Digital Certificates – From Concept to Implementation

Vanguard Las Vegas, NV
Session FD3
June 25th 2012

Wai Choi, CISSP
IBM Corporation
RACF/PKI Development & Design
Poughkeepsie, NY

e-mail: wchoi@us.ibm.com

Trademarks

The following are trademarks of the International Business Machines Corporation in the United States and/or other countries.

- CICS*
- DB2*
- IBM*
- IBM (logo)*
- OS/390*
- RACF*
- Websphere*
- z/OS*

* Registered trademarks of IBM Corporation

The following are trademarks or registered trademarks of other companies.

Identrus is a trademark of Identrus, Inc
VeriSign is a trademark of VeriSign, Inc
Microsoft, Windows and Windows NT are registered trademarks of Microsoft Corporation.

* All other products may be trademarks or registered trademarks of their respective companies.

Notes:

Performance is in Internal Throughput Rate (ITR) ratio based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput improvements equivalent to the performance ratios stated here.

IBM hardware products are manufactured from new parts, or new and serviceable used parts. Regardless, our warranty terms apply.

All customer examples cited or described in this presentation are presented as illustrations of the manner in which some customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics will vary depending on individual customer configurations and conditions.

This publication was produced in the United States. IBM may not offer the products, services or features discussed in this document in other countries, and the information may be subject to change without notice. Consult your local IBM business contact for information on the product or services available in your area.

All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.

Information about non-IBM products is obtained from the manufacturers of those products or their published announcements. IBM has not tested those products and cannot confirm the performance, compatibility, or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Prices subject to change without notice. Contact your IBM representative or Business Partner for the most current pricing in your geography.

Agenda

- **Part 1 - Introduction to digital certificates**
  - Symmetric vs. Asymmetric Encryption
  - What are digital certificates
  - Certificate types and contents
- **Part 2 - Overview of certificate utilities available on z/OS**
  - RACF RACDCERT
  - System SSL gskkyman
  - PKI Services

Agenda

- **Part 3 - RACDCERT in depth**
  - Certificate Name Filtering
  - Host ID Mapping
  - Certificate / Key Sharing
  - Certificate renewal
  - Common Gotchas
  - Enhancements
- **Part 4 – Hot topics on certificates**
  - An example to set up secure FTP
  - Build or Buy
  - Outage due to expired certificates

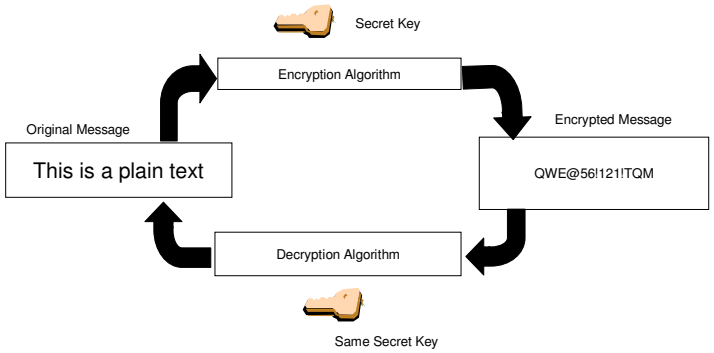
Agenda

- **Part 5 - Introduction to PKI Services**
  - Comparison with RACDCERT
  - Enhancements
  - A user experience
- **Part 6 - Hands on Lab on PKI Services**
  - **Submit and approve a certificate request for**
    - A certificate with key pair generated by the browser – EX 1
    - A certificate with key pair generated by PKI Services – EX 2
    - A certificate with key pair generated on a z/OS server – EX 3
  - **View the installed certificate from the IE broswer – EX 4**
  - **Revoke/Suspend a certificate – EX 5**
  - **Check the certificate status – EX 6**
    - Certificate Revocation List (CRL)
    - Online Certificate Status Protocol (OCSP)
  - **Customize PKI Services – EX 7**
    - Configuration file – pkiserv.conf
    - Template file – pkiserv.tmpl

Part 1 – Introduction to Digital Certificates

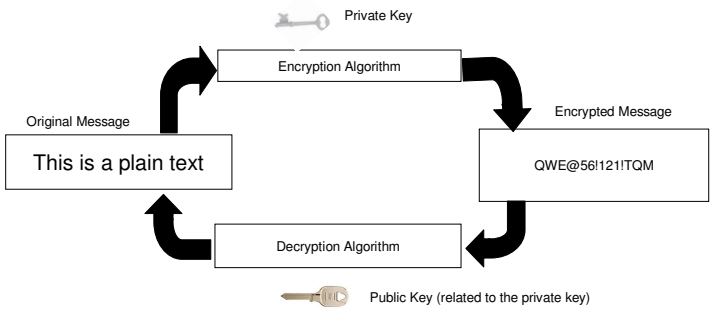
Symmetric Encryption

- **Same key used for both encryption and decryption**
- **Provide data confidentiality**
- **Fast, used for bulk encryption/decryption**
- **Securely sharing and exchanging the key between both parties is a major issue**
- **Common algorithms: DES, Triple DES, AES**



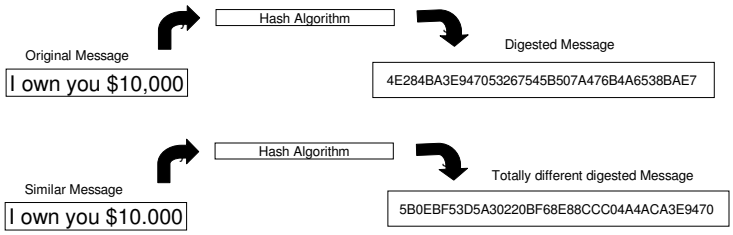
Asymmetric Encryption

- **2 different keys - Public/private key pairs**
- **A public key and a related private key are numerically associated with each other.**
- **Provide data confidentiality, integrity and non repudiation**
- **Data encrypted/signed using one of the keys may only be decrypted/verified using the other key.**
- **Very expensive computationally**
- **Public key is freely distributed to others, private key is securely kept by the owner**
- **Common algorithms: RSA, DSA, ECC**



Message Digest (Hash)

- **A fixed-length value generated from variable-length data**
- **Unique:**
  - the same input data always generates the same digest value
  - tiny change in data causes wide variation in digest value
  - Theoretically impossible to find two different data values that result in the same digest value
- **One-way: can't reverse a digest value back into the original data**
- **NOT based on a key**
- **Play a part in data integrity and origin authentication**
- **Common algorithms: SHA1, SHA256**

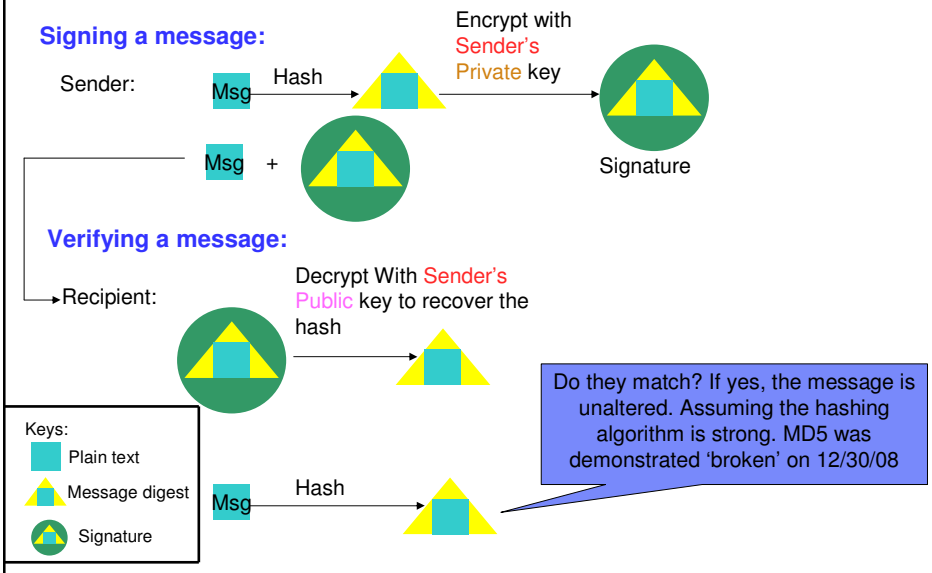


Encryption (for confidentiality)

Encrypting a message:

Decrypting a message:

Signing (for integrity and non repudiation)



What is a Digital Certificate (1 of 2)

- **Generally digital certificates provide identity to a person or a server**
  - Person - like an ID card
  - Server – like a business license
- **To establish an identity or credential to be used in electronic transactions**
- **It binds the public key to the identity to be used by applications that are based on public key protocols. (e.g. SSL/TLS)**
- **Issued by a trusted third party called Certificate Authority (CA) that can ensure validity**

What is a Digital Certificate(2 of 2)

- **Packaging of the information is commonly known as the x.509 digital certificate. X.509 defines the format and contents of a digital certificate.**
  - IETF RFC 5280
- **Digital certificates been in existence for over 20 years**
- **Have evolved over time to not only bind basic identity information to the public key but also how public key can be used, additional identity data, revocation etc.**

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

Version 1, 2, 3

This is the hash/encrypt algorithm used in the signature, eg. sha256RSA – Beware of MD5!!!

The certificate binds a public key to a subject

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

The private key is NOT in the certificate. It is kept in a key store

You can NOT change ANY of the certificate information!

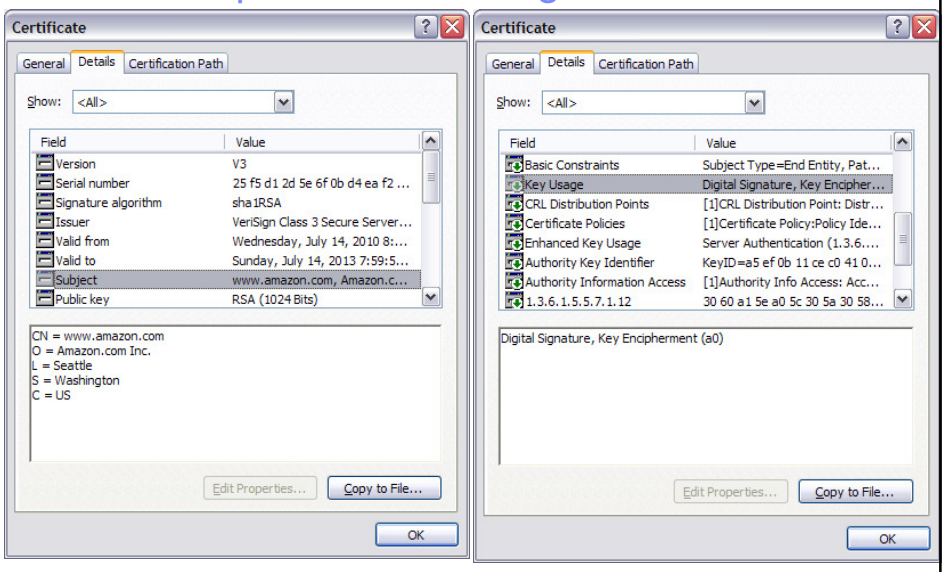
Extensions of a x.509 digital Certificate(1 of 2)

- Adds additional definitions to a certificate and its identity information
- 15+ currently defined
- Top 6 extensions of interest
  - Authority Key Identifier
  - Subject Key Identifier
  - Key Usage
  - Subject Alternate Name
  - BasicConstraints
  - CRL Distribution Point

Extensions of a x.509 digital Certificate(2 of 2)

- **Authority Key Identifier – Unique identifier of the signer**
- **Subject Key Identifier – Unique identifier of the subject**
- **Key Usage – defines how the public key can used**
  - Digital Signature
  - Key Encipherment
  - Key Agreement
  - Data Encipherment
  - Certificate Signing
  - CRL signing
- **Subject Alternate Name – additional identity information**
  - Domain name
  - E-mail
  - URI
  - IP address
- **Basic Constraints – Certificate Authority Certificate or not**
- **CRL Distribution – Locating of Revoked certificate information**

Example of a x.509 digital Certificate



Relationship between Certificate and Certificate Store

- Certificate must be placed in a certificate store before it can be used by an application to perform identification or validation
- The application needs to retrieve the certificate and/or its corresponding private key from the store
- On z/OS, many components like Communication Server, HTTP Server call System SSL APIs to access the store
- Certificate store = key ring = key file

Types of digital certificates – who issues it

- **Self signed**
  - Self-issued
  - Issuer and subject names identical
  - Signed by itself using associated private key
- **Signed Certificate**
  - Signed/issued by a trusted Certificate Authority Certificate using its private key.
  - By signing the certificate, the CA certifies the validity of the information. Can be a well-known commercial organization or local/internal organization.

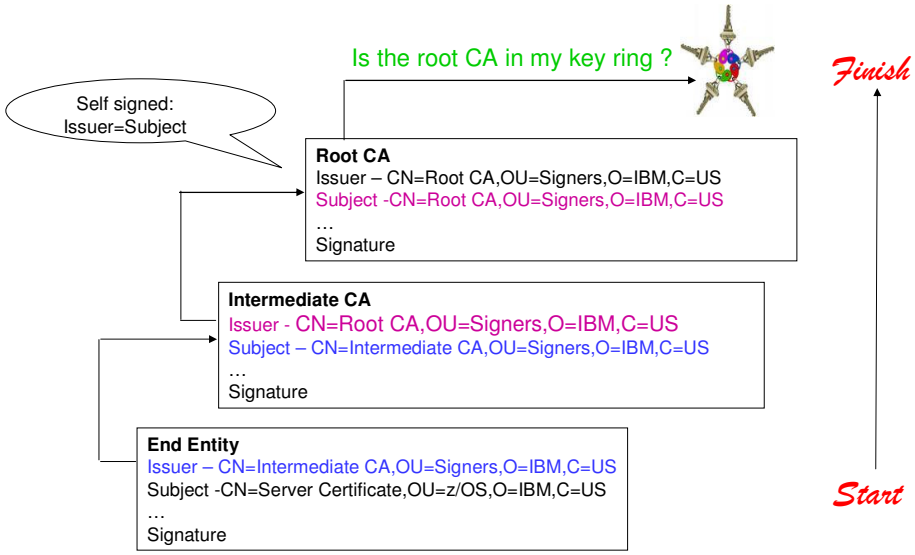
Types of digital certificates – what is the usage

- **Secure Socket Layer (SSL) certificate**
  - Install on a server that needs to be authenticated, to ensure secure transactions between server and client
- **Code Signing certificate**
  - Sign software to assure to the user that it comes from the publisher it claims
- **Personal certificate**
  - Identify an individual, enable secure email – to prove that the email really comes from the sender and /or encrypt the email so that only the receiver can read it
- **More (name it whatever you want)…**
  - wireless certificate, smart card certificate…
- **Certificate Authority (CA) certificate**
  - Used to sign other certificates
  - Root CA: the top
  - Intermediate CA: signed by root CA or other intermediate CA

Types of digital certificates – what is the usage

- **Site certificate (in RACF)**
  - The usage assigned to a certificate when it is connected to a RACF key ring indicates its intended purpose
  - There may be a few certificate validation applications which treat a certificate that is connected to a key ring with usage site as a valid certificate authority certificate to bypass the normal certificate verification tests during SSL handshake, for example, an expired certificate can be considered trusted
  - Having a SITE certificate in RACF does not benefit you if the validation application does not make use of it

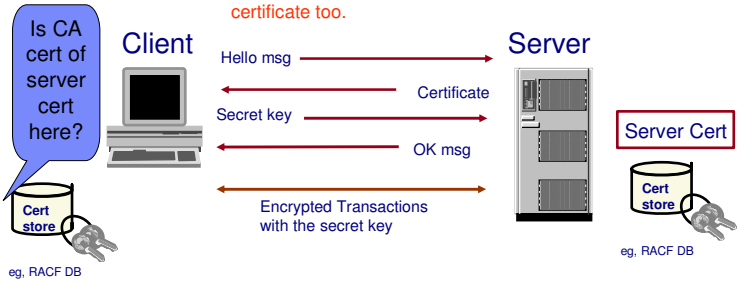
Certificate Chain Validation



Key ring plays a role in SSL 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

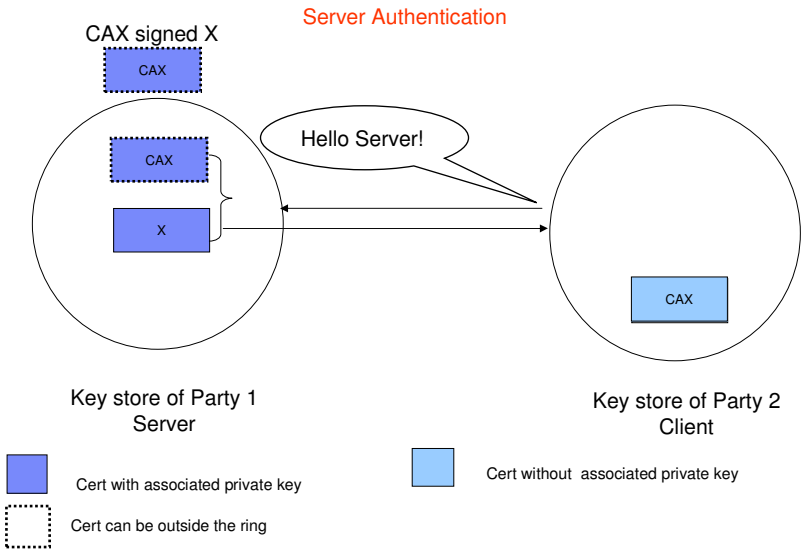
* Note the above steps illustrate server authentication. For client authentication, server needs to validate client's certificate too.



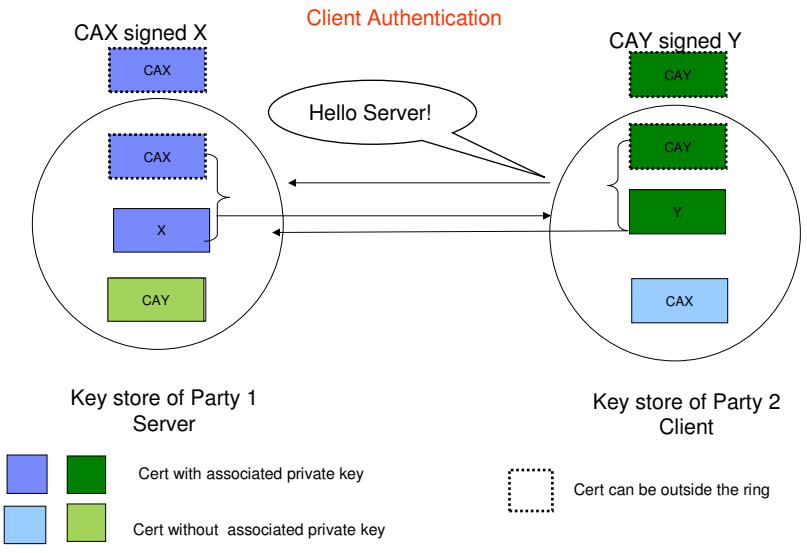
Remember the simple rule – PVC(1 of 3)

- **PVC - Parent Validates Child**
  - Child<-Parent<-Grandparent<-Great Grandparent<-…. <-Great Great….<-Root Grandparent
    - Ensure the content of the whole certificate chain has not been altered
      - Signature on the child verified by parent's public key
      - Signature on the root verified by its own public key
    - Trusting the Root Grandparent
      - Putting the root in the key store is the indication of trusting all its descendents

Remember the simple rule – PVC(2 of 3)



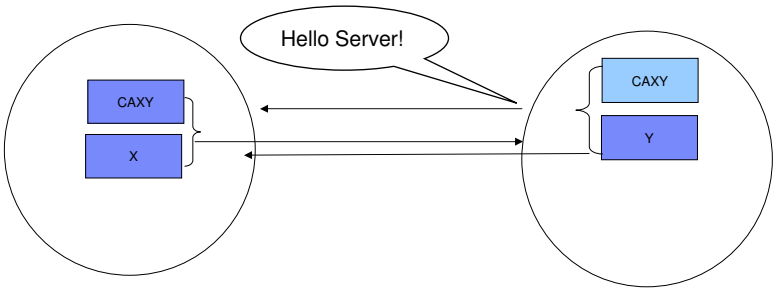
Remember the simple rule – PVC(3 of 3)



Simplify the set up

Client Authentication

CAXY signed X and Y



Key store of Party 1
Server

Key store of Party 2
Client

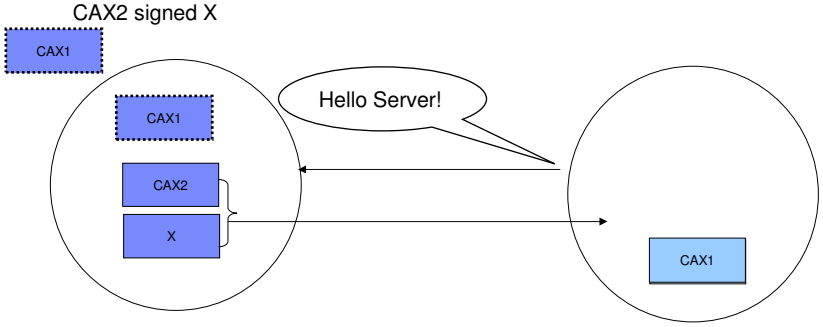
Cert with associated private key

Cert without associated private key

Similar set up in the Chain scenario

Server Authentication

CAX1 signed CAX2
CAX2 signed X



Key store of Party 1
Server

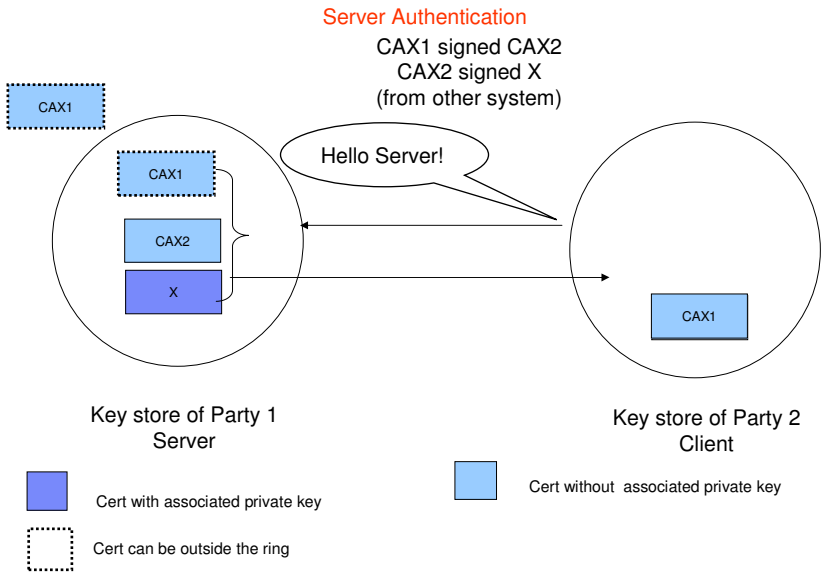
Key store of Party 2
Client

Cert with associated private key

Cert can be outside the ring

Cert without associated private key

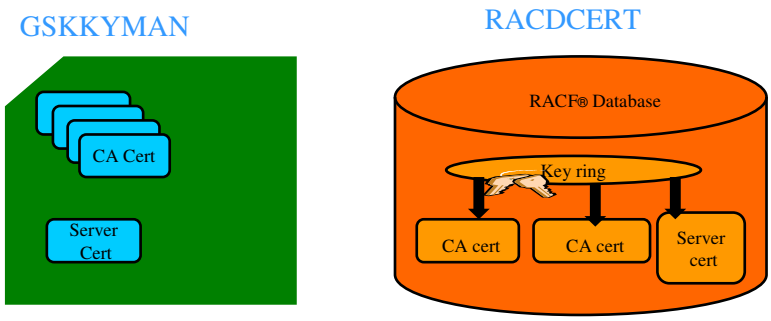
Third Party CAs scenario



Part 2 - Overview of certificate utilities available on z/OS

Certificate Stores on z/OS

- gskkyman manages certificates stored in a key database file
- RACDCERT manages certificates stored in a RACF key ring.



Certificate Store Protection

- gskkyman key database files
  - Protected by the file system's permission bits and password
  - Upon creation, permission bits are 700 giving the issuer of gskkyman read and write to the file only.
  - Applications using these files need at least read to the file
- RACF Key Rings
  - RACF key rings are protected by resource profiles.
  - Users rings need read access to IRR.DIGTCERT.LISTRING or <ring owner>.<ring name>.LST to be able to read the contents of their key ring
    - IRR.DIGTCERT.LISTRING – Global control
    - <ring owner>.<ring name>.LST – Granular control

Certificate Utilities

- gskkyman is a Unix based utility shipped as part of the System SSL product in the z/OS Cryptographic Services Element
- RACDCERT is a TSO command shipped as part of RACF
- Provide basic certificate functions
  - Create/delete certificate store (HFS key database file / SAF key ring)
  - Create certificate requests (to be signed by trusted Certificate Authority)
  - Import/Export certificates (with and without private keys)
  - Create self-signed certificates
- Do not have all the functions of a real Certificate Authority

Certificate Authority on z/OS

- PKI Services provides full certificate life cycle management
  - Request, create, renew, revoke certificate
  - Provide certificate status through Certificate Revocation List(CRL) and Online Certificate Status Protocol (OCSP)
  - Generation and administration of certificates via customizable web pages
  - Support Simple Certificate Enrollment Protocol (SCEP) for routers to request certificates automatically
  - Automatic notification or renewal of expiring certificates

Defining a Certificate

- **How will the certificate be used?**
- **What certificate store is to be used?**
- **What is the size of the public/private keys?**
- **What subject name to use?**
- **Need additional identity information and extensions?**
- **Validity period of the certificate?**
- **Who will be the certificate authority?**

Defining a Certificate Request to be signed by a CA

- A **certificate signing request** (also **CSR**) is a message sent from the certificate requestor to a certificate authority to obtain a signed digital certificate
- Contains identifying information and public key for the requestor
- Corresponding private key is not included in the CSR, but is used to digitally sign the request to ensure the request is actually coming from the requestor
- CSR may be accompanied by other credentials or proofs of identity required by the certificate authority, and the certificate authority may contact the requestor for further information.
- If the request is successful, the certificate authority will send back an identity certificate that has been digitally signed with the private key of the certificate authority.

If you use gskkyman…

Create a key database

Database Menu

1 - Create new key database
2 - Open key database
3 - Change database password
4 - Change database record length
5 - Delete database
6 - Create key parameter file
7 – Display certificate file (Binary or Base64 ASN.1 DER)

0 - Exit Program

Name of key database

Enter your option number: 1
Enter key database name (press ENTER to return to menu: /tmp/my.kdb
Enter database password (press ENTER to return to menu: password
Re-enter database password: password
Enter password expiration in days (press ENTER for no expiration): <enter>
Enter database record length (press ENTER to use 2500): <enter>

This will add a number of well-known trusted CA certificates to the key database.

Importing a signing Certificate Authority Certificate(1 of 2)

Key Management Menu

Database: /tmp/my.kdb

1 - Manage keys and certificates
2 - Manage certificates
3 - Manage certificate requests
4 - Create new certificate request
5 - Receive requested certificate or a renewal certificate
6 - Create a self-signed certificate
7 - Import a certificate
8 - Import a certificate and a private key
9 - Show the default key
10 - Store database password
11 - Show database record length

0 - Exit program

Enter option number (press ENTER to return to previous menu): 7

Importing a signing Certificate Authority Certificate(2 of 2)

File contains the CA certificate

Enter import file name (press ENTER to return to menu): cacert.b64
Enter label (press ENTER to return to menu): CA Certificate

Certificate imported.

Creating a new certificate request

```
Key Management Menu

Database: /tmp/my.kdb

1 - Manage keys and certificates
2 - Manage certificates
3 - Manage certificate requests
4 - Create new certificate request
5 - Receive requested certificate or a renewal certificate
6 - Create a self-signed certificate
7 - Import a certificate
8 - Import a certificate and a private key
9 - Show the default key
10 - Store database password
11 - Show database record length

0 - Exit program

Enter option number (press ENTER to return to previous menu): 4
```


Fill in the information about the requestor (1of 2)

```
Certificate Key Algorithm
1 - Certificate with an RSA key
2 - Certificate with a DSA key
3 - Certificate with an ECC key
Select certificate key algorithm (press ENTER to return to menu): 1

RSA Key Size
1 - 1024-bit key
2 - 2048-bit key
3 - 4096-bit key
Select RSA key size (press ENTER to return to menu): 2

Signature Digest Type
1 - SHA-1
2 - SHA-224
3 - SHA-256
4 - SHA-384
5 - SHA-512
Select digest type (press ENTER to return to menu): 2
```

Fill in the information about the requestor(2 of 2)

File to contain certificate request

Enter request file name (press ENTER to return to menu): certreq.arm
Enter label (press ENTER to return to menu): Server Certificate
Enter subject name for certificate
Common name (required): Server Certificate
Organizational unit (optional): Production
Organization (required): IBM
City/Locality (optional): Endicott
State/Province (optional): New York
Country/Region (2 characters - required): US

Enter 1 to specify subject alternate names or 0 to continue: 1

Receiving a signed certificate request

Key Management Menu

Database: /tmp/my.kdb

1 - Manage keys and certificates
2 - Manage certificates
3 - Manage certificate requests
4 - Create new certificate request
5 - Receive requested certificate or a renewal certificate
6 - Create a self-signed certificate
7 - Import a certificate
8 - Import a certificate and a private key
9 - Show the default key
10 - Store database password
11 - Show database record length

0 - Exit program

File contains cert returned from CA

Enter option number (press ENTER to return to previous menu): 5
Enter certificate file name (press ENTER to return to menu): svrcert.arm

Marking a certificate as the default

Key and Certificate Menu

Label: Server Certificate

1 - Show certificate information
2 - Show key information
3 - Set key as default
4 - Set certificate trust status
5 - Copy certificate and key to another database
6 - Export certificate to a file
7 - Export certificate and key to a file
8 - Delete certificate and key
9 - Change label
10 - Create a signed certificate and key
11 - Create a certificate renewal request

0 - Exit program

Enter option number (press ENTER to return to previous menu): 3

If you use RACDCERT…
(ISPF Panel or Command)

RACDCERT deals with 3 object types

RACDCERT functions(1 of 2)

- **Certificate generation**
  - RACDCERT GENCERT – generate key pair and certificate
  - RACDCERT GENREQ – generate a certificate request
- **Certificate installation**
  - RACDCERT ADD – install a certificate and public/private key
- **Certificate administration**
  - RACDCERT ADDRING – create a key ring
  - RACDCERT CONNECT – place a certificate in a key ring
  - RACDCERT REMOVE – remove a certificate from a key ring
  - RACDCERT LISTRING – display key ring information
  - RACDCERT DELRING – delete a key ring
  - RACDCERT LIST – display certificate information from an installed certificate
  - RACDCERT ALTER – change certificate installation information
  - RACDCERT DELETE – delete certificate and key pair
  - RACDCERT CHECKCERT – display certificate information from a dataset
  - RACDCERT EXPORT – export a certificate

RACDCERT functions(2 of 2)

- **Certificate administration…**
  - RACDCERT MAP – create a certificate filter
  - RACDCERT ALTMAP – change the certificate filter
  - RACDCERT DELMAP – delete a certificate filter
  - RACDCERT LISTMAP – display certificate filter information
  - RACDCERT REKEY – renew certificate with new key pair
  - RACDCERT ROLLOVER – finalize the REKEY process
- **… more**

RACDCERT Panel on Key Ring

```
                RACF - Digital Certificate Key Ring Services
OPTION ===> _

   For user: ________

Enter one of the following at the OPTION line:

    1   Create a new key ring
    2   Delete an existing key ring
    3   List existing key ring(s)
    4   Connect a digital certificate to a key ring
    5   Remove a digital certificate from a key ring
```

RACDCERT Panel on Certificate

```
            RACF - Digital Certificate Services
OPTION ===>

  Select one of the following:

        1. Generate a certificate and a public/private key pair.

        2. Create a certificate request.

        3. Write a certificate to a data set.

        4. Add, Alter, Delete, or List certificates or
           check whether a digital certificate has been added to
           the RACF database and associated with a user ID.

        5. Renew, Rekey, or Rollover a certificate.
```


Create a key ring

Name of key ring

RACDCERT ID(FTPserver) ADDRING(MyRACFKeyRing)

Adding Certificate Authority(CA) Certificate to a key ring

Dataset contains the CA certificate

RACDCERT CERTAUTH ADD('user1.cacert') TRUST WITHLABEL('CA Certificate')

RACDCERT ID(FTPServer) CONNECT (CERTAUTH LABEL('CA Certificate') RING(MyRACFKeyRing) USAGE(CERTAUTH))

Creating a new certificate request

RACDCERT ID(FTPServer) GENCERT SUBJECTSDN(CN('Server Certificate')OU('Production')O('IBM')L('Endicott')SP('New York')C('US'))
SIZE(1024) WITHLABEL('Server Certificate')
ALTNAME(DOMAIN('mycompany.com'))

RACDCERT ID(FTPServer) GENREQ(LABEL('Server Certificate'))
DSN('user1.certreq')

Dataset to contain certificate request

Adding Certificate signed by CA to a key ring

RACDCERT ID(FTPServer) ADD('user1.svrcert')
WITHLABEL('Server Certificate')

Dataset contains cert returned from CA

RACDCERT ID(FTPServer) CONNECT(ID(SUIMGTF)
LABEL('Server Certificate') RING(MyRACFKeyRing)
USAGE(PERSONAL) DEFAULT)

Listing a RACF Key Ring

RACDCERT ID(FTPServer) LISTING(MyRACFKeyRing)

```
Ring:
     >MyRACFKeyRing<
  Certificate Label Name          Cert Owner      USAGE      DEFAULT
  ------------------------------  --------------  ---------  -------
  CA Certificate                  CERTAUTH        CERTAUTH   NO
  Server Certificate              ID(FTPServer)   PERSONAL   YES
```

Note: RACF key rings allow for a certificate's private key to be stored into ICSF's (Integrated Cryptographic Service Facility) PKDS (Public Key Dataset) for added security.

Certificate Formats

- **X.509 certificates can exist in many different forms**
  - Single certificate
  - PKCS #7 certificate package
    - Contains 1 or more certificates
  - PKCS #12 certificate package
    - A password encrypted package containing 1 or more certificates and the private key associated with the end-entity certificate.
    - Only package type that contains a private key
- **Can be in binary or Base64 encoded format**

Base64 encoding

- **Converting binary data to displayable text for easy cut and paste.**

```
-----BEGIN CERTIFICATE-----
MIICPTCCAaagAwIBAgIIR49S4QANLvEwDQYJKoZIhvcNAQEFBQAwNzELMAkGA1UE
BhMCVVMxDTALBgNVBAoTBFRlc3QxGTAXBgNVBAMMEFRlc3Rfc2VsZl9zaWduZWQw
HhcNMDgwMTE3MTMwNjQxWhcNMDkwMTE2MTMwNjQxWjA3MQswCQYDVQQGEwJVUzEN
MAsGA1UEChMEVGVzdDEZMBcGA1UEAwwQVGVzdF9zZWxmX3NpZ25lZDCBnzANBgkq
hkiG9w0BAQEFAAOBjQAwgYkCgYEA9tKOv5gLaceozMfMeVd891fCjBVoR+dpzhwK
R2B/QcQYBGLfqS4YM/wGSh6YrmVygO0VxocriySbcxRuBayw3pE4/3JI2myINmLp
bFIdPCnqk/qvFK+1N+nrEnBK9yls7NmxDIuQQfFsX/o/DpoxwzwXf+JbWDwirQR
NyLiTGMCAwEAAaNSMFAwHQYDVR0OBBYEFAwDFLjOUCRa62BVs3jVyHewuOWEMB8G
A1UdIwQYMBaAFAwDFLjOUCRa62BVs3jVyHewuOWEMA4GA1UdDwEB/wQEAwIE8DAN
BgkqhkiG9w0BAQUFAAOBgQAC5sW1f3EdE0k9zc8wKNt1sczWkQBrVy4Rdrl7ERqN
D2OfkBJQuXiNwN18pF6WPWfYG80MNwhP4oJSVePnzElh4Wzi2wl/zI8rINSW7px3
w16lz+8jEI84q/N0q0toPTAtEb6fIzwjkLtctt3oF+IjunvE5QoRsXRJbbTMD/EG
jw==
-----END CERTIFICATE-----
```


Exporting Certificates through gskkyman(1 of 2)

Key and Certificate Menu

Label: Server Certificate

1 - Show certificate information
2 - Show key information
3 - Set key as default
4 - Set certificate trust status
5 - Copy certificate and key to another database
6 - Export certificate to a file
7 - Export certificate and key to a file
8 - Delete certificate and key
9 - Change label
10 - Create a signed certificate and key
11 - Create a certificate renewal request

0 - Exit program

Enter option number (press ENTER to return to previous menu):

Exporting Certificates through gskkyman(2 of 2)

Option 6 – Public Certificate Information

Export File Format

1 - Binary ASN.1 DER
2 - Base64 ASN.1 DER
3 - Binary PKCS #7
4 - Base64 PKCS #7

Option 7 – Public Certificate Information and Private Key

Export File Format

1 - Binary PKCS #12 Version 1 (Few very old applications still use V1)
2 - Base64 PKCS #12 Version 1
3 - Binary PKCS #12 Version 3
4 - Base64 PKCS #12 Version 3

Exporting Certificates through RACDCERT(1 of 2)

- **RACDCERT ID(userid) EXPORT**
  (LABEL('label-name'))
  DSN(output-data-set-name)
  FORMAT(CERTDER | CERTB64 | PKCS7DER | PKCS7B64 | PKCS12DER | PKCS12B64)
  PASSWORD('pkcs12-password')

- **Example - Export Server Certificate with its private key**
  - RACDCERT ID(FTPServer) EXPORT LABEL('Server Certificate') DSN('USER1.SERVER.CERT') FORMAT(PKCS12DER) PASSWORD('passwd')

Exporting Certificates through RACDCERT(2 of 2)

- **Precaution needed for CERTAUTH certificate when you plan to preserve the certificate and the private key by exporting them in a pkcs12 package**
  - If the original CERTAUTH certificate got deleted and you re-add this package, the field that used for recording serial numbers that it has issued is not reserved
  - For example, if this CA certificate has issued 100 certificates, the next certificate to be issued should have serial number 101; but after re-adding it, the certificate to be issued will have serial number 1, which is already used – all the certificates issued by the same CA should have a unique serial number!
- **Before deleting CERTAUTH certificate, find out the last certificate's serial number it issued**
- **After re-adding, use r_datalib to bump up the serial number field to the appropriate number**

Summary(1 of 2)

- **Digital certificates provide electronic identity and public key information to be utilized through public key protocols (ie. SSL/TLS)**
- **Utilizing trusted CAs is key to ensure validity of the digital certificate**
- **Protect the private key!!!**
- **Larger the public/private key pair size, greater security, but more computation intense**

Summary (2 of 2)

- **When transferring certificates, use a format acceptable to the receiving side.**
- **When transferring certificates, be sensitive to binary and text modes to ensure proper transfer**

Part 3 - RACDCERT in depth

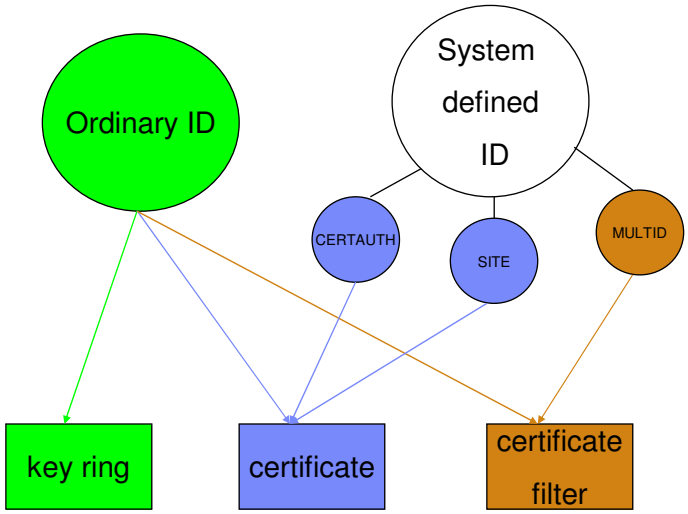
Top 10 RACDCERT questions(1 of 2)

10. The person who created the certificate left the company. When I delete his ID, will all the certificates he created still be OK?
9. How can I change the 'owner' field in the certificate profile?
8. Why can't I manage the certificate through the RALTER, RDELTE… commands?
7. How can I generate a request using RACDCERT?
6. I requested a certificate from a commercial CA. I tried to RACDCERT ADD it, I couldn't since the issuer's name is too long. What's going on?
5. I cut and paste a simple RACDCERT GENCERT command to generate a self-signed certificate from my colleague in the same system. The command works for him but not for me. Why?
4. I try to renew a self-signed certificate. I got error message saying 'No certificate was found'. Why?

Top 10 RACDCERT questions(2 of 2)

3. I connect a certificate to a key ring. I am sure the certificate and the ring are there. But I got message 'No certificate was found'. Why?
2. I used RACDCERT to generate a request and submit it to the external CA. When I get back the certificate from CA and install it, the FTP server program does not work, complaining about no private key found. Why?
1. Whose certificates need to be installed in whose key ring for FTP to work?

Relationship between entities and owner IDs



Relationship between Certificate and Key ring

- Certificate must be placed in a key ring before it can be used by other middleware through R_datalib
- Three types of certificates in a ring that the middleware can utilize:
  - **Personal certificate (for identification)**
    - Under ordinary MVS ID or with USAGE PERSONAL
    - Its private key is also known to RACF
    - Sent to the client when SSL is initiated
  - **Certificate Authority certificate (for validation)**
    - Under CERTAUTH ID or with USAGE CERTAUTH
    - Its private key is not known to RACF
    - Used to authenticate the incoming certificate
  - **SITE certificate (for identification)**
    - Under SITE ID or with USAGE SITE
    - Similar to personal certificate
    - But its private key can be shared (usual way to share key before V1R9)
  - **SITE certificate (for validation)**
    - Under SITE ID or with USAGE SITE
    - There may be a few certificate validation applications which treat a certificate that is connected to a key ring with usage site as a valid certificate authority certificate to bypass the normal certificate verification tests during SSL handshake, for example, an expired certificate can be considered trusted

Certificate stored as a profile(1 of 2)

- **A certificate profile in the DIGTCERT class is created for a certificate added or created**
  - The profile name is of the form
    <cert serial #>.<issuer's distinguished name>
  - Examples:
    Command:

```
RACDCERT CERTAUTH GENCERT SUBJECTDN(OU('Master CA') O('IBM') C('US'))
WITHLABEL('MyCA')
```

    Profile created: **00.OU=Master¢CA.O=IBM.C=US**

    Command:

```
RACDCERT ID(testid) GENCERT SUBJECTDN(OU('Test Dept') O('IBM) C('US'))
WITHLABEL('TestCert') SIGNWITH(CERTAUTH LABEL('MyCA'))
```

    Profile created: **01.OU=Master¢CA.O=IBM.C=US**
  - Serial number of a self-signed certificate is 0
  - The subsequent serial numbers will be incremented in the order of 1
  - The blanks in the distinguished name are substituted with '¢' in the profile (¢ is not a 7 bit ASCII character)
  - If the Issuer's name is long, instead of using the name directly to form the profile name, a hash of the name will be used.

Now you should be able to answer …

John cut and pasted a simple RACDCERT GENCERT command to generate a self-signed certificate from his colleague, Mary, in the same system. The command works for Mary but not for John. Why?

- **1st, Mary issued: RACDCERT ID(Mary) GENCERT SUBJECT(CN('XYZ')) WITHLABEL('Marycert') - OK**
- **2nd, John issued: RACDCERT ID(John) GENCERT SUBJECT(CN('XYZ')) WITHLABEL('Johncert') – Error**

Certificate stored as a profile (2 of 2)

- **This profile represents the certificate. NOT a protection profile!**
  - The owner field in this profile indicates the issuer of the RACDCERT command, NOT the certificate owner
  - The certificate profile can NOT be managed through the resources management commands, like RALTER, RDELETE…
  - Managed through RACDCERT commands
- **There are function specific profiles in the facility class for authority checking**
  - Read, Update or Control on IRR.DIGTCERT.<function>
    - `Eg. IRR.DIGTCERT.GENCERT, IRR.DIGTCERT.ADD`

GENREQ needs GENCERT

- **RACDCERT GENCERT without specifying SIGNWITH generates a self-signed certificate**
  - **RACDCERT GENCERT SUBJECTSDN(CN('mycert') OU('RACF')…)**
- **Need 2 RACDCERT commands to generate a request**
  - **RACDCERT GENCERT (usually a self-signed one)**
    - **This is a stepping stone to get the request, will be replaced once the certificate is returned**
    - **RACDCERT ID(ftpd) GENCERT SUBJECTSDN(CN('ftpcert') OU('RACF')…) WITHLABEL('ftpcert')**
  - **RACDCERT GENREQ <use the certificate label from GENCERT above >**
    - **RACDCERT ID(ftpd) GENREQ(LABEL('ftpcert')) DSN('user1.ftpreq')**
  - **Send the request to external CA for signing**
  - **When the certificate is returned from the external CA, install it in RACF**

RACDCERT ID(1 of 2)

- If no ID type is specified, the ID of the user issuing the command is used
  - User1's certificate is displayed if user1 issues the following command
    - `RACDCERT LIST(LABEL('cert1'))`
  - User2's certificate is displayed if user1 issues the following command (assuming user1 has the authority to list other's certificate)
    - `RACDCERT ID(user2) LIST(LABEL('cert2'))`

Now you should be able to answer …

I try to renew a self-signed certificate. I got error message saying '*IRRD107I No matching certificate was found for this user*'. Why?

- User1 issues the following command to renew a self-signed certificate 'FTPCert' owned by FTPID:
  - `RACDCERT ID(FTPID) GENCERT(<request based on the original self-signed FTPCert> SIGNWITH(LABEL('FTPCert'))`

RACDCERT ID (2 of 2)

- There is an exception in the CONNECT command which involves 2 entities, ring and cert.
- Syntax: RACDCERT <ring owner id> CONNECT(<cert owner id> <cert label>…)
- Which case has the exception?
  - `RACDCERT ID(Mary) CONNECT(ID(John) LABEL…)`
    - Ring owner: Mary, Cert owner: John
  - `RACDCERT ID(Mary) CONNECT(LABEL…)`
    - Ring owner: Mary, Cert owner: Mary
  - `RACDCERT CONNECT(ID(John) LABEL…)`
    - Ring owner: Issuer of command, Cert owner: John
  - `RACDCERT CONNECT(LABEL…)`
    - Ring owner: Issuer of command, Cert owner: Issuer of command

Now you should be able to answer …

I connect a certificate to a key ring. I am sure the certificate and the ring are there. But I got message '*IRRD107I No matching certificate was found for this user*'. Why?

- `The following commands are issued by user1:`
  - `RACDCERT ID(FTPID)LISTRING(ring1)- found it`
  - `RACDCERT LIST(LABEL('mycert'))- found it`
  - `RACDCERT ID(FTPID) CONNECT(LABEL('mycert')ring(ring1) DEFAULT) -error`

When is the key pair generated?

- ✓ **RACDCERT GENCERT with NO request input**
  - **Examples:**
    - **RACDCERT GENCERT SUBJECTSDN(CN('mycert') OU('RACF')…)**
    - **RACDCERT GENCERT SUBJECTSDN(CN('mycert') OU('RACF')…) SIGNWITH (<some CA certificate in RACF>)**
  - **Generates:**
    - **Key pairs – public key and private key**
    - **Certificate**
  - **Private key can be stored in RACF or ICSF**
  - **Public key is put on the certificate**
- × **RACDCERT GENCERT with the input of a request (RACF as the CA to sign request from another system)**
  - **Example:**
    - **RACDCERT GENCERT(<request from other source>) SIGNWITH(<some CA certificate in RACF>**
  - **Generates only**
    - **Certificate**
  - **Public key is put on the certificate**
  - **Must specify the SIGNWITH keyword**

Now you should be able to answer …

I used RACDCERT to generate a request and submit it to the external CA. When I get back the certificate from CA and install it, the FTP server program does not work, complaining about no private key found. Why?

- **Here are the steps:**
  - RACDCERT ID(ftpid) GENCERT …WITHLABEL('temp cert')
  - RACDCERT ID(ftpid) GENREQ(LABEL('temp cert')) DSN(<output request_dsn>)
  - RACDCERT ID(ftpid) DELETE(LABEL('temp cert'))
  - Send the request to external CA
  - Get back the certificate
  - RACDCERT ID(ftpid) ADD (<dataset contains the returned cert>) WITHLABEL('real cert')
  - RACDCERT ID(ftpid) CONNECT (LABEL('real cert') RING(<ftpring>)…DEFAULT)

Certificate Name Filtering and Host ID Mapping

A certificate represents a RACF user

- **User can be identified to RACF through certificate if his certificate is in the RACF DB**
  - One-to-one certificate to user ID association
- **If there are thousands of users, thousands of certificates need to be installed…**

More Sophisticated Certificate Support from RACF (1 of 7)

RACF provides other types of certificate and ID associations which require no user certificate to be installed:

- **Solution 1: Certificate Name Filtering**
- **Solution 2: HostIdMapping**

More Sophisticated Certificate Support from RACF (2 of 7)

- **Certificate Name Filtering – RACDCERT MAP**
  - Create a filter based on a set of rules ('filters') on the **subject's** or **issuer's** distinguished names (or **both**)
    OU=…O=…C=US CN=…OU=…O=…C=US
  - The owning ID of the filter should be PROTECTED and RESTRICTED
  - Need to raclist DIGTNMAP class
  - Search sequence:
    1. subject's-full-name.issuer's-full-name
       OU=…O=…C=US.CN=…OU=…O=…C=US
    2. subject's-partial-name.issuer's-full-name
       O=…C=US.CN=…OU=…O=…C=US
    3. subject's-full-name
       OU=…O=…C=US
    4. subject's-partial-name
       O=…C=US
    5. issuer's-full-name
       CN=…OU=…O=…C=US
    6. issuer's-partial-name
       OU=…O=…C=US

More Sophisticated Certificate Support from RACF (3 of 7)

- Can map one or more certificates to a filter => allow multiple user to share the same ID
- **Examples:**
  - **Create a filter to associate ID VUSER to any user presenting a certificate issued by VeriSign Class 1 Individual Subscriber**

```
RACDCERT ID(VUSER) MAP IDNFILTER('OU=VeriSign Class 1
Individual Subscriber.O=VeriSign, Inc.L=Internet')…
```

  - **Create a filter to associate ID RACFGP to any user presenting a certificate with subject's distinguished name OU=RACF.O=IBM**

```
RACDCERT ID(RACFGP)MAP SDNFILTER ('OU=RACF.O=IBM')…
```


More Sophisticated Certificate Support from RACF (4 of 7)

- Will cause losing some degree of granularity in access control. As shown in the above examples, all the users are given the authorizations of ID vuser, racfgp.
- Still retain full auditing accountability because subject's and issuer's distinguished names in the certificate will be in the audit record.

More Sophisticated Certificate Support from RACF (5 of 7)

- Can also mapped to different IDs based on system and application criteria, eg.
  - The user of the certificate needs access to more than one application, and each application requires a different user ID.
  - The same application might run on more than one system, and each system requires a different user ID.
- The filter is not associated with an ID directly, but through a profile in the DIGTCRIT class

More Sophisticated Certificate Support from RACF (6 of 7)

- **Example:**
  - **Create a filter to associate to any user presenting a certificate issued by VeriSign Class 1 Individual Subscriber using ID1 if the certificate is passed through application APP1; using ID2 if the application is APP2**

```
RACDCERT MULTIID MAP IDNFILTER(('OU=VeriSign Class 1 Individual
Subscriber.O=VeriSign, Inc.L=Internet') CRITERIA(APPLID=&APPLID)…
```

Assuming these profiles also created:

```
RDEFINE DIGTCRIT APPLID=APP1 APPLDATA(ID1)
RDEFINE DIGTCRIT APPLID=APP2 APPLDATA(ID2)
```

- **Certificate Name Filtering will be used only if the certificate is not installed in RACF**

More Sophisticated Certificate Support from RACF (7 of 7)

- **Host ID Mapping**
  - A client can present a certificate containing a HostIdMapping extension to the server
  - This extension contains a subject id and a host name, eg. user1@abc.com
  - RACF will honor this extension if
    - the issuing CA cert is marked HIGHTRUST
    - the host name in the extension matches a profile IRR.HOST.<host name> in the SERVAUTH class
    - The presenter of the cert has access to the above profile, eg. IRR.HOST.abc.com
  - The subject id, eg. user1 will then be used to access the resource
  - Host ID Mapping will be used only if the certificate is not installed in RACF AND there is no certificate name filter
  - RACDCERT can't create this extension, PKI Services can

Renewal and Sharing in RACDCERT

Two ways to renew a certificate(1 of 4)

Eventually a certificate will expire. To avoid complications, you should renew it before it expires.

- **Renew a certificate with the original key pair**
  - **If the certificate is a self-signed certificate:**
    1. Create a new certificate request from the original certificate and save the request in a dataset 'request_dsn':

```
RACDCERT CERTAUTH GENREQ(LABEL('original cert'))
 DSN(request_dsn)
```

    2. Create the new certifcate using the request in step 1:

```
RACDCERT CERTAUTH GENCERT(request_dsn) SIGNWITH(CERTAUTH
 LABEL('original cert'))
```

  - **If the certificate is not a self-signed certificate:**
    1. Same as step 1 above
    2. Send the request to the original certificate CA
    3. After you receive the new certificate and save it in a dataset 'cert_dsn', add it back under the same ID:

```
RACDCERT CERTAUTH ADD(cert_dsn)
```

Note: Don't delete the 'original cert'!!!

Two ways to renew a certificate (2 of 4)

- **Renew a certificate with a new key pair**

The longer a key pair is used, the more likely it is to be cracked. The key pair should be periodically changed. Two RACDCERT 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

- ❖Private key of the old certificate is deleted so that it may not be used again for signing or encryption
- ❖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

Two ways to renew a certificate (3 of 4)

- **Renew a certificate with a new key pair…**
  - **If the certificate is a self-signed certificate:**
    1. Make a self copy of the original certificate:

```
RACDCERT CERTAUTH REKEY(LABEL('original
cert'))WITHLABEL('original cert2')
```

    2. Roll over the original certificate to the new one:

```
RACDCERT CERTAUTH ROLLOVER(LABEL('original cert'))
NEWLABEL('original cert2')
```


Two ways to renew a certificate (4 of 4)

- **Renew a certificate with a new key pair…**
  - **If the certificate is not a self-signed certificate:**
    1. Make a self copy of the original certificate

```
RACDCERT ID(myid) REKEY(LABEL('original cert'))
WITHLABEL('original cert2')
```

    2. Create a certificate request from the copied certificate in step 1:

```
RACDCERT ID(myid) GENREQ(LABEL('original
cert2')) DSN(request_dsn)
```

    3. Send the request to the original certificate CA
    4. After you receive the new certificate and save it in a dataset 'cert_dsn', add it back under the same ID:

```
RACDCERT ID(myid) ADD(cert_dsn)
```

    5. Roll over the original certificate to the new one:

```
RACDCERT ID(myid) ROLLOVER(LABEL('original
cert')) NEWLABEL('original cert2')
```

Share keyring, certificate, private key?

Share keyring ===>Share certificate===>Share private key

- You may share a keyring and hence its content (certificates) amongst different IDs by giving permission to IRR.DIGTCERT.LISTRING
  - eg. Set up client A, B, C…for server authentication *(remember the quiz: no private key is involved in the client side)*
- Sharing private key is not recommended, but in case you really want to …eg. Avoid buying a separate certificate for another server or client, there is a way

Share just the certificates between multiple clients for server authentication

- Create a keyring under one ID, say CLN1
  - `RACDCERT ID(CLN1) ADDRING(CommonRing)`
- Connect all the CA certificates to this ring
  - `RACDCERT ID(CLN1) CONNECT(CERTAUTH LABEL('VeriSign Cert' RING(CommonRing))`
  - `RACDCERT ID(CLN1) CONNECT(CERTAUTH LABEL('GeoTrust Cert' RING(CommonRing))`
  - …
- Permit both IDs to use this ring
  - `PERMIT IRR.DIGTCERT.LISTRING CLASS(FACILITY) ACCESS(READ) ID(CLN1)`
  - `PERMIT IRR.DIGTCERT.LISTRING CLASS(FACILITY) ACCESS(UPDATE) ID(CLN2)`
- If you don't want to share, you need to create a separate ring and connect the same CA certificates for CLN2

V1R8 provides a better solution – Virtual Key Ring

- All the certificates under a RACF user ID are considered 'connected' to a virtual key ring automatically
- Eliminate the work to create key rings and connect certificates to them for all the clients which use the same set of certificates for validation
- This solution is useful for client side SSL applications that don't do client authentication, for example, multiple FTP clients talking to the same server
- Example: in FTP.DATA
  - KEYRING *AUTH*/*

Share a private key between ID SRV1 and SRV2(1 of 2)

- Create a keyring under one ID, say SRV1
  - `RACDCERT ID(SRV1) ADDRING(ShareRing)`
- Create a certificate under CERTAUTH or SITE, not a personal ID
  - `RACDCERT SITE GENCERT... WITHLABEL('Share Cert')`
- Connect the cert to this ring
  - `RACDCERT ID(SRV1) CONNECT(SITE LABEL('Share Cert') RING(ShareRing) USAGE(PERSONAL) DEFAULT)`
- Permit both IDs to use this ring
  - `PERMIT IRR.DIGTCERT.LISTRING CLASS(FACILITY) ACCESS(READ) ID(SRV1)`
  - `PERMIT IRR.DIGTCERT.LISTRING CLASS(FACILITY) ACCESS(UPDATE) ID(SRV2)`

Share a private key between ID SRV1 and SRV2 (2 of 2)

- Permit both IDs to use this private key
  - `RDEF FACILITY IRR.DIGTCERT.GENCERT UACC(NONE)`
  - `PERMIT IRR.DIGTCERT.GENCERT CLASS(FACILITY) ACCESS(CONTROL) ID(SRV1 SRV2)`

*!!!Note: When you share private key as described above, you are allowing the two IDs to access **any** private keys that are stored under SITE or CERTAUTH.*

- There is a better solution in V1R9 that will be discussed later

V1R9 provides another solution – Granular access control on Key Ring

- Access is based on a profile of a specific key ring in a new class called RDATALIB
- The class RDATALIB must be RACLISTed
- A resource with the format <ringOwner>.<ringName>.LST is used to provide access control to a specific key ring on R_datalib READ functions
- This new support also allows the retrieval of another person's private key
- So instead of giving access to all the rings
  - PERMIT IRR.DIGTCERT.LISTRING CLASS(FACILITY) ACCESS(UPDATE) ID(CLN2)
- Just give access to that particular ring
  - PERMIT CLN1.COMMONRING.LST CLASS(RDATALIB) ACCESS(READ) ID(CLN2)
- If you want to share the private key, then
  - PERMIT CLN1.COMMONRING.LST CLASS(RDATALIB) ACCESS(UPDATE) ID(CLN2)

Other enhancements (1 of 2)

V1R11:

- Enable RACDCERT to generate certificate with supplied public key that was already stored in PKDS.
- Allow adding to RACF a certificate with multi byte UTF8 characters in the subject distinguished name, even they are outside the IBM-1047 code page

V1R12

- Support Elliptic Cryptographic Curve (ECC) keys stored in RACF, in addition to RSA and DSA keys
- Support longer Distinguished Name beyond 246 characters (roll back to R10 and R11)
- Support adding and creating certificates with validity dates beyond year 2041(roll back to R10 and R11)

Other enhancements (2 of 2)

V1R13

- Support secure Elliptic Cryptographic Curve (ECC) keys stored in ICSF

Part 4 – Some hot topics on certificates

- Example
- Build or Buy
- Outage caused by expired certificate

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) Note1	FTP.DATA file KEYRING MYRING or AT-TLS policy
TN3270 Server	RACDCERT ID(TNSVR) CONNECT(LABEL('TN Cert') RING(MYRING) DEFAULT) Note1	Telnet profile file KEYRING SAF MYRING or AT-TLS policy
IP Security (IPSEC)	RACDCERT ID(IPSEC) CONNECT(LABEL('IPSEC Cert') RING(MYRING) DEFAULT) *Note1*	Iked.conf file KEYRING MYRING or AT-TLS policy
HTTP Server	RACDCERT ID(WEBSVR) CONNECT(LABEL('WEB Cert') RING(MYRING) DEFAULT) *Note: must be connected as default*	httpd.conf file Keyfile MYRING SAF
Websphere MQ	RACDCERT ID(QM1) CONNECT(LABEL ('ibmWebSphereMQMQ1') RING(MYRING)) *Note: label of the cert must start with 'ibmWebSphereMQ'*	MQ command ALTER QMGR SSLKEYR (MYRING)

Note1: cert connected as default or use a specified label indicated in AT-TLS policy

FTP Server authentication

- Scenario
  - My business partner runs a secure FTP server on Windows. I need to send files from z/OS to it daily.
- Set up
  - If the partner's root CA certificate of the FTP server certificate is already in your RACF database, eg. It is one of the default well-known CA certificates shipped with RACF
    - Update your ftp.data file with the CERTAUTH's virtual key ring:

      KEYRING *AUTH*/*
  - If the partner's root CA certificate of the FTP server certificate is not already in your RACF database
    - One more step – add it to the RACF database

      RACDCERT CERTAUTH ADD('<dataset that contains the partner's CA cert>') WITHLABEL('<partner CA>')

FTP Client authentication(1 of 2)

- Scenario
  - My partner's FTP server in Windows needs to authenticate my server on z/OS before it accepts the files I send
- Set up
  - Create a certificate for your FTP client certificate
    - RACDCERT ID(FTPID) GENCERT… WITHLABEL('<mycert>') SIGNWITH(CERTAUTH LABEL('<my CA cert>')

      OR
    - Create a request using GENREQ and send it to an external CA, after receiving it, add it to RACF (See slide 65 – GENREQ)
  - Create a key ring for the FTP client
    - RACDCERT ID(FTPID) ADDRING(ftpring)
  - Connect the client cert to the FTP client ring as the default cert
    - RACDCERT ID(FTPID) CONNECT(LABEL('mycert>') RING(ftpring) DEFAULT)
  - Connect your CA cert (<my CA cert>) to the FTP client ring

FTP Client authentication(2 of 2)

- Add your partner's CA cert to the RACF database
- Connect your partner's CA cert to FTP client ring
- Update your ftp.data file with the client key ring:

  KEYRING FTPID/ftpring

Planning, Planning, Planning(1 of 3)

- To set up a certificate for secure traffic the first time is not that difficult
- The difficult part is the maintenance on its life cycle
- Certificate expiration causes system outage
- Things to consider:
  - How many certificates are actively used in the system?
  - Categorize them by
    - certs locally created VS certs by external provider
    - certs used to authenticate the incoming requests VS certs to identify your servers to the other parties
      - What CA certs will you trust?
      - Each server will have its own ring and own cert or shared?

Planning, Planning, Planning(2 of 3)

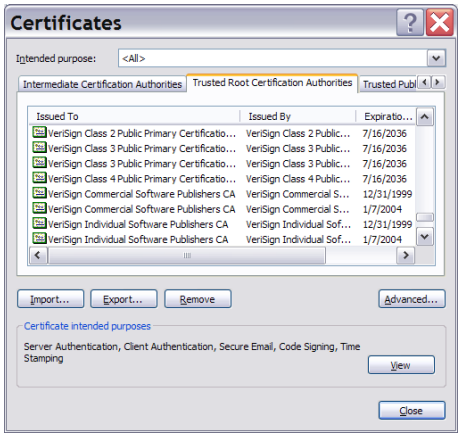
- If you are a local CA which issues certs to the other systems
  - who should be responsible to keep track of the expiry date? 'you' as the issuer or 'they' as the requestors?
  - when to renew your CA cert?
    - A 10 year validity CA cert should not issue 2 year validity cert after the 8th year

Planning, Planning, Planning (3 of 3)

- How to keep track of the expiration dates of all the certificates in the system?
  - Spreadsheets?
  - Utilities?
  - Automation for renew?
  - Use certificate management vendor products?

Build or Buy? (1 of 2)

- Who will be validating your certificate?
  - Global internet customers
    - Easier to buy from a well known CA since it is already installed in the browsers' certificate store



Build or Buy? (2 of 2)

- Internal servers, employees
  - Build your own since you can have the internal CA certs distributed easily
- Business partners
  - Either way
  - If you already built a trust relationship with the partners, there should be no problem for them to install your CA cert

Part 5 - Introduction to PKI Services

z/OS PKI Services Overview(1 of 3)

- Not a priced product. Licensed with z/OS
  - not getting enough marketing focus
  - not sure if IBM will continue the investment in this 'free' component
- A component on z/OS since V1R3, V1R13 was available last year
- Closely tied to RACF
  - The CA cert must be installed in RACF's key ring
  - Authority checking goes through RACF's callable service

z/OS PKI Services Overview(2 of 3)

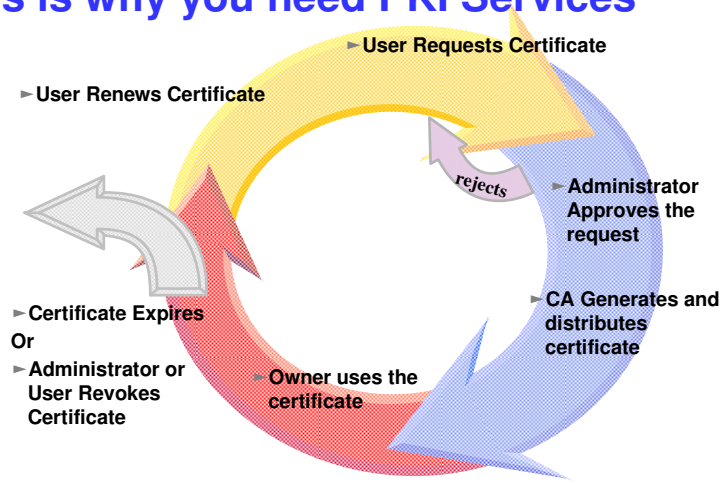
– **Provide more functions than RACDCERT**
  - Full certificate life cycle management: request, create, renew, revoke
  - Generation and administration of certificates via customizable web pages
  - Support automatic or administrator approval process
  - Certificates can be picked up from the requestor's machine
  - Support multiple revocation checking mechanisms
    - Certificate Revocation List (CRL)
    - Online Certificate Status Protocol (OCSP)
  - Certificates and CRLs can be posted to LDAP and/or stored in an HFS file
  - Support Simple Certificate Enrollment Protocol (SCEP) to enable routers to request/renew certificates automatically

z/OS PKI Services Overview (3 of 3)

  - Provides email notification
    - to notify end user for completed certificate request and expiration warnings
    - to notify administrator for pending requests
    - to send the automatic renewed certificate

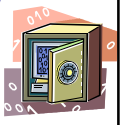
– Provides Trust Policy Plug-in for certificate validation

Certificate Life Cycle – This is why you need PKI Services



Other benefits of using PKI Services

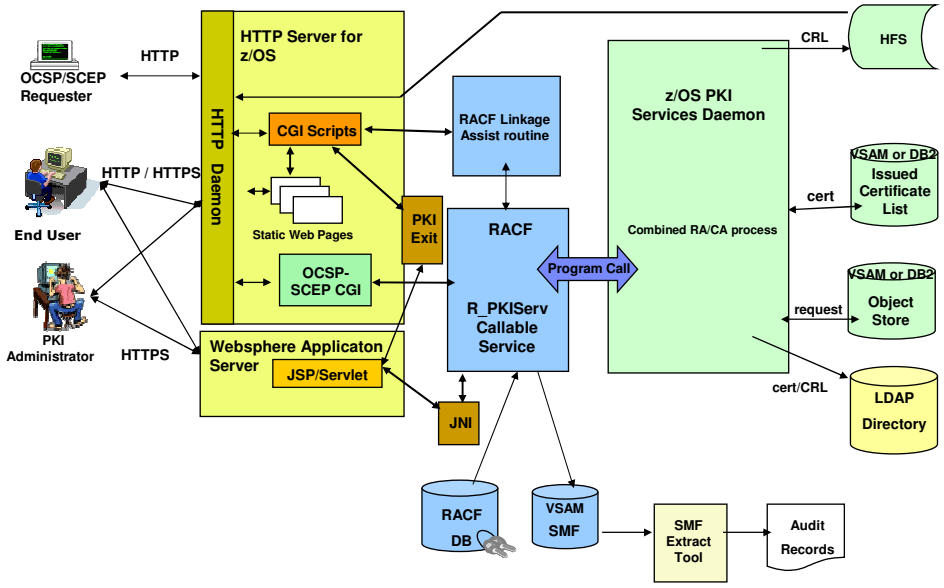
- An alternative to purchasing third party certificates
  - Cost efficient for banks, government agencies to host Digital Certificate management
- Provide options for requestor to generate his own key pair or request the PKI CA to generate it
- IdenTrust™ compliant
  - ensures adherence to a common standard to provide a solid foundation for trust between financial institutions and their customers
- Relatively low MIPS to drive thousands of certificates
- Leverage existing z/OS skills and resources
- Run in separate z/OS partitions (integrity of zSeries® LPARs)
- Scalable (Sysplex exploitation)
- Secure the CA private key with zSeries cryptography



Major Prerequisite Products

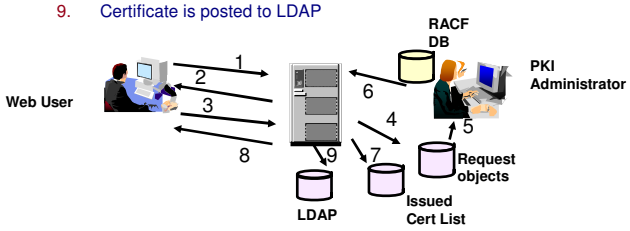
- **RACF (or equivalent)**
  - For storing PKI CA certificate
  - For authorization
- **IBM z/OS HTTP Server / Websphere Application Server**
  - For web page interface
- **LDAP Directory (z/OS or other platforms)**
  - For publishing issued certificates and CRLs
  - For email notification
- **ICSF (optional)**
  - For more secure CA private key
  - For PKI CA to generate key pair
- **z/OS Communications Server (optional)**
  - For email notification
- **DB2 (optional)**
  - An alternative for PKI backend VSAM stores

z/OS PKI Services structure



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

1. User contacts PKI Services to request for certificate
2. CGI/JSP constructs a web page for user to input information
3. CGI/JSP 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/JSP 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



Customization

- **Configuaration file** - pkiserv.conf (used by the PKI Services daemon)
  - Contains mainly setup information for PKI Services
  - May contain certificate information applies to all types of certificates that PKI Services creates
- **Template file** - pkiserv.tmpl (used by the PKI Services CGIs), pkitmpl.xml (used by PKI Services JSPs)
  - Provides different types of certificate template
    - Browser certificate – key generated by browser
    - Server certificate – key generated by server
    - Key certificate – key generated by PKI CA
  - Each template contains certificate information that is specific to a certain type of certificate
    - S/MIME, IPSEC, SSL, CA, Windows Logon…

Continuous enhancements (1 of 3)

V1R11:

- Support long Subject Distinguished Name up to 1024 characters (PTF UA52092)
- Provide option for the user to request PKI CA to generate the key pair
- Provide support for key recovery for those generated by the PKI CA
- Support SHA256 in the signing algorithm
- Implement the web pages with XML and JSPs to facilitate the integration with PKI Services from other applications

Continuous enhancements (2 of 3)

V1R12

- Support Elliptic Cryptographic Curve (ECC) keys, in addition to RSA and DSA keys
- Support Certificate Management Protocol (CMP) clients to communicate with PKI Services
- Provide automatic detection and correction on the potential problem causing by the used serial number
- Provide utilities to post certificates and Certificate Revocation List (CRL) on demand.
- Provide configurable time switches for the housekeeping tasks
- Support the creation of custom extensions to certificate
- Support the creation of Subject Alternate Name that contains multiple instances of each of the General Name forms
- Support the creation of certificates with expiration dates in the far future

Continuous enhancements (3 of 3)

V1R13

- Hardware Elliptic Curve Cryptography (ECC) support - PKI Services can use a hardware ECC certificate as its CA certificate.
- Optional Use of DB2 for ObjectStore and Issued Certificate List backends - Allow PKI customers who needs a large number of certificates to make use of the scalability and flexibility of DB2. Utilities to migrate existing PKI data from VSAM to DB2 are provided.
- Enable the Mozilla base browsers on both the Windows and Linux platform to use the smart card to generate certificates. Support PKI web pages to run on Internet Explorer version 8.
- Support large Certificate Revocation Lists (CRLs) for customers whose applications only support a small number of CRL distribution points.

Using RACF or PKI Services as a CA?

Use RACDCERT if	Use PKI Services if
Just need to generate a handful of certificates	**Need to generate a large number of certificates**
You can manually keep track of the expiration dates of the certs	**You want to get notification on the expiration dates of the certs**
You want to manually send the certs to the other parties	**You want the other parties to retrieve the certs themselves**
You don't care if the certs are revoked	**You want the certs to be checked for revocation status**
You just need basic extensions in the certs	**You want more supported extensions in the certs**

Note: PKI Services does not have any function to manage the key ring. Ring management is provided by RACF.

An user experience - saves millions by using z/OS PKI Services

Data is provided by Vicente Ranieri Junior who works with Banco do Brasil in deploying PKI Services

Banco do Brasil

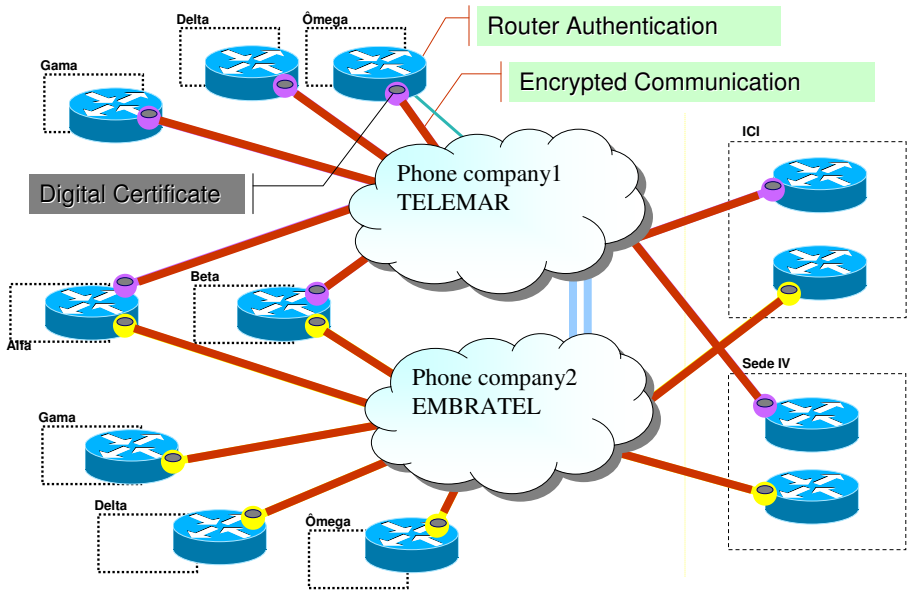
- Owned by the Brazilian government
- The largest bank in Brazil
- Over 200 years old
- It maintains 4,000 banking locations throughout the country and more than a hundred international branches in 23 countries
- It has more than 40,000 ATM machines - the largest number of ATM machines in the financial market
- 87,000 Employees
- More than 30,000,000 customers
- Currently, Banco do Brasil is among the 3 largest IBM zSeries customers worldwide

www.bb.com.br

Banco do Brasil Problem

- In 2003, following a market trend, Banco do Brasil outsourced its network to two telephone companies in Brazil
- Banco do Brasil lost the control over the path security where their critical data are flowing
- In order to enhance the network security, the telephone companies had to establish a VPN tunnel for each router pair in the network providing privacy and authentication

www.bb.com.br



Number of Certificates needed at Banco do Brasil

- **For Equipments and Applications – routers, internet banking**
  - 2007 : 14,000 digital certificates
  - Near Future: 66,000 digital certificates
- **For People – employees, bank lawyers**
  - 2007 : 2,000 digital certificates
  - Near Future: 80,000 digital certificates

The increase in projection number for certificates is due the 'extended services network' in which pharmacies, lottery booths need to be authenticated via certificates to perform small banking services.

Let's look at the YEARLY cost

Cost of certs for Equipment and Applications					
First Year			Projected		
Qty	Price per Cert	Total	Qty.	Price per Cert	Total
14,000	995.00	13,930,000.00	66,000	995.00	65,670,000.00

Cost of certs for People					
First Year			Projected		
Qty	Price per Cert	Total	Qty.	Price per Cert	Total
2,000	* 13.00	26,000.00	80,000	* 13.00	1,040,000.00

*** Special Price from Brazilian Government Agency CA**

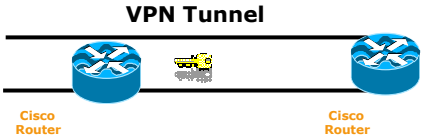
Solutions considered

- **OpenCA**
  - Pros : Free
  - Cons: No support
- **Windows Server Certificate Services**
  - Pros : Support available
  - Cons: Scalabity issue, business continuity concern
- **z/OS PKI Services**
  - Pros : Free, scalable, support available
  - Cons: Some required certificate fields and protocol not supported yet

Banco do Brasil Solution(1 of 4)

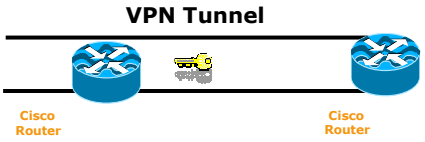
- **Banco do Brasil submitted requirements to IBM to enhance PKI Services**
- **After knowing that the requirements were in plan, Banco do Brasil decided to start exploiting z/OS PKI Services to issue its VPN digital certificates**

Banco do Brasil Solution(2 of 4)



- In Brazil, there are 2 ways to be a certified CA
  - get a certification from the PKI Brazil government department which requires the PKI application runs alone on a separate machine (the bank is working on getting the acceptance that LPAR isolation is as good as a stand alone machine)
  - the issuer and the requester sign an agreement
- Banco do Brasil signed an agreement with the telephone companies

Banco do Brasil Solution(3 of 4)



- Banco do Brasil network had its security dramatically improved with almost no additional cost (z/OS is their prime operating system and RACF was already deployed)
- In a week's time, PKI Services was set up and running in the test system
- Low consumption of MIPS to run PKI Services
- There are no extra head counts to run PKI Services
- The customer cost was only related to customize z/OS PKI Services pages to meet their requirements

PKI Services Certificate Generation Application

Install our CA certificate into your browser

Shipped sample

Choose one of the following:

- **Request a new certificate using a model**

  Select the certificate template to use as a model: 1-Year PKI SSL Browser Certificate

  Request Certificate

- **Pick up a previously requested certificate**

  Enter the assigned transaction ID

  Select the certificate return type: PKI Browser Certificate

  Pick up Certificate

- **Renew or revoke a previously issued browser certificate**

  Renew or Revoke Certificate

- **Administrators click here**

  Go to Administration Page

email: webmaster@your-company.com

ICP-BB INTERMEDIARIA 02 - Microsoft Internet Explorer fornecido por Banco do Brasil

Arquivo Editar Exibir Favoritos Ferramentas Ajuda

Endereço https://acint02lab.bb.com.br/PKIServ/public-cgi/camain.rexx?

Autoridade Certificadora Intermediária 02 LAB

PKI Services - GERAÇÃO DE CERTIFICADOS

After customization

Baixar o certificado de nossa AC RAIZ
Baixar o certificado de nossa AC INTERMEDIARIA 01
Baixar o certificado de nossa AC INTERMEDIARIA 02
Baixar "Termo de Compromisso para Acesso Remoto"

Escolha uma opção:

- **Solicitar um novo certificado utilizando um modelo**

  Selecione o tipo de certificado desejado: 1-ANO CONTINGENCIA SSL BROWSER

  Solicitar Certificado

- **Receber certificado solicitado**

  Informe o ID da transação

  Selecione o tipo de certificado: CERTIFICADO PKCS10

  Receber Certificado

Intranet local

Iniciar

Shipped sample

Retrieve Your 1-Year PKI SSL Browser Certificate

Please bookmark this page

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.

Enter the assigned transaction ID

1jTQjs0h/cpk2SHV++++++++

If you specified a pass phrase when submitting the certificate request, type it here, exactly as you typed it on the request form

****

Retrieve and Install Certificate

To check that your certificate installed properly, follow the procedure below:

Netscape V6 - 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.

Netscape V4 - Click the Security button, then Certificates-> Yours. Your certificate should appear in the list. Select it then click Verify.

Internet Explorer V5 - Click Tools->Internet Options, then Content, Certificates. Your certificate should appear in the Personal list. Click Advanced to see additional information.

Home page

email: webmaster@your-company.com

ICP-BB INTERMEDIARIA 02 - Microsoft Internet Explorer fornecido por Banco do Brasil

Arquivo Editar Exibir Favoritos Ferramentas Ajuda

Endereço https://acint02lab.bb.com.br/PKIServ/ssl-cgi-bin/caretrieve.rexx?TransactionId=1j4qgIsG3rVaY6faUE%2B%2B%2B%2B%2B%2B%2B&Template=1-ANO+CONTINGENCIA+S

Autoridade Certificadora Intermediária 02 LAB

After customization

Receber Certificado do tipo:
1-ANO CONTINGENCIA SSL BROWSER

Sugestão: Caso seu Certificado ainda não esteja disponível, acrescente esta página aos seus "Favoritos". Com isto, na próxima verificação, o browser irá mostrar automaticamente o ID da transação, facilitando verificar o status e a recepção de seu certificado.

Informe o ID da transação

1j4qgIsG3rVaY6faUE++++++

Digite a senha informada quando da solicitação do Certificado

Clique aqui para receber e instalar o certificado

Atenção usuários dos navegadores Firefox, Mozilla e Netscape: Após clicar no botão acima, estes navegadores não retornam mensagem informando que o certificado foi instalado com sucesso.

Para verificar se seu certificado está corretamente instalado siga os procedimentos abaixo:

Firefox 1.5- Clique Ferramentas -> Opções -> Avançado -> Segu rança -> Certificados -> Seus certificados -> Dê um duplo clique em seu certificado para para obter informações mais detalhadas sobre o mesmo.
Internet Explorer V5- Clique em Ferramentas -> Opções da Internet -> Conteúdo -> Certificados

Concluído Intranet local

Iniciar 2 Intern... F9440982... Apresent... 3 Micros... Sessão C ... 15:06

Banco do Brasil Solution(4 of 4)

- **Both telephone companies that outsourced Banco do Brasil network request and receive the VPN digital certificates through PKI Services web interface**
- **The phone companies send the serial numbers of the routers that need certificates to a manager**
- **They then use the RACF IDs in the Bank's system to request certificates for the routers**
- **The administrator checks if there's an email from the manager on the routers before the requests are approved**
- **The certificates are issued with 1 to 2 years' validity period**

Performance

- Measured in a z900 model 2064-104 with hardware encryption and VSAM buffering
- 19.2 certificates created per second
- With 1+ million certificates created, queries with a requestor value specified as criteria returned in less than 1 second.
- With 1+ million certificates created and 5% revoked, CRL refreshing in LDAP (using 3055 CRL distribution points) took an average of 3 minutes.

Summary

- z/OS PKI Services is a complete Certification Authority package running under z/OS.
- It provides full certificate life cycle management
- No cost per issued digital certificate
- It is a very Secure, Scalable and Available PKI solution

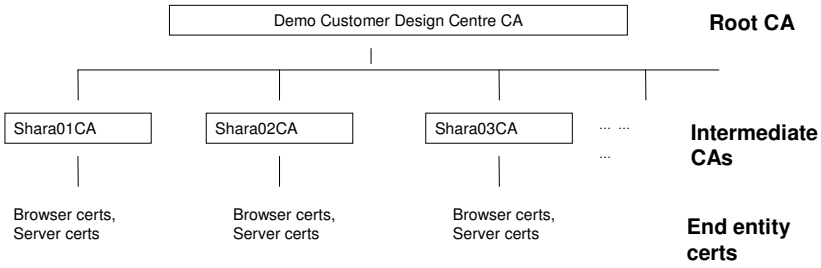
Part 6 - Hands on Lab on PKI Services

A separate hardcopy will be provided

Objectives of this Lab

At the end of this lab, you will be able to

- **Submit and approve a certificate request for**
  - **A certificate with key pair generated by the browser – EX 1**
  - **A certificate with key pair generated by PKI Services – EX 2**
  - **A certificate with key pair generated on a z/OS server – EX 3**
- **View the installed certificate from the IE broswer – EX 4**
- **Revoke/Suspend a certificate – EX 5**
- **Check the certificate status – EX 6**
  - **Certificate Revocation List (CRL)**
  - **Online Certificate Status Protocol (OCSP)**
- **Customize PKI Services – EX 7**
  - **Configuration file – pkiserv.conf**
  - **Template file – pkiserv.tmpl**



Exercise Instructions:
Note 1: All the references of xx refer to the number part of your assigned id, eg. 01 if your assigned ID is sharb01)

Note 2: You will play both roles as an end user and as an administrator in the lab. The tasks performed by an end user and an administrator are indicated by a male and a female icon respectively - 

Note 3: If you are not familiar with the MVS/OMVS system, you may refer to Appendix 1 to get some hints.

PCOMM icon: 'mvs'

Host Password: firstpw

Window Password:

You are on your way to earn your 66 millions!!!

References

- **IBM Education Assistant web site:**
  http://publib.boulder.ibm.com/infocenter/ieduasst/stgv1r0/index.jsp
- **RACF web site:**
  http://www.ibm.com/servers/eserver/zseries/zos/racf
- **PKI Services web site:**
  http://www.ibm.com/servers/eserver/zseries/zos/pki
- **IBM Redbooks**
  z/OS V1 R8 RACF Implementation
- **Security Server Manuals:**
  RACF Command Language Reference
  RACF Security Administrator's Guide
- **Cryptographic Server Manual**
  Cryptographic Services System Secure Sockets Layer Programming
- **RFCs**
  RFC2459 - Internet X.509 Public Key Infrastructure Certificate and CRL Profile
  RFC5280 - Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile