

The Digital Certificate Journey from RACF to PKI Services

RACF USER GROUP New York, NY

October 14th 2004

Wai Choi
IBM Corporation
Poughkeepsie, NY

Phone: (845) 435-7623
e-mail: wchoi@us.ibm.com



Agenda

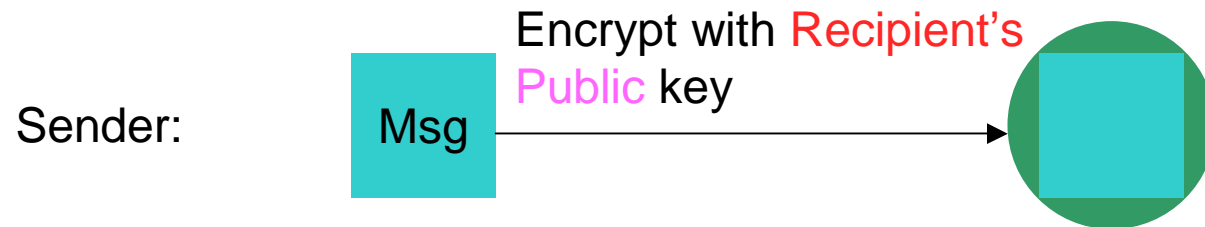
- **PKI Basics**
- **RACF Support for Certificates**
- **PKI Services Support for Certificates**
- **What's new on PKI Services for z/OS Release 5**
- **What's new on RACDCERT support for z/OS Release 6**
- **PKI Services demo**

What is PKI?

- **Public Key Infrastructure based on the public key cryptography to create, manage, store, distribute, verify digital certificates**
- **Not like the secret key cryptography which encrypts and decrypts with the same key**
- **Involves a public-private key pair, encrypts with one key and decrypts with its partner key**
- **To facilitate 2 main goals of secure communication – confidentiality and integrity**

Encryption (for confidentiality) Under Public Key Cryptography

Encrypting a message:



Decrypting a message:



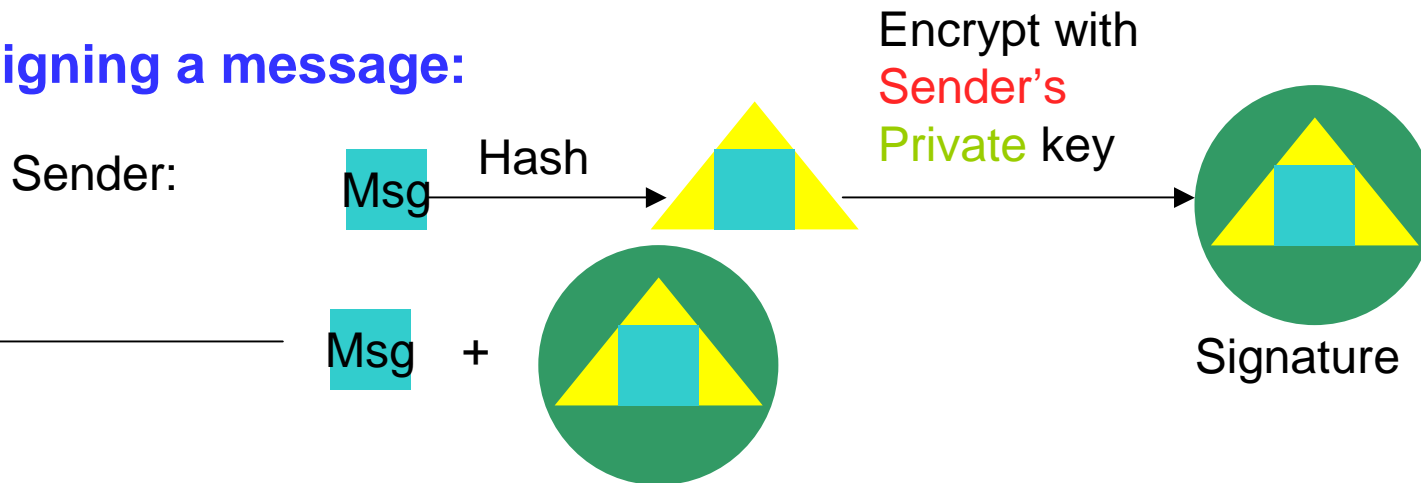
Keys:

 Plain text

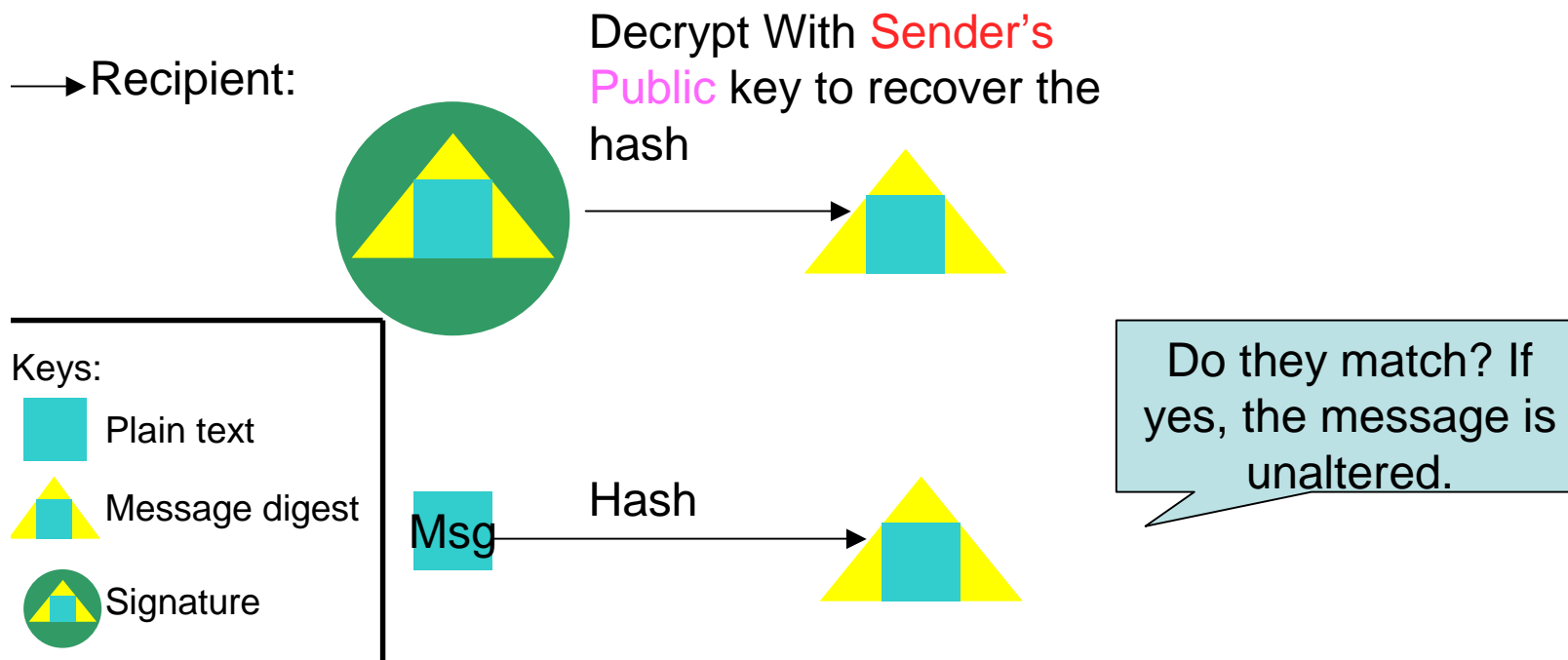
 Encrypted text

Signing (for integrity) Under Public Key Cryptography




Signing a message:



Verifying a message:



Keys:

-  Plain text
-  Message digest
-  Signature

Public key and Certificate

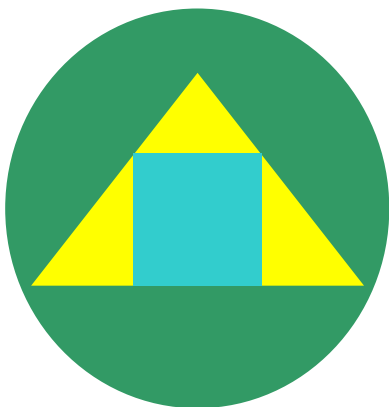
- **Public key alone can not tell who the key owner is**
- **Need a trusted third party, Certificate Authority (CA), to bind a public key to a subject through a certificate**
- **The Certificate Authority signs the certificate with its private key to prove its authenticity**

What's inside a Certificate?

Certificate Info

version
serial number
signature algorithm ID
issuer's name
validity period
subject's name
subject's public key
extensions

Certificate Signature

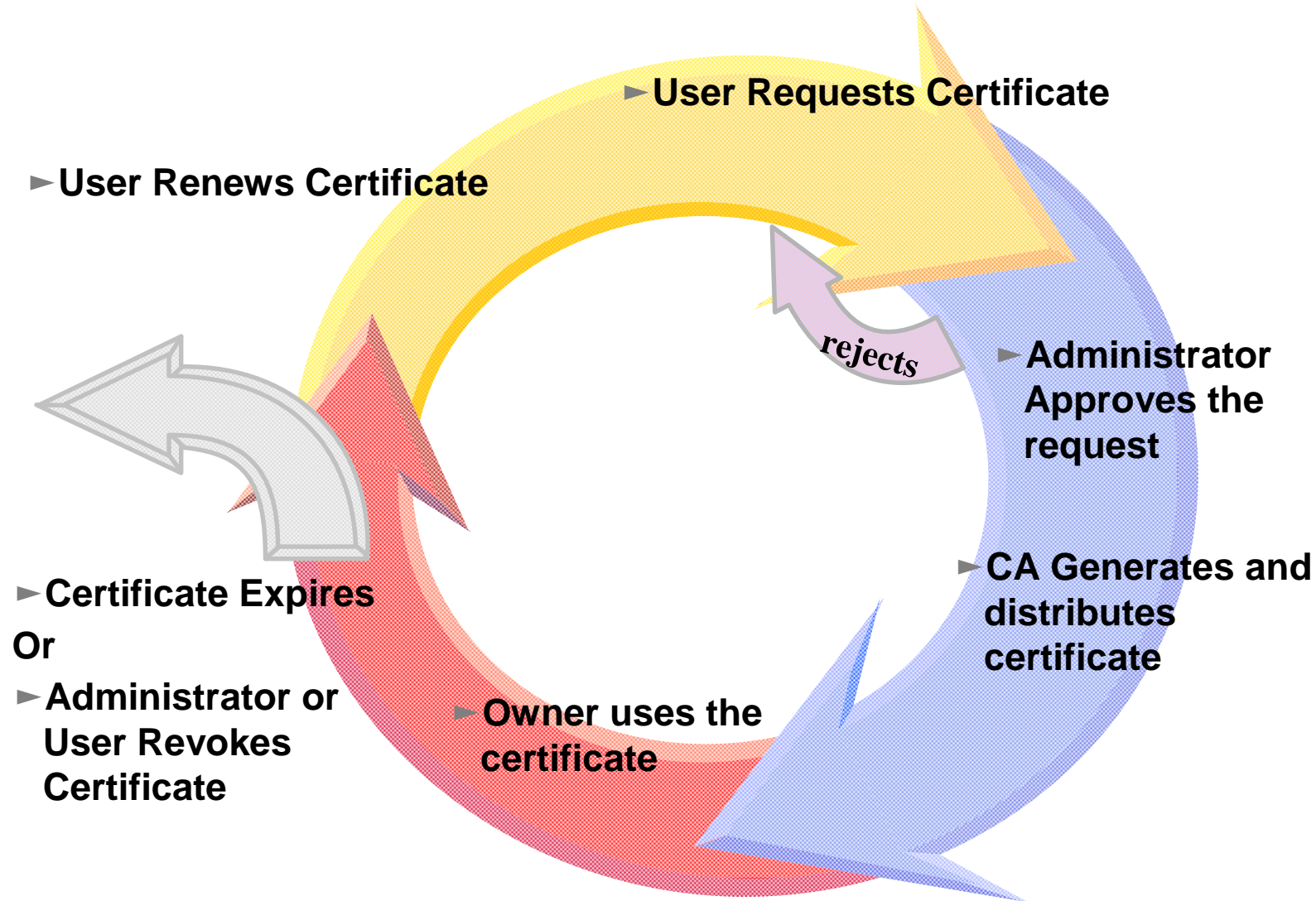


This is the hash/encrypt algorithm used in the signature

The certificate binds a public key to a subject

CA signs the above cert info by encrypting the hash with its private key

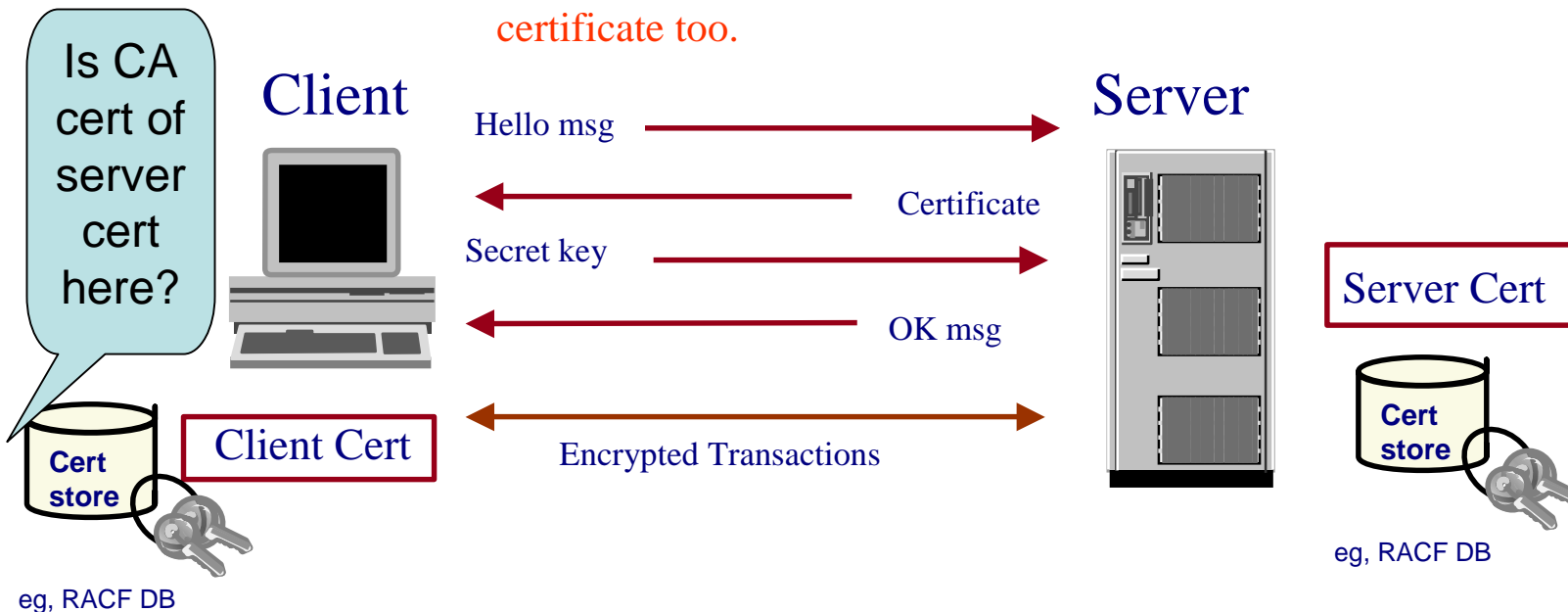
Certificate Life Cycle – This is why you need PKI



A common use of Certificate - handshake

1. Client sends a 'hello' msg to server
2. Server sends its certificate to client
3. Client validates the server's certificate
4. Client encrypts a secret key with server's public key and sends it to server
5. Server decrypts the secret key with its private key
6. Server encrypts a 'handshake OK' msg with the secret key and sends it to client
7. Client trusts server, business can be conducted

* Note the above steps illustrate server authentication. For client authentication, server needs to validate client's certificate too.



eg, RACF DB

eg, RACF DB

Two Basic PKI Operations

Certificate generation (In response to a user request)

- Both RACF and PKI Services can be used as a Certificate Authority

Certificate validation

involves the questions of:

- Whether you *trust* the issuer of the certificate – is it in your certificate store, key ring...
- Whether the certificate has a valid *signature* of the issuer
- Whether the certificate is *expired*
- Whether the certificate has been *revoked* (see slide 36)
 - PKI Services supports Certificate Revocation Lists (CRLs), RACF doesn't
- Whether the certificate contains *information that is specific* to your application that uses that certificate. This includes specific extensions that your application is looking for.

RACF support for Certificates

1. Certificate generation – RACDCERT GENCERT, GENREQ (see slide 37 for examples)

● RACDCERT GENCERT:

- **GENerates a CERTificate, and optionally public-private key**
- **Can utilize z/OS hardware crypto for private key generation, storage and operation**
- **Certificate can be self signed or signed by another certificate's corresponding private key**
- **May create certificates for other servers, includes those on non-z/OS platform**

RACF support for Certificates...

● RACDCERT GENREQ:

- **GENerates a REQuest by copying information, including the public key, from a previously created certificate**
- **Usually issued after GENCERT a self signed certificate**
- **The request is signed with the private key associated with the specified certificate**
- **The request is saved to a data set in the Base64 format which can be used in cut and paste (see slide 40)**
- **The created certificate request can be submitted (e.g. e-mail it, paste it into a web page, etc) to a CA for issuance**
- **Note there is no key generation in GENREQ**

RACF support for Certificates...

2. Certificate installation – RACDCERT ADD, ADDRING, CONNECT (see slides 38, 39 for examples)

● RACDCERT ADD:

- Install a certificate to RACF with or without private key

● RACDCERT ADDRING

- Create a key ring (for handshake process)
- Certificate must be placed in a key ring before it can be used by other middleware, eg. SSL

□ RACDCERT CONNECT

- Place a certificate in a key ring
- You may place your own certificates and other's certificate in your key ring
- A certificate can be placed in more than one key ring, even with different usages – PERSONAL, CERTAUTH and SITE
- The TRUST status of the connected certificate in the ring tells if the certificate can be used

RACF support for Certificates...

3. Certificate administration – RACDCERT LIST, ALTER, DELETE, REMOVE, ...

● RACDCERT LIST:

- Display certificate information for a userid

● RACDCERT ALTER

- Change the TRUST/NOTRUST status or the label of a certificate

□ RACDCERT DELETE

- Remove a certificate from RACF

□ RACDCERT REMOVE

- Remove a certificate from a key ring

□ And more...

Common exploiters of certificates on z/OS

Exploiter	Connect the server cert to the ring, eg. 'MYRING'	Where/How to specify the RACF key ring
FTP Server	RACDCERT ID(FTPSVR) CONNECT(LABEL('FTP Cert')) RING(MYRING) DEFAULT)	FTP.DATA file KEYRING MYRING
TN3270 Server	RACDCERT ID(TNSVR) CONNECT(LABEL('TN Cert')) RING(MYRING) DEFAULT)	PROFILE.TCPIP file KEYRING SAF MYRING
HTTP Server	RACDCERT ID(WEBSVR) CONNECT(LABEL('WEB Cert')) RING(MYRING) DEFAULT)	httpd.conf file Keyfile MYRING SAF
Websphere MQ	RACDCERT ID(QM1) CONNECT(LABEL ('ibmWebSphereMQ MQ1')) RING(MYRING)) <i>Note: label of the cert must start with 'ibmWebSphereMQ'</i>	MQ command ALTER QMGR SSLKEYR (MYRING)

Introduction to PKI Services

- **New component on z/OS since V1R3**
- **Closely tied to RACF, but supports more functions than RACDCERT**
- **Complete Certificate Authority /Registration Authority (CA/RA) package**
 - **Full certificate life cycle management: request, create, renew, revoke**
- **Generation and administration of certificates via customizable web pages**
- **Support automatic or administrator approval process**
- **Create Certificate Revocation Lists (CRLs)**
- **Certificates and CRLs can be posted to LDAP**
- **Provides email notification for completed certificate request and expiration warnings**

Introduction to PKI Services...

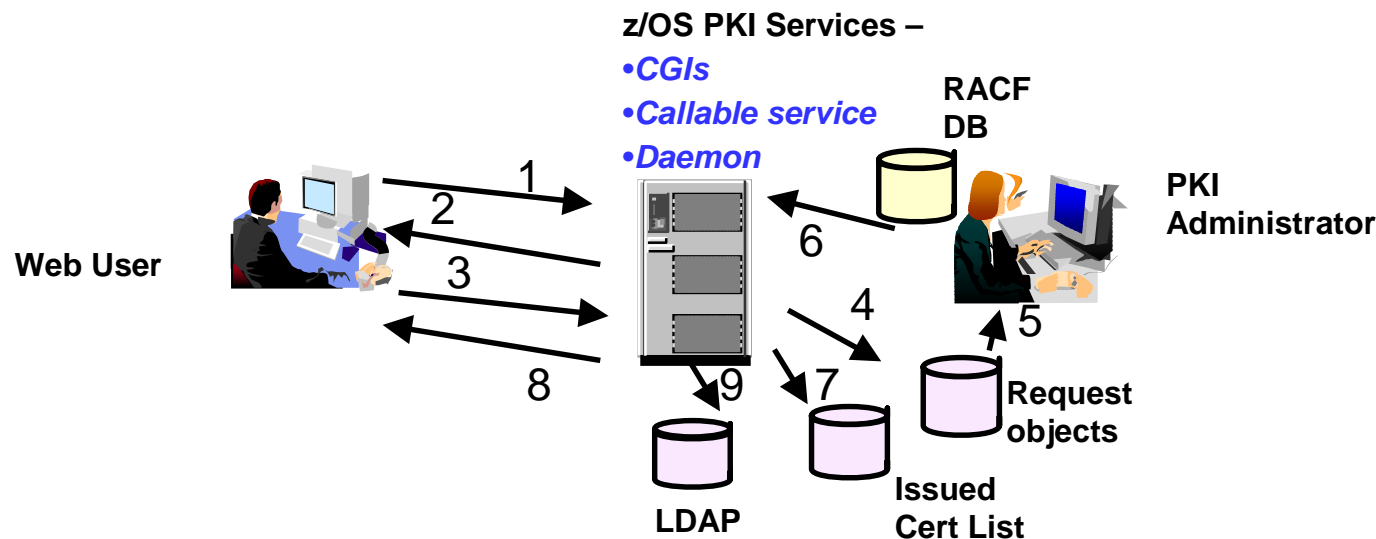
- **Provides Trust Policy Plug-in for certificate validation**
- **Manual - "PKI Services Guide and Reference"**

Benefits of using PKI Services on z/OS

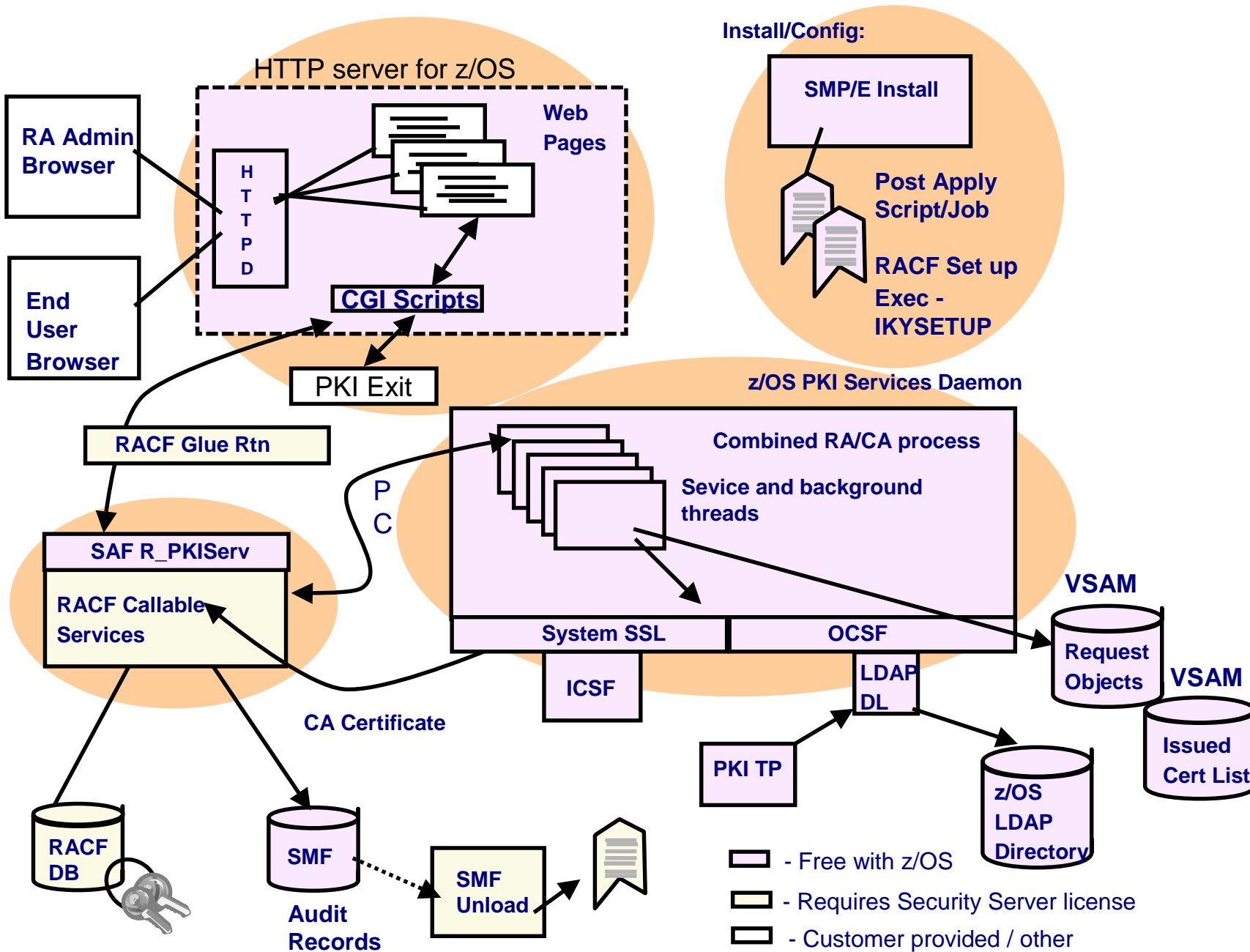
- **No additional cost with z/OS, no need to pay a third party CA for certificates**
- **Relatively low mips to drive thousands of certificates**
- **Leverage existing z/OS skills and resources**
- **Ability to host Digital Certificate management for the banks, government agencies...**
- **Run independently of other workloads**
- **Run in separate z/OS partitions (integrity of zSeries LPARs)**
- **Scalable (Sysplex exploitation)**
- **Secure with zSeries cryptography**

z/OS PKI Services Process Flow – a simplified sample view

1. User contacts PKI Services to request for certificate
2. CGI constructs a web page for user to input information
3. CGI packages all the info and send to the callable service
4. Callable service calls the daemon to generate the request object and put it in the Request objects DB
5. Administrator approves the request through the administrator web page
6. CGI calls callable service which in turn calls the daemon to create the certificate, sign with the CA key in the RACF DB
7. Certificate is placed in the Issued Cert List DB
8. Certificate is sent to the user
9. Certificate is posted to LDAP



z/OS PKI Services Architecture

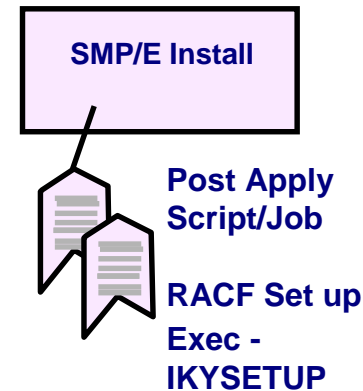


z/OS PKI Services Architecture...

● IKYSETUP

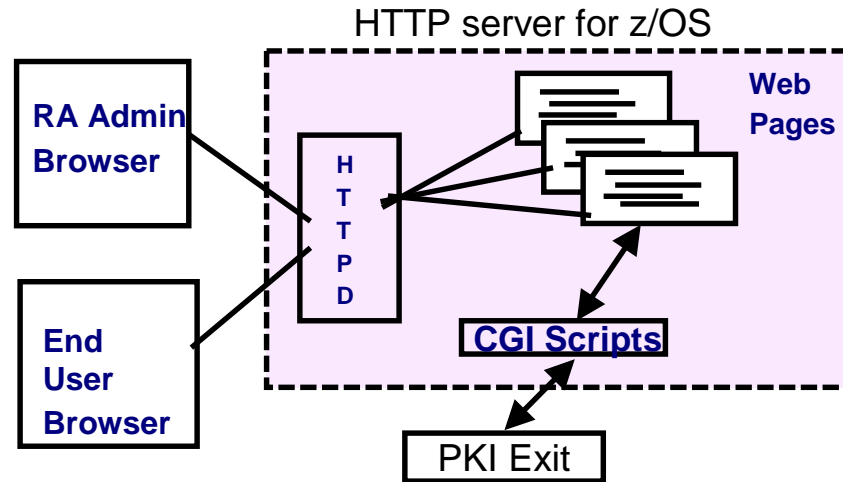
- A REXX exec shipped in SYS1.SAMPLIB to perform RACF administration tasks for setting up PKI Services:
 - Add administration groups, daemon user IDs, client surrogate IDs
 - Set up access control profiles
 - Create or install PKI Services CA certificate and Web Server certificate for the PKI web pages

Install/Config:



z/OS PKI Services Architecture...

Browser/CGI interface



- Web page contents are defined in a certificate template file, pkiserv.tmpl (see slide 41 for more details)
- The CGI tasks include:
 - read the template file to form the web pages
 - package up all the input values from the web page and constant values from the template file as parameters to call the R_PKIServ callable service
 - provide hooks to call installation-provided exit routine for customization

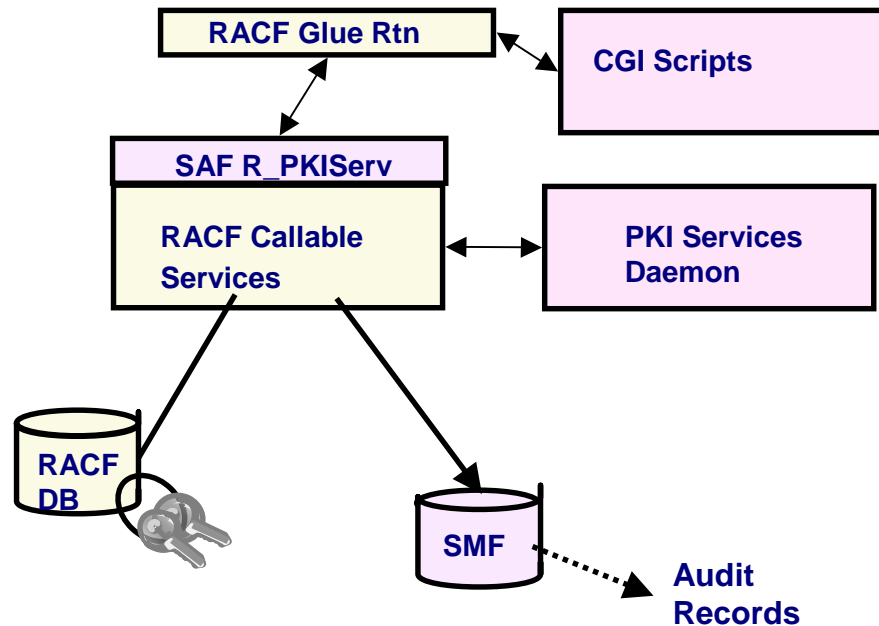
z/OS PKI Services Architecture...

● Browser/CGI interface...

- You may request a browser certificate or a server cert
 - Browser certificate
 - The public-private key pair is generated by the browser
 - The browser will then create a certificate request signed with the private key
 - The returned certificate can be installed directly into the browser
 - Server certificate
 - Similar process as in browser certificate generation except that the public-private key pair is generated on the server side
- Note PKI Services is not involved in generating key pairs in browser nor server certificate generation.

z/OS PKI Services Architecture...

● SAF callable service – R_PKIServ



- Interface between CGIs and the PKI Services Daemon (through the glue routine)
- Performs authorization checking and parameter validation
- Provides functions for end user and administrator

z/OS PKI Services Architecture...

- **SAF callable service – R_PKIServ...**

- End user functions are acting on certificates:

- Request

- Export

- Verify

- Renew

- Suspend

- Revoke

z/OS PKI Services Architecture...

● SAF callable service – R_PKIServ...

– Administrator functions are acting on certificate requests and certificates:

□ Query (request, certificate)

□ Approve (request)

□ Modify (request)

□ Reject (request)

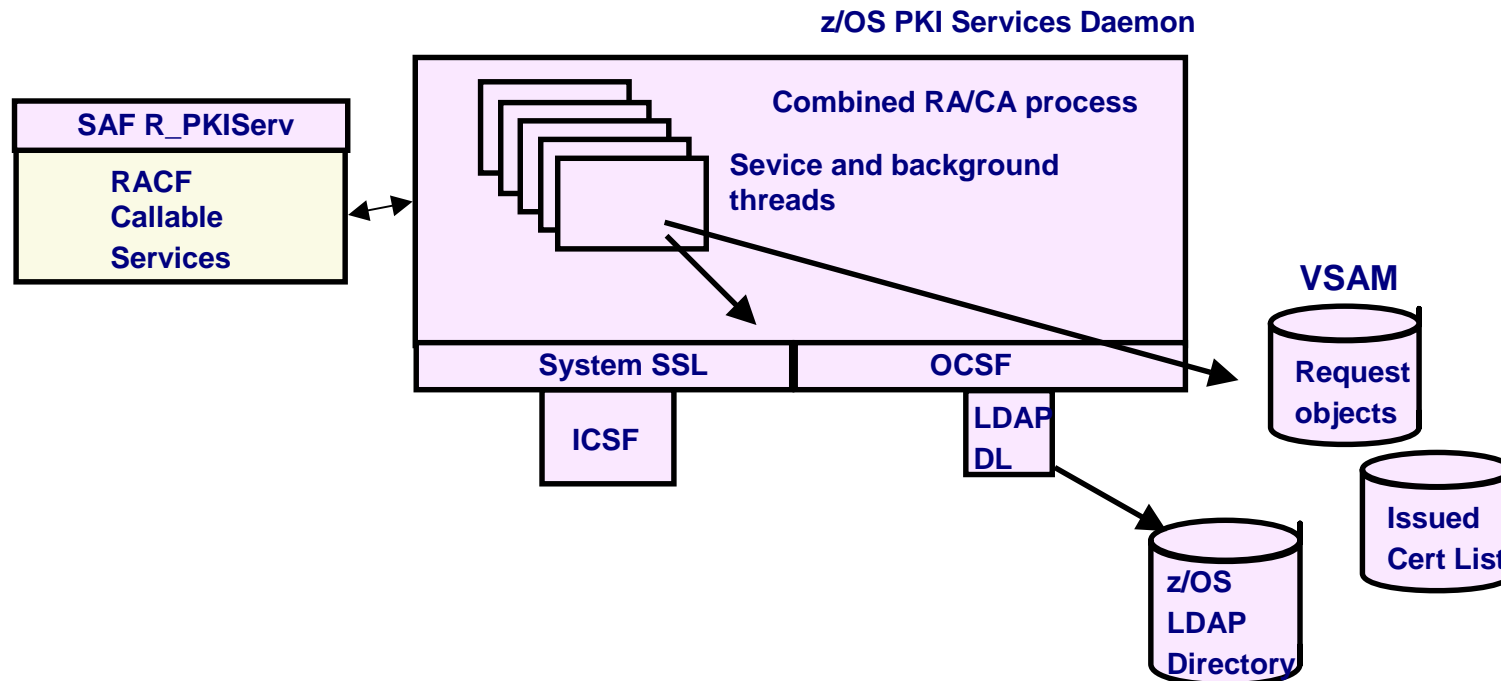
□ Suspend (certificate)

□ Resume (certificate)

□ Revoke (certificate)

z/OS PKI Services Architecture...

● PKI Services Daemon



- Invoked by the R_PKIServ callable service
- Perform the real work
- Read the configuration file, pkiserv.conf, to determine the set up values (see slide 42 for more details)
- VSAM datasets are used to store request objects (object store) and Issued Certificates List (ICL)

What's new in PKI Services on z/OS Release 5

- **Support Multiple Application Domains**
 - To subset administration users and end users into different domains, providing different templates, functions
- **Support Certificate Suspension and Resumption**
 - Certificates may be suspended for a period of time (grace period) and may be resumed later
- **New certificate extensions**
 - Certificate Revocation List (CRL) Distribution Points
 - Extended Key Usage
 - Authority Information Access

What's new in PKI Services on z/OS Release 5...

- **Enhanced certificate extension**

- Certificate Policies

- may be created on a per template basis

- Key Usage

- create a 1-1 correspondence between key usage name and key usage bit (cf HANDSHAKE = digitalSignature + keyEncipherment)

- **Performance/Scaling Enhancements**

- Improve the performance of queries on the object store and ICL through

- Addition of VSAM Alternate Indexes

- Setting up VSAM buffering in the PKI Services started procedure JCL

- Use AMP= on the DD cards

What's new in RACDCERT Support on z/OS Release 6

The longer a key pair is used, the more likely it is to be cracked. The key pair should be periodically changed. Two new functions are provided:

- **RACDCERT REKEY**

- Make a self-signed copy of the original certificate with a new public-private key pair

- **RACDCERT ROLLOVER**

- Finalize the REKEY operation

- Cert with usage PERSONAL: all keyring occurrences of the old certificate will be replaced with the new one

- Cert with usage CERTAUTH or SITE: the new cert will be added to all keyring occurrences of the old one

Major Prerequisite Products

- ▶ **RACF (or equivalent)**
 - For storing PKI CA certificate
- ▶ **IBM z/OS HTTP Server**
 - For web page interface
- ▶ **LDAP Directory**
 - For publishing issued certificates and CRLs
- ▶ **ICSF (optional)**
 - For more secure CA private key
- ▶ **z/OS Communications Server (optional)**
 - For email notification

References

- **PKI Services web site:**

<http://www.ibm.com/servers/eserver/zseries/zos/pki>

- **PKI Services Red Book:**

<http://www.redbooks.ibm.com/abstracts/sg246968.html>

- **RACF web site:**

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

- **Cryptographic Services**

- ▶ PKI Services Guide and Reference (SA22-7693)
- ▶ OCSF Service Provider Developer's Guide and Reference (SC24-5900)
- ▶ ICSF Administrator's Guide (SA22-7521)
- ▶ System SSL Programming (SC24-5901)

- **Security Server Manuals:**

- ▶ RACF Command Language Reference (SC28-1919)
- ▶ RACF Security Administrator's Guide (SC28-1915)
- ▶ RACF Callable Services Guide (SC28-1921)
- ▶ LDAP Administration and Use (SC24-5923)

- **IBM HTTP Server Manuals:**

- ▶ Planning, Installing, and Using (SC31-8690)

- **Other Sources:**

32 PKIX - <http://www.ietf.org/html.charters/pkix-charter.html>

Questions???

PKI Services Demo

Appendices

- How to determine if a certificate is revoked
- RACDCERT examples
- A Base64 encoded certificate request example
- PKI Services template file information
- PKI Services configuration file information
- PKI Services key ring information
- PKI Services utilities -- vosview and iclview

How to determine if a certificate is revoked

- The application contacts the CA every time when the certificate is used. The contact information is specified in the certificate's Authority Information Access (AIA) extension.
- The CA publishes CRL to a public place, eg. LDAP server, periodically. The application checks if the certificate is on the Certificate Revocation List (CRL) published by the CA.
- As time goes, the CRL may be very large, publishing and retrieving CRL may be time consuming. Creating CRL Distribution Points to publish partial CRLs is a way to solve this problem. Again CRL Distribution Point is a certificate extension.

RACDCERT Examples

● RACDCERT GENCERT:

- ▶ Create a self signed certificate and public-private key pair using ICSF for a CA -- CERTAUTH represents the 'ID' of a CA, no 'SIGNWITH' needed for self signed cert

```
-RACDCERT CERTAUTH GENCERT SUBJECT(...) ICSF...
```

- ▶ Create a certificate and public-private key pair using PCICC for ID WEBSRV, signed with a CA certificate named 'theCA cert'

```
-RACDCERT ID(WEBSRV) GENCERT SUBJECT(...) PCICC  
SIGNWITH(CERTAUTH LABEL('theCA cert'))...
```

- ▶ Create a certificate based on a certificate request specified in the dataset 'CERTREQ.B64' for ID MYID – you get the request from other system and you want your local CA to sign it. This generates certificate only, no key pair is created.

```
-RACDCERT ID(MYID) GENCERT('CERTREQ.B64')  
SIGNWITH(CERTAUTH LABEL('theCA cert'))...
```

● RACDCERT GENREQ:

- ▶ Generate a request based on an existing certificate named 'My Self Signed Cert' in RACF for ID OUTSRV and put the request in the data set 'CERTREQ.B64'

```
-RACDCERT ID(OUTSRV) GENREQ(LABEL('My Self Signed  
Cert')) DSN(CERTREQ.B64)...
```