate to the web server ring with the following command:

```
RACDCERT ID(SUWEB) CONNECT(ID(SUWEB) LABEL('SSL PKI') RING(SSLRING) +
USAGE(PERSONAL) DEFAULT)
```

2.7.Display certificates

CHECKCERT keyword

You can use the RACDCERT command with keyword CHECKCERT to display information for any certificate. You can issue the command from ISPF option 6

```
RACDCERT CHECKCERT('PKI.CAPKI.DERBIN')
```

where PKI.CAPKI.DERBIN is the dataset to which we saved the CA certificate in DER format or from ISPF option 3.4, typing it in front of the corresponding dat set:

```
racdcert checkcert(/) PKI.CAPKI.DERBIN
```

with the following output:

Example 2. Output from RACDCERT CHECKCERT for CA certificate

```
Digital certificate information for CERTAUTH:
Label: PKIServicesCA
Certificate ID: 2QiJmZmDhZmjgcnC1EDJ4+LWQNfiycVA19LJ8UBA
Status: HIGHTRUST
Start Date: 2003/04/30 00:00:00
End Date: 2020/01/01 23:59:59
Serial Number:
>00<
Issuer's Name:
>OU=PKI Dept.0=PKIServicesCA.C=US<
Subject's Name:
>OU=PKI Dept.0=PKIServicesCA.C=US<
Key Usage: CERTSIGN
Private Key Type: ICSF
Private Key Size: 1024
```

Note: You can use the CHECKCERT keyword for any certificate saved in a dataset in any format.

LIST keyword

You can use the RACDCERT command with keyword LIST from ISPF option 6:

```
RACDCERT CERTAUTH LIST(LABEL('PKIServicesCA'))
```

Note: The keyword CERTAUTH must be used when listing a CA certificate. The value of LABEL is case sensitive. Also CERTAUTH may be placed after list(label(' ')).

The output from the above command is shown in the Example below:

Example 3. Output from RACDCERT LIST for CA certificate

```
Digital certificate information for CERTAUTH:
Label: PKIServicesCA
Certificate ID: 2QiJmZmDhZmjgcnC1EDJ4+LWQNfiycVA19LJ8UBA
Status: HIGHTRUST
```

```
Start Date: 2003/04/30 00:00:00
End Date: 2020/01/01 23:59:59
Serial Number:
>00<
Issuer's Name:
>OU=PKI Dept.0=PKIServicesCA.C=US<
Subject's Name:
>OU=PKI Dept.0=PKIServicesCA.C=US<
Key Usage: CERTSIGN
Private Key Type: ICSF
Private Key Size: 1024
Ring Associations:
Ring Owner: SUPKI
Ring:
>PKIRING<
Ring Owner: SUWEB
Ring:
>SSLRING<
```

It is important to note the following, when diagnosing errors:

- The first line must indicate that this is a CERTAUTH certificate.
- The Private Key Type and Size must be present.
- If the Serial Number is not equal to 00, this indicates that the certificate has been renewed or was issued by another certificate authority.

Note, that the only difference in the outputs from CHCHECKCERT and LIST is that the latter adds information for ring associations. In this information, you should ensure that one of the associations listed has the PKI Services daemon user ID as the ring owner and that the ring name matches your CA key ring name.

The command to display the web server certificate is:

```
RACDCERT ID(SUWEB) LIST(LABEL('SSL PKI'))
```

Note: The keyword ID() must be used when listing a user certificate. Also ID() may be placed after list(label(' ')). Don't forget the value of LABEL is case-sensitive !

The output from this command is shown below:

Example 4. Listing certificate for web server

```
Digital certificate information for user SUWEB:
Label: SSL PKI
Certificate ID: 2Qbmxcli4+Ti4tNA19LJ8UBA
Status: TRUST
Start Date: 2003/04/30 00:00:00
End Date: 2020/01/01 23:59:59
Serial Number:
>00<
Issuer's Name:
>OU=PKI Dept.0=PKIServicesCA.C=US<
Subject's Name:
>CN=antoff.it.com.0=antoff It.OU=Web server.C=US<
Private Key Type: Non-ICSF
Private Key Size: 1024
Ring Associations:
Ring Owner: SUWEB
Ring:
>SSLRING<
```

RLIST command

You can use the RACF RLIST command against the profile representing the CA certificate in class DIGTCERT:

RL DIGTCERT 00.OU=PKI⚡Dept.O=antoff⚡IT.C=US ALL

with the following output:

Example 5. Output from RLIST command

```
CLASS      NAME
-----
DIGTCERT  00.OU=PKI⚡Dept.O=antoff⚡IT.C=US

LEVEL OWNER   UNIVERSAL ACCESS YOUR ACCESS WARNING
-----
00    ANTOFF  ALTER                ALTER        NO

INSTALLATION DATA
-----
NONE

APPLICATION DATA
-----
irrcerta

SECLEVEL
-----
NO SECLEVEL

CATEGORIES
-----
NO CATEGORIES

SECLABEL
-----
NO SECLABEL

AUDITING
-----
FAILURES(READ)

GLOBALAUDIT
-----
NONE

NOTIFY
-----
NO USER TO BE NOTIFIED

CREATION DATE LAST REFERENCE DATE LAST CHANGE DATE
(DAY) (YEAR) (DAY) (YEAR) (DAY) (YEAR)
-----
120  03    120  03                120  03
ALTER COUNT CONTROL COUNT UPDATE COUNT READ COUNT
-----
000000    000000    000000    000000

USER ACCESS ACCESS COUNT
-----
NO USERS IN ACCESS LIST
```

```
ID ACCESS ACCESS COUNT CLASS ENTITY NAME
```

```
-----  
NO ENTRIES IN CONDITIONAL ACCESS LIST
```

Now, let's RLIST the web server certificate issued by our CA:

```
RL DIGTCERT 01.OU=Webserver.O=antoff@IT.C=US ALL
```

Note: The value of the UACC denotes the status of the certificate: ALTER if the certificate is TRUSTed or HIGHTRUSTed and ??????? - if the certificate is NOTRUSTed. The APPLDATA field contains the userid associated with the certificate - in our case of CA certificate, it is the IBM supplied userid irrcerta. Userid irrcerta will be always in APPLDATA for NOTRUSTed certificates.

Part of the output is shown below:

Example 6. Output from RLIST DIGTCERT

```
CLASS    NAME  
-----  
DIGTCERT 01.OU=Webserver.O=antoff@IT.C=US  
LEVEL OWNER    UNIVERSAL ACCESS YOUR ACCESS WARNING  
-----  
00    ANTOFF    ALTER                ALTER        NO  
  
INSTALLATION DATA  
-----  
NONE  
  
APPLICATION DATA  
-----  
SUWEB  
-----
```

Note:The APPLDATA field contains the userid associated with certificate - the web server started task userid: SUWEB.

To display all digital certificates created in your RACF database, issue the command:

```
SR CLASS(DIGTCERT)
```

with the following partial output:

Example 7. Display of all digital certificates

```
0D8B4FEEAAD2185BF4756A9D29E17FFB.OU=Class1@Public@Primary@Certification@Autho  
rity.O=VeriSign,Inc..C=US  
00.personal-basic@thawte.com.CN=Thawte@Personal@Basic@CA.OU=Certification@Servi  
ces@Division.O=Thawte@Consulting.L=Cape@Town.SP=Western@Cape.C=ZA  
00.personal-freemail@thawte.com.CN=Thawte@Personal@Freemail@CA.OU=Certification
```

2.8. Display Key rings

The RACDCERT command with keyword LISTRING is used to display information about key rings.

LISTRING keyword

To display the CA key ring, enter the following command:

```
RACDCERT ID(SUPKI) LISTRING(PKIRING)
```

with the output shown below:

Example 8. Display key ring for PKI Services userid

Digital ring information for user SUPKI:

Ring:

>PKIRING<

Certificate Label Name	Cert Owner	USAGE	DEFAULT
PKIServicesCA	CERTAUTH	PERSONAL	YES

The ring information must have USAGE PERSONAL and DEFAULT YES.

To display the Web Server keyring use the command:

RACDCERT ID(SUWEB) LISTRING(SSLRING)

with the output shown below:

Example 9. Display key ring for Web Server userid

Digital ring information for user SUWEB:

Ring:

>SSLRING<

Certificate Label Name	Cert Owner	USAGE	DEFAULT
PKIServicesCA	CERTAUTH	PERSONAL	NO
SSL PKI	ID(SUWEB)	PERSONAL	YES

RLIST command

Issue command

RLIST DIGTRING SUPKI.PKIRING ALL

with the output shown below:

Example 10. Output from RLIST DIGTRING

CLASS NAME

DIGTRING SUPKI.PKIRING

LEVEL	OWNER	UNIVERSAL ACCESS	YOUR ACCESS	WARNING
00	ANTOFF	NONE	NONE	NO

INSTALLATION DATA

NONE

APPLICATION DATA

SUPKI

SECLEVEL

NO SECLEVEL

CATEGORIES

NO CATEGORIES

SECLABEL

NO SECLABEL

AUDITING

FAILURES(READ)

GLOBALAUDIT

NONE

NOTIFY

NO USER TO BE NOTIFIED

CREATION DATE LAST REFERENCE DATE LAST CHANGE DATE
(DAY) (YEAR) (DAY) (YEAR) (DAY) (YEAR)

120 03 120 03 120 03

ALTER COUNT CONTROL COUNT UPDATE COUNT READ COUNT

000000 000000 000000 000000

USER ACCESS ACCESS COUNT

NO USERS IN ACCESS LIST

ID ACCESS ACCESS COUNT CLASS ENTITY NAME

NO ENTRIES IN CONDITIONAL ACCESS LIST

Note: The keyword ID() must always be used to list any key ring. Also ID() may be placed after LISTRING(). And again, the key ring name is case-sensitive !

To display all digital key rings created in your RACF database, issue the command:

SR CLASS(DIGTRING)

with output shown below:

Example 11. Display of all digital Key rings

```
LDAPKI.LDAPKI
PKISRV.DPKIRING
PKISRV3.WEBPKI3
SUPKI.PKIRING
WEBSRV.SSLRING
WEBSRV.WEBPKI3
SUWEB.SSLRING
```

2.9 Daemon and server control for PKI userid and Surrogate userid

We permit SUPKI to BPX.DAEMON and BPX.SERVER in class FACILITY and PKISRV to BPX.SERVER in class FACILITY:

```
PE BPX.DAEMON CL(FACILITY) ID(SUPKI) ACC(R)
PE BPX.SERVER CL(FACILITY) ID(SUPKI) ACC(R)
PE BPX.SERVER CL(FACILITY) ID(PKISRV) ACC(R)
```

2.10 Allow PKI userid to act as CA

The following commands allow a PKI userid to act as a CA:

```
RDEF FACILITY IRR.DIGTCERT.GENCERT UACC(NONE) OWNER(PKIADM) AUDIT(ALL)
RDEF FACILITY IRR.DIGTCERT.LIST UACC(NONE) OWNER(PKIADM)
RDEF FACILITY IRR.DIGTCERT.LISTRING UACC(NONE) OWNER(PKIADM)
PE IRR.DIGTCERT.GENCERT CL(FACILITY) ID(SUPKI) ACC(CONTROL)
PE IRR.DIGTCERT.LIST CL(FACILITY) ID(SUPKI) ACC(R)
PE IRR.DIGTCERT.LISTRING CL(FACILITY) ID(SUPKI) ACC(R)
```

2.11 Allow Web server to access its own key ring

To authorize your Web server to access its own certificate and key ring, issue the following commands:

```
PE IRR.DIGTCERT.LIST CL(FACILITY) ID(SUWEB) ACC(R)
PE IRR.DIGTCERT.LISTRING CL(FACILITY) ID(SUWEB) ACC(R)
```

2.12 Allow Web server userid to switch identity to Surrogate userid

The commands to allow the web server userid to act on behalf of the surrogate userid PKISRV (or switch identity to Surrogate userid) are:

```
RDEF SURROGAT BPX.SRV.PKISRV
PE BPX.SRV.PKISRV CL(SURROGAT) ID(SUWEB) ACC(R)
```

2.13 Profile for PKI Services procedure in class STARTED

```
RDEF STARTED PKISRV**.** STDATA(USER(SUPKI) GROUP(STC)) OWNER(PKIADM)
```

This covers more than one PKI server, e.g. in case you would like to have a few members in SYS1.PROCLIB named PKISRV1, PKISRV2 etc. However, you can run only one PKI server per LPAR at a time.

2.14 Allow access for SUPKI to OCSF

For the PKI Services procedure to start, the PKI Services userid needs access to the OCSF services. Provide this access by:

```
PE CDS.** CL(FACILITY) ID(SUPKI) ACC(R)
```

Note: Userids having SPECIAL can issue the RACDCERT command with all keywords and parameters. For a list of all possible profiles in the class FACILITY for protection of the RACDCERT command refer to chapter 20 in the *Security Server RACF Security Administration Guide SA22-7683*.

2.15 ICSF

Here we assume that ICSF is active. The command to store a certificate into ICSF is:

```
RACDCERT CERTAUTH ADD('PKI.CAPKI.BACKUP.P12BIN') +
PASSWORD('xxxxxxx') ICSF
```

ADD(data-set-name) Specifies that a digital certificate is to be defined. The specified data set must contain the digital certificate. The data set containing the digital certificate or certificate package must be cataloged, and cannot be a PDS or a PDS member. The RECFM expected by RACDCERT is VB. When the ADD keyword is specified, RACDCERT dynamically allocates and opens the specified dataset, and reads the certificate from it as binary data.

PASSWORD(' pkcs12-password') specifies the password that is associated with the PKCS#12 certificate package. This keyword is required if the data set

is PKCS#12 and it must not be specified if the data set is not PKCS#12. The 'pkcs12-password' can be up to 255 characters in length, is case sensitive, and can contain blanks.

ICSF specifies that RACF should attempt to store the private key associated with this certificate in the ICSF PKDS. This applies when the key is introduced to RACF by issuing the ADD keyword for PKCS#12 certificate packages, and when an existing certificate profile containing a non-ICSF private key is replaced by issuing the ADD keyword.

Note: The issuer of the RACDCERT command must have READ access to the *data-set-name* data set to prevent an authorization abend from occurring when the data set is read. Each userid can be associated with more than one digital certificate but they must be added individually. The specified data set should contain only one digital certificate. The command reads the certificate from the data set, updates the user's profile, and creates the DIGTCERT profile.

Note: The password specified will be visible on the screen, so care should be taken to prevent it from being viewed when entered. Because PKCS#12 passwords do not follow the normal TSO/E rules for password content, they cannot be suppressed as they normally would be.

If the GENCERT keyword creates a public/private key pair and ICSF is being used to store private keys, GENCERT creates an ICSF key label in the format IRR.DIGTCERT.userid.cvtsname.ebcdic-stck-value, where userid is the owning user ID, CVTSNAME is the system name as taken from the CVT, and ebcdic-stck-value is an EBCDIC version of the current store clock value. If the key is associated with a CA certificate, userid is set to CERTIFAUTH. If the key is associated with a site certificate, then userid is set to SITECERTIF. You have to make sure that you have created a profile in class CSFKEYS as follows:

```
RDEF CSFKEYS IRR.DIGTCERT.** OWNER(PKIADM)
PE IRR.DIGTCERT.** CL(CSFKEYS) ID(PKIADM) ACC(R)
```

Note: This is the violation message if you were not authorized to profile IRR.DIGTCERT.** in class CSFKEYS:

```
ICH408I USER(ANTOFF ) GROUP(SYS1 ) NAME(THEO ANTOFF)
IRR.DIGTCERT.CERTIFAUTH.SC64.B953A8B607F4EC81 CL(CSFKEYS )
INSUFFICIENT ACCESS AUTHORITY
FROM IRR.DIGTCERT.** (G)
ACCESS INTENT(READ ) ACCESS ALLOWED(NONE)
```

2.16 Protect certificate functions

A list of all PKI functions (end-user and administrative-user) can be found in the **Security Server RACF Security Administration Guide SA22-7683**.

You have to allow your surrogate PKI userid to use end-user certificate functions with the commands:

```
RDEF FACILITY IRR.RPKISERV.** OWNER(PKIADM)
PE IRR.RPKISERV.** CL(FACILITY) ID(PKISRV) ACC(CONTROL)
```

Note: By default (and we are using the default) in the pkiserv.tmpl file, 4 of the 8 types of PKI certificates are auto-approved (the two SAF certificates, the 2 Year PKI Browser Certificate for Authenticating to z/OS and the 5 Year PKI Intermediate CA certificate). The reason to give CONTROL to PKISRV for the end-user functions of the PKI callable service is to cater for those auto-approvals not requiring administrator's actions.

A single profile IRR.RPKISERV.PKIADMIN in class FACILITY can protect all 6 administrative-user functions (for a description of these functions, see p.88). To disallow access to the

administrative-user functions of PKI Services for the surrogate PKI userid, place PKISRV on its access list with NONE:

**RDEF FACILITY IRR.RPKISERV.PKIADMIN OWNER(PKIADM)
PE IRR.RPKISERV.PKIADMIN CL(FACILITY) ID(PKISRV) ACC(NONE)**

Note: The reason for explicit access NONE for PKISRV is that we placed this userid on the access list of IRR.RPKISERV.** covering both end-user and administrative functions. Discrete profile IRR.RPKISERV.PKIADMIN covers all administrative functions, but it is independent of the preceding generic profile, not a more specific profile derived from it.

We assume that all members of your PKIADM group have SYSTEM SPECIAL. If this is not the case, then members without SYSTEM SPECIAL must have UPDATE to profile IRR.RPKISERV.PKIADMIN.

2.17.Run RACF commands

Now, once you have got the whole picture you may wish to use one of three almost equivalent options to execute the commands:

1. Run the REXX member IKYSETUP from SYS1.SAMPLIB after copying it to your own library and updating with your shop values according to the tables in chapter 4 of z/OS Security Server PKI Services Guide and Reference SA22-7693.
2. Run the CLIST shown in example 12 as a JCL job. It is a stack of all RACF commands explained in 2.4, "Setting up RACF environment for PKI services" on page 55 after you have carefully checked that you have already all the controls discussed and explained in 2.3, "Setting up RACF environment for PKI prereqs" on page 49.
3. Execute the commands included in chapter 2.4 as you progress reading it. Do not forget refreshes for RACLISTED or GENERIC classes.

Example 12. Define RACF environment for PKI Services

```
PROC 0
/*****/
/* RACF ENVIRONMENT FOR PKI SERVICES */
/* */
/* A sample CLIST to execute all RACF commands for installing */
/* PKI Services */
/* You must change this CLIST to satisfy your own shop standards */
/* Comment out commands which you may decide not to run */
/* Prereq 1: run CLIST RACF ENVIRONMENT FOR PREREQS FOR PKI SERVICES */
/* Prereq 2: all members of group PKIADM have SYSTEM SPECIAL */
/* If you do not want your PKI Admin to have SYSTEM SPECIAL than give */
/* them access CONTROL to all profiles IRR.DIGTCERT in class FACILITY */
/* Maintained by: your RACF/PKI administrators */
/* History: */
/* date */
/* */
/*****/
/* GROUPS */
/* */
/*****/
AG PKIADM SUPGROUP(JOBROLE) OWNER(JOBROLE) OMVS(AUTOGID)
CO (user1 user2 etc userids of your PKI administrators) GROUP(PKIADM)
AG STC SUPGROUP(DFLT) OWNNER(DFLT) OMVS(AUTOGID)
/* ALG (your RACF admin group) OMVS(AUTOGID) */
/* ALG (your default group for started task userids) GID(AUTOGID) */
/*****/
/* */
```

```

/* USERIDS */
/* */
/*****
AU SUPKI NAME('PKI DAEMON') DFLTGRP(STC) OWNER(STC) +
OMVS(AUTOUID HOME('/') PROG('/bin/sh') ASSIZE(256000000) +
THREADS(512)) NOPASSWORD
AU PKISRV NAME ('PKI SURROG') DFLTGRP(MISC) OWNER(MISC) +
OMVS(AUTOUID HOME('/') PROG('/bin/sh') ASSIZE(256000000) +
THREADS(512)) NOPASSWORD RESTRICTED
/*****
/* */
/* PERMITS DUE TO RESTRICTED ATTRIBUTE FOR PKISRV */
/* */
/*****
PE IRR.DIGTCERT.** CL(FACILITY) ID(PKISRV) ACC(R)
PE IRR.DIGTCERT.ADD CLASS(FACILITY) ID(PKISRV) ACC(R)
PE IRR.DIGTCERT.ADDRING CLASS(FACILITY) ID(PKISRV) ACC(R)
PE IRR.DIGTCERT.CONNECT CLASS(FACILITY) ID(PKISRV) ACC(R)
PE IRR.DIGTCERT.DELETE CLASS(FACILITY) ID(PKISRV) ACC(R)
PE IRR.DIGTCERT.DELRING CLASS(FACILITY) ID(PKISRV) ACC(R)
PE IRR.DIGTCERT.LIST CLASS(FACILITY) ID(PKISRV) ACC(R)
PE IRR.DIGTCERT.LISTRING CLASS(FACILITY) ID(PKISRV) ACC(R)
PE IRR.DIGTCERT.REMOVE CLASS(FACILITY) ID(PKISRV) ACC(R)
PE ** CL(PROGRAM) ID(PKISRV) ACC(R)
/*PE ** CL(APPL) ID(PKISRV) ACC(R) */
/*****
/* */
/* DATASETS */
/* */
/*****
AD 'PKI.**' OWNER(PKIADM) AUDIT(ALL)
PE 'PKI.**' ID(PKIADM) ACC(ALTER)
PE 'PKI.**' ID(SUPKI) ACC(CONTROL)
/*****
/* */
/* CA CERTIFICATE */
/* */
/*****
RACDCERT GENCERT CERTAUTH SUBJECTSDN(OU('PKI Dept') +
O('antoff IT') C('US')) WITHLABEL('PKIServicesCA') +
NOTAFTER(DATE(2020/01/01))
RACDCERT CERTAUTH ALTER(LABEL('PKIServicesCA')) HIGHTRUST
RACDCERT CERTAUTH EXPORT(LABEL('PKIServicesCA')) +
DSN('PKI.CAPKI.BACKUP.P12BIN') FORMAT(PKCS12DER) +
PASSWORD('xxxxxxx')
RACDCERT CERTAUTH EXPORT(LABEL('PKIServicesCA')) +
DSN('PKI.CAPKI.DERBIN') FORMAT(CERTDER)
RACDCERT CERTAUTH ADD('PKI.CAPKI.BACKUP.P12BIN') +
PASSWORD('xxxxxxx') ICSF
/*****
/* */
/* PKI SERVICES KEY RING */
/* */
/*****
RACDCERT ADDRING(PKIRING) ID(SUPKI)
RACDCERT ID(SUPKI) CONNECT(CERTAUTH LABEL('PKIServicesCA') +
RING(PKIRING) USAGE(PERSONAL) DEFAULT)
/*****
/* */
/* WEB SERVER CERTIFICATE (SSL) */
/* */
/*****

```

```

RACDCERT GENCERT ID(SUWEB) +
SIGNWITH(CERTAUTH LABEL('PKIServicesCA')) +
SUBJECTSDN(CN('antoff.it.com') O('antoff IT') +
OU('Web server') C('US')) +
WITHLABEL('SSL PKI') NOTAFTER(DATE(2020/01/01))
/*****/
/*                                     */
/* WEB SEREVER KEY RING                 */
/*                                     */
/*****/
RACDCERT ADDRING(SSLRING) ID(SUWEB)
RACDCERT ID(SUWEB) CONNECT(CERTAUTH LABEL('PKIServicesCA') +
RING(SSLRING))
RACDCERT ID(SUWEB) CONNECT(ID(SUWEB) +
LABEL('SSL PKI') RING(SSLRING) USAGE(PERSONAL) DEFAULT)
/*****/
/*                                     */
/* PROFILE IN CLASS SURROGAT FOR SURROGATE USERID */
/*                                     */
/*****/
RDEF SURROGAT BPX.SRV.PKISRV OWNER(PKIADM)
PE BPX.SRV.PKISRV CL(SURROGAT) ID(SUWEB) ACC(R)
/*****/
/*                                     */
/* DAEMON AND SERVER CONTROL             */
/*                                     */
/*****/
PE BPX.DAEMON CL(FACILITY) ID(SUPKI) ACC(R)
PE BPX.SERVER CL(FACILITY) ID(SUPKI) ACC(R)
PE BPX.SERVER CL(FACILITY) ID(PKISRV) ACC(R)
/*****/
/*                                     */
/* CERTIFICATE ADMINISTRATION            */
/*                                     */
/*****/
RDEF FACILITY IRR.DIGTCERT.** OWNER(PKIADM)
RDEF FACILITY IRR.DIGTCERT.ADD OWNER(PKIADM)
RDEF FACILITY IRR.DIGTCERT.ADDRING OWNER(PKIADM)
RDEF FACILITY IRR.DIGTCERT.CONNECT OWNER(PKIADM)
RDEF FACILITY IRR.DIGTCERT.DELETE OWNER(PKIADM)
RDEF FACILITY IRR.DIGTCERT.DELRING OWNER(PKIADM)
RDEF FACILITY IRR.DIGTCERT.GENCERT OWNER(PKIADM)
RDEF FACILITY IRR.DIGTCERT.LIST OWNER(PKIADM)
RDEF FACILITY IRR.DIGTCERT.LISTRING OWNER(PKIADM)
RDEF FACILITY IRR.DIGTCERT.REMOVE OWNER(PKIADM)
PE IRR.DIGTCERT.** CL(FACILITY) ID(SUPKI PKISRV) ACC(C)
PE IRR.DIGTCERT.ADD CL(FACILITY) ID(*) ACC(R)
PE IRR.DIGTCERT.ADD CL(FACILITY) ID(SUPKI PKISRV) ACC(U)
PE IRR.DIGTCERT.ADDRING CL(FACILITY) ID(*) ACC(R)
PE IRR.DIGTCERT.CONNECT CL(FACILITY) ID(*) ACC(R)
PE IRR.DIGTCERT.DELETE CL(FACILITY) ID(*) ACC(R)
PE IRR.DIGTCERT.DELRING CL(FACILITY) ID(*) ACC(R)
PE IRR.DIGTCERT.GENCERT CL(FACILITY) ID(SUPKI PKISRV) ACC(C)
PE IRR.DIGTCERT.LIST CL(FACILITY) ID(*) ACC(R)
PE IRR.DIGTCERT.LIST CL(FACILITY) ID(SUPKI PKISRV) ACC(C)
PE IRR.DIGTCERT.LISTRING CL(FACILITY) ID(*) ACC(R)
PE IRR.DIGTCERT.LISTRING CL(FACILITY) ID(SUPKI PKISRV) ACC(C)
PE IRR.DIGTCERT.REMOVE CL(FACILITY) ID(*) ACC(R)
RDEF FACILITY IRR.RPKISERV.** OWNER(PKIADM)
RDEF FACILITY IRR.RPKISERV.PKIADMIN OWNER(PKIADM)
PE IRR.RPKISERV.** CL(FACILITY) ID(PKISRV) ACC(C)
PE IRR.RPKISERV.PKIADMIN CL(FACILITY) ACC(NONE)

```

```

*****/
/* */
/* CLASS STARTED */
/* */
/*****/
RDEF STARTED PKISRV.** STDATA(USER(SUPKI) GROUP(STC)) +
OWNER(PKIADM)
/*****/
/* */
/* ACCESS TO OCSF */
/* */
/*****/
PE CDS.** CL(FACILITY) ID(SUPKI) ACC(R)
/*****/
/* */
/* CLASSES CSFKEYS AND CSFSERV */
/* */
/*****/
RDEF CSFKEYS IRR.DIGTCERT.** OWNER(PKIADM)
PE IRR.DIGTCERT.** CL(CSFKEYS) ID(SUPKI PKIADM) ACC(R)
PE CSF* CL(CSFSERV) ID(PKISRV PKIADM LDAPSTU SUPKI) ACC(R)
/*****/
/* */
/* SETROPTS REFRESH */
/* */
/*****/
SETR RACLIST(CSFSERV) REFR
SETR RACLIST(CSFKEYS) REFR
SETR GENERIC(DATASET) REFR
SETR RACLIST(FACILITY) REFR
SETR RACLIST(STARTED) REFR
SETR RACLIST(SURROGAT) REFR
SETR WHEN(PROGRAM) REFR
/*****/

```

3 RACF administration for PKI Services

This section describes the possible activity by RACF administrators after PKI Services has been set up and customized.

3.1 Creating a helpdesk function

Usually your helpdesk group needs to have inquiry ability in the frame of PKI Services. The commands for this scenario are shown below:

```

AG HELPDESK SUPGROUP(JOBROLE) OWNER(JOBROLE) OMVS(AUTOGID)
AO HDUSR1 DFLTGRP(EMPL) OWNER(EMPL) OMVS(AUTOUID)
CO HDUSR1 GROUP(HELPDESK)
PE 'PKI.**' ID(HELPDESK) ACC(R)
PE IRR.RPKISERV.PKIADMIN CL(FACILITY) ID(HELPDESK) ACC(R)
PE IRR.DIGTCERT.LIST CL(FACILITY) ID(HELPDESK) ACC(C)
PE IRR.DIGTCERT.LISTRING CL(FACILITY) ID(HELPDESK) ACC(C)

```

Access CONTROL to the last two profiles ensures that helpdesk members can list anybody's certificates or key rings by issuing the commands:

```
RACDCERT ID(userid) LIST
```

for any userid having certificates registered in RACF and

```
RACDCERT CERTAUTH LIST
```

in case they wish to find all CA certificates.

On the other hand IRR.DIGTCERT.LIST and IRR.DIGTCERT.LISTRING must have UACC(READ) or ID(*) on their access list with READ. Thus every userid will be able to display their certificates or key rings by issuing:

RACDCERT LIST to obtain detailed information for each of his certificates

RACDCERT LISTRING(*) to obtain detailed information for each of his key rings

Note: Helpdesk can use RACDCERT ID(userid) LISTRING(*) to find all rings associated with a userid, but a similar command RACDCERT LIST(*) fails with a message

```
IKJ56712I INVALID KEYWORD, *  
IKJ56703A REENTER THIS OPERAND - .
```

3.2 Administering certificates with HostIdMapping extension

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

In order to allow the web server to accept logins from clients who have been issued PKIServices certificates with HostIdMapping extensions, you have to create profiles in the RACF class SERVAUTH.

Define a profile in class SERVAUTH for each server (host) name you want your Web server to honor when accepting logins for certificates containing HostIdMapping extensions. The profile has the format IRR.HOST.hostname where the value of hostname is usually a domain name, such as wtsc63oe.itso.ibm.com. This domain name must be entered in the entry for HostIdMap in /web/pki1/pkiserv/pkiserv.tmpl, but without the subject ID portion in the APPL section.

RDEF SERVAUTH IRR.HOST.ANTOFF.IT.COM UACC(NONE)

Permit your Web server started task userid with READ :

PE IRR.HOST.ANTOFF.IT.COM CL(SERVAUTH) ID(SUWEB) ACC(R)

Now activate and raclist class SERVAUTH:

```
SETR CLASSACT(SERVAUTH)  
SETR RACLIST(SERVAUTH)  
SETR RACLIST(SETRAUTH) REFR
```

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.

3.3 Display certificates using utilities iclview and vosview

Sometimes, you may wish to display your Issued Certificates List (ICL) or your Certificates Request List (CRL). These lists are kept respectively in VSAM datasets PKI.WEBPKI.ICL and PKI.WEBPKI.OST. These lists are independent of RACF, although ICL may contain certificates registered in RACF.

To display ICL and CRL two utilities can be used: iclview and vosview, respectively. Issue the OMVS command and do the following to display all certificates issued by PKI Services:

```
cd /usr/lpp/pkiswerv/lib
/usr/lpp/pkiserv/bin/iclview/ \`pki.webpki1.icl\`
The output from iclview utility is shown in example 13 below:
```

Example 13. Output from iclview utility

```
Cert 1:
ISSUED (CA's certificate)
Issued at 2003-04-30 17:59:57
Last changed 2003-04-30 17:59:57
Subject: OU=PKI Dept,0=antoff IT,C=US
Issuer: OU=PKI Dept,0=antoff IT,C=US
Requester:
AppData:
Serial Number: 0
Email flag: Off

Cert 2: Theo Antoff
ISSUED (Issued certificate)
Issued at 2003-05-15 16:30:05
Last changed 2003-05-15 16:30:05
Subject: CN=Theo Antoff,OU=Class 1 Internet Certificate,0=antoff IT,C=US
Issuer: OU=PKI Dept,0=antoff IT,C=US
Requester: Theo Antoff
AppData: 1YBSSL
Serial Number: 15
Email flag: Off
```

To display all certificate requests received by PKI Services after issuing OMVS from ISPF, option 6, do:

```
cd /usr/lpp/pkiswerv/lib
/usr/lpp/pkiserv/bin/vosview/ \`pki.webpki.ost\`
```

The output from vosview utility is shown in example 14:

Example 14. Displaying all Certificate requests received by PKI Services

```
Object key = 1
Last used key = 108, CRL serial number = 4, ARL serial number = 4
High DP = 0, Low DP = 1
name = ""
tid = ??????????????????????????????????
appdata =
comment =
data len = 20
flags = 0 - Type = ??? ObjSt?????????
Creation time is: 2003/05/16 13:49:06
Last modified time is: 2003/05/19 12:05:55
-----
Object key = 2
name = ""
tid = ??????????????????????????????????
appdata =
comment =
data len = 33
flags = 0 - Type = ??? ObjSt?????????
Creation time is: 2003/05/16 13:49:08
Last modified time is: 2003/05/18 20:49:15
-----
```

Object key = 108
name = "ta"
tid = 1jJVzyGROUwe2Qn07++++++
apldata = 1YBSSL
comment =
data len = 1116
flags = 1040010 - Type = Cert State = RA CertReqActive
Creation time is: 2003/05/19 12:05:55

Last modified time is: 2003/05/19 12:05:55

Records starting with Object key = 1 and Object key =2 contain system data only Record starting with object key=108 is a request sent by user "ta" for 1Year PKI SSL Browser certificate (the type of certificate is represented by apldata =1YBSSL). For more information about the meaning of the other fields in the records, refer to Chapter 20 of **z/OS Security Server PKI Services Guide and Reference**.

3.4 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.

Perform the following steps to establish PKI Services as an intermediate certificate authority:

1. Determine which certificate authority will be acting as a higher authority for your PKI Services. (This could be a public Certificate Authority, such as VerySign).
2. Create a new certificate request from your self-signed CA certificate by entering the following RACF command:

```
RACDCERT CERTAUTH GENREQ(LABEL('PKIServicesCA')) +  
DSN('PKI.CAPKI.BACKUP.P12BIN')
```

3. Send the certificate request to the higher certificate authority, following the procedures that higher authority requires.
4. After the certificate has been issued, receive the certificate back into the certificate data set (PKI.CAPKI.BACKUP.P12BIN).

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 copy and paste the certificate into the empty data set. 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.

5. Receive the certificate back into the RACF data base by entering the following RACF command:

```
RACDCERT CERTAUTH ADD('PKI.CAPKI.BACKUP.P12BIN')
```

6. Export the certificate in DER format to the export data set by entering the following RACF command:

**RACDCERT CERTAUTH EXPORT(LABEL('PKIServicesCA')) DSN('PKI.CAPKI.DERBIN') +
FORMAT(CERTDER)**

7. To make your new certificate available to your clients, set up the /var/pkiserv/ directory by performing step 2 through step 4 on page 67 in “Steps for setting up the /var/pkiserv directory” from **Security Server PKI Services Guide and Reference SA22-7693**.

3.5 Renewing your PKI Services CA 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.

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:

**RACDCERT CERTAUTH GENREQ(LABEL('PKIServicesCA')) +
DSN('PKI.CAPKI.BACKUP.P12BIN')**

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:

**RACDCERT CERTAUTH GENCERT('PKI.CAPKI.BACKUP.P12BIN') SIGNWITH(CERTAUTH +
LABEL('PKIServicesCA'))**

3. Alternately, if your PKI Services certificate authority is an intermediate certificate authority, repeat the steps in 2 to 4 on p.67 in **Security Server PKI Services Guide and Reference SA22-7693**.

3.6 Controlling applications that call 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 maintaining 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 submitted request 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>

where, **<function>**

is one of the end-user function names in the preceding list. The userid for the application (userid 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 Task Control Block (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 applicants user id.

CONTROL (or user ID is RACF SPECIAL) Access is permitted, and no subsequent access checks are made.

For requests in which the application has READ and UPDATE access, subsequent access checks are performed against the IRR.DIGTCERT.<function> FACILITY profiles. These checks identical to the checks the RACDCERT TSO command makes.

The following table summarizes the access requirements for the user ID whose access is checked:

Table 2. 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

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 to perform read operations (QUERYREQS, QUERYCERTS, REQDETAILS, and CERTDETAILS)

UPDATE 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.

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.

3.7. Using encrypted passwords for LDAP servers

PKI Services uses an LDAP directory to store certificates. LDAP requires authenticating (binding) to the directory. You can do this by using a distinguished name and passwords. Passwords for binding (to multiple LDAP directories) can be encrypted or in clear text. Your security policy should stipulate whether or not to use encrypted LDAP bind passwords. You store information about passwords in the PKI Services configuration file, pkiserv.conf. If you

do not need the bind password for the LDAP server to be encrypted, you specify the values for Server1, AuthName1 and AuthPwd1 in the pkiserv.conf configuration file, otherwise you should comment them out.

Perform the following steps to use encrypted LDAP bind passwords:

1. Define a fixed-name profile LDAP.BINDPW.KEY in RACF class KEYSMSTR by entering the following command, replacing the highlighted value with your own key selected randomly using hexadecimal values from 0 to F:

```
RDEFINE KEYSMSTR LDAP.BINDPW.KEY SSIGNON(KEYENCRYPTED(A7D8E09ACDEF35AC))
```

SSIGNON is a segment having two possible keywords:

KEYMASKED used when ICSF is not active

KEYENCRYPTED used when ICSF is active

The value of the key is A7D8E09ACDEF35AC. (Replace this with your own key.)

2. Activate the KEYSMSTR class by entering the following command:

```
SETROPTS CLASSACT(KEYSMSTR)
```

Now, you have two alternatives:

- use profiles in RACF class LDAPBIND for each LDAP directory.
- use fixed-name profile IRR.PROXY.DEFAULTS in class FACILITY.

Profiles in class LDAPBIND

For each LDAP directory, create your own profile (in our case LDAPKI) in RACF class LDAPBIND by entering the following command:

```
RDEFINE LDAPBIND LDAPKI +  
PROXY(LDAPHOST(ldap://antoff.it.com:3389) +  
BINDDN('CN=LDAPADMIN') +  
BINDPW(LDAPADMIN))
```

Note: The bind password is case-sensitive.

Now you have to update the pkiserv.conf file as follows:

```
# Server1=antoff.it.com:3389  
# AuthName1=CN=LDAPADMIN  
# AuthPwd1=LDAPADMIN  
-----  
BindProfile1=LDAPKI
```

Note: The first 3 statements are commented out. The last statement points to the profile in class LDAPBIND.

Profile in Class FACILITY

If you intend to use profile IRR.PROXY.DEFAULTS in class FACILITY instead of the LDAPBIND class for encrypted LDAP bind passwords, enter the following command to create the profile:

```
RDEFINE FACILITY IRR.PROXY.DEFAULTS +  
PROXY(LDAPHOST(ldap://antoff.it.com:3389) +  
BINDDN('CN=LDAPADMIN') +  
BINDPW(LDAPADMIN))
```

You have to update pkiserv.conf by commenting out all 4 statements:

```
# Server1=antoff.it.com:3389
# AuthName1=CN=LDAPADMIN
# AuthPw1=LDAPADMIN
-----
#BindProfile1=LDAPKI
```

As can be seen using LDAPBIND is more flexible because it allows the creation of as many profiles for LDAP servers as you would need to authenticate to, while using FACILITY provides only one LDAP server.

List PROXY Segment

You may wish to list the segment PROXY to check your work with the RLIST command. If you are using the LDAPBIND class, enter the following:

RLIST LDAPBIND LDAPKI PROXY NORACF

This command displays the following information:

```
CLASS    NAME
-----  ----
LDAPBIND LDAPKI

PROXY INFORMATION
-----
LDAPHOST= LDAP://antoff.it.com:3389
BINDDN=  CN=LDAPADMIN
BINDPW=  YES
```

If you are using the IRR.PROXY.DEFAULTS profile of the FACILITY class, enter the following command:

RLIST FACILITY IRR.PROXY.DEFAULTS PROXY NORACF

This command displays the following information:

```
CLASS NAME
-----  ----
FACILITY IRR.PROXY.DEFAULTS
PROXY INFORMATION
-----
LDAPHOST= LDAP://antoff.it.com:3389
BINDDN=  CN=LDAPADMIN
BINDPW=  YES
```