#SHAREorg

SHARE
Technology · Connections · Results

Digital Certificate Demystified

Ross Cooper, CISSP®
IBM Corporation

August 2013
SHARE Session: 13651

SHARE in Boston

Trademarks

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

AIX*	Domino*	Language Environment*	SYSREXX	z10
BladeCenter*	DS6000	MVS	System Storage	z10 BC
BookManager*	DS8000*	Parallel Sysplex*	System x*	z10 EC
CICS*	FICON*	ProductPac*	System z	zEnterprise*
DataPower*	IBM*	RACF*	System z9	zSeries*
DB2*	IBM eServer	Redbooks*	System z10	
DFSMS	IBM logo*	REXX	System z10 Business Class	
DFSMSdss	IMS	RMF	Tivoli*	
DFSMShsm	InfinBand	ServerPac*	WebSphere*	
DFSMSrmm				
DFSORT				

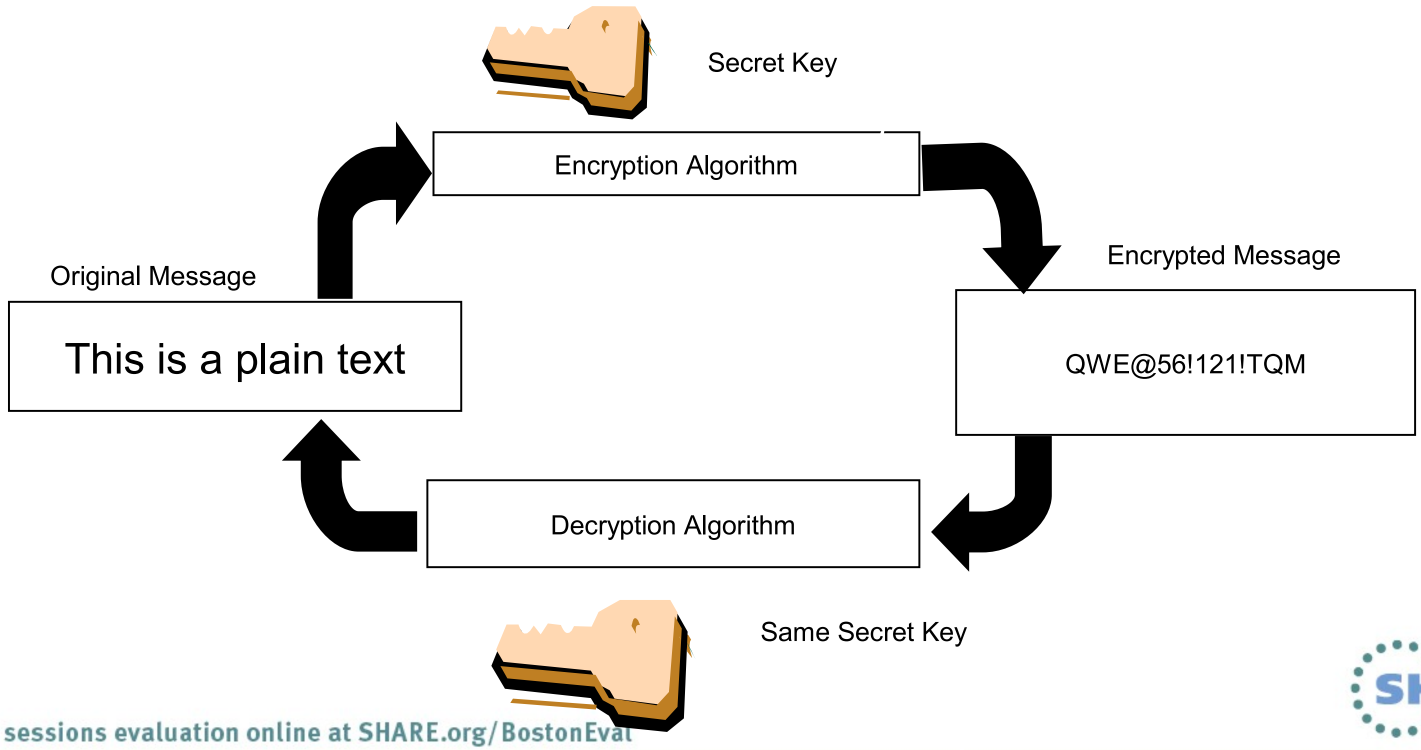
* Registered trademarks of IBM Corporation

Agenda

- **Cryptography**
- What are **Digital Certificates**
- Certificate **Types** and **Contents**
- Certificate **Formats**
- Certificate **Validation**
- Certificates and **SSL**
- Certificate **Life Cycle**

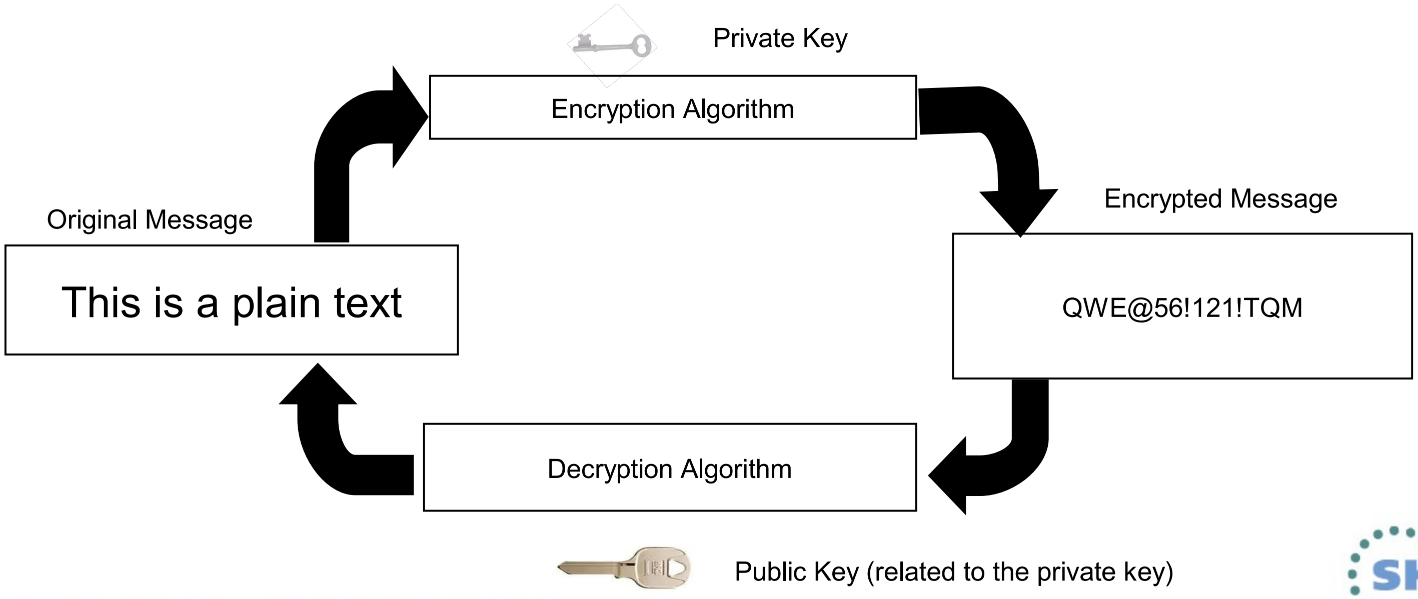
Symmetric Encryption

- **Provide data confidentiality**
- **Same key** used for both encryption and decryption
- **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



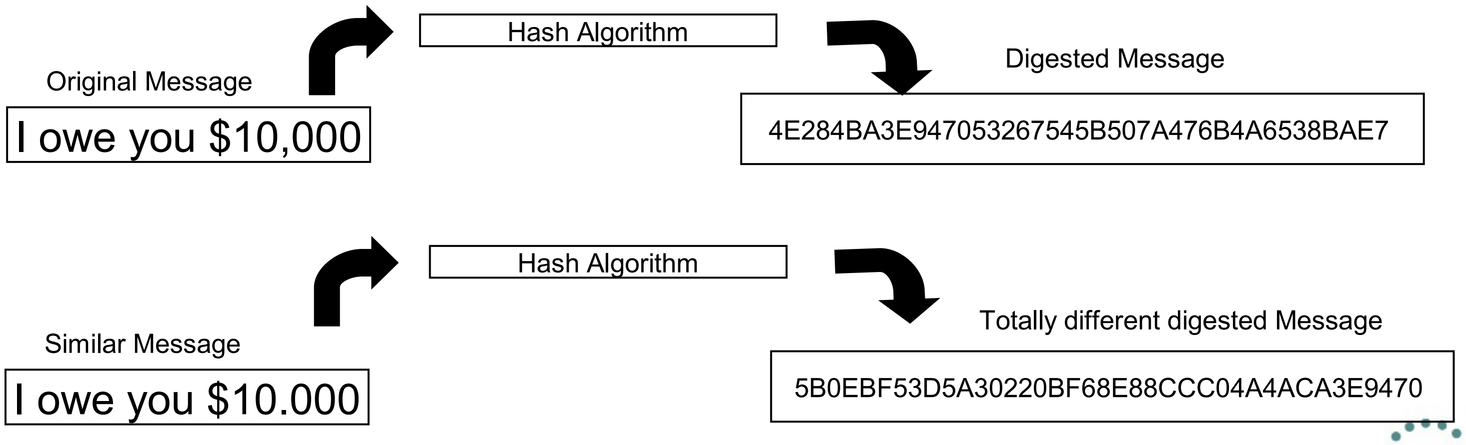
Asymmetric Encryption

- **Public / private key pairs** - 2 different keys
- 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.
- **Slow,** Very expensive computationally
- **Public key is freely distributed** to others, private key is securely kept by the owner
- **Common algorithms**: RSA, DSA, ECC



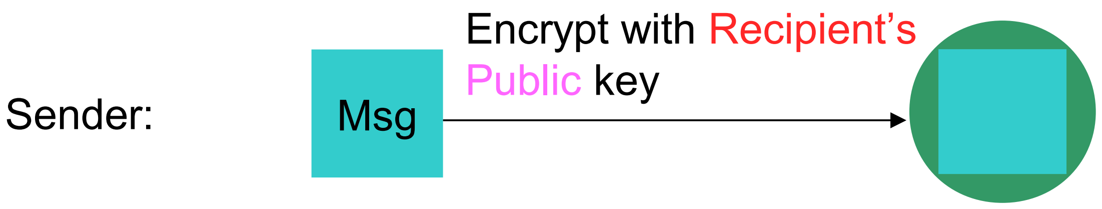
Message Digest (Hash or Fingerprint)

- 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
- **No keys involved** – Result determined only by the algorithm
- Play a part in data integrity and origin authentication
- **Common algorithms**: SHA1, SHA256

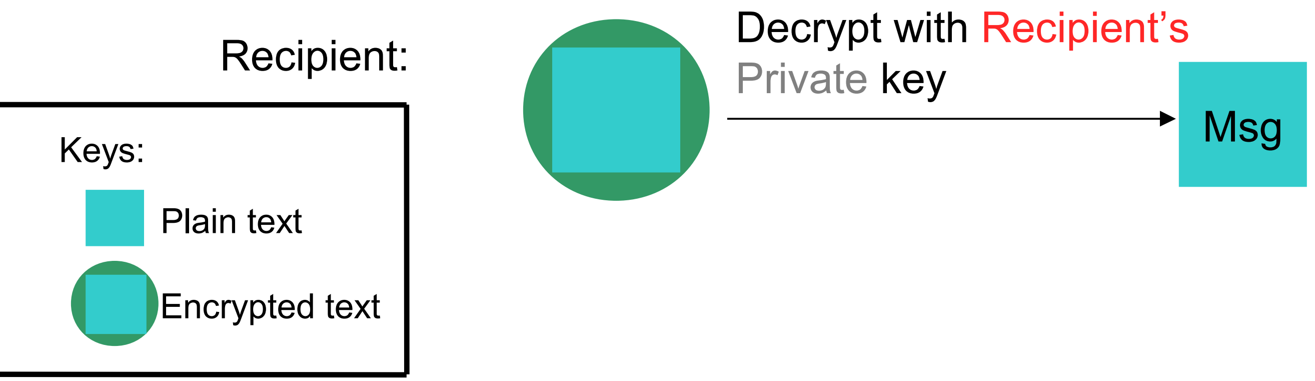

Original Message
I owe you $10,000
Hash Algorithm
Digested Message
4E284BA3E947053267545B507A476B4A6538BAE7
Similar Message
I owe you $10.000
Hash Algorithm
Totally different digested Message
5B0EBF53D5A30220BF68E88CCC04A4ACA3E9470

Asymmetric Encryption (for confidentiality)

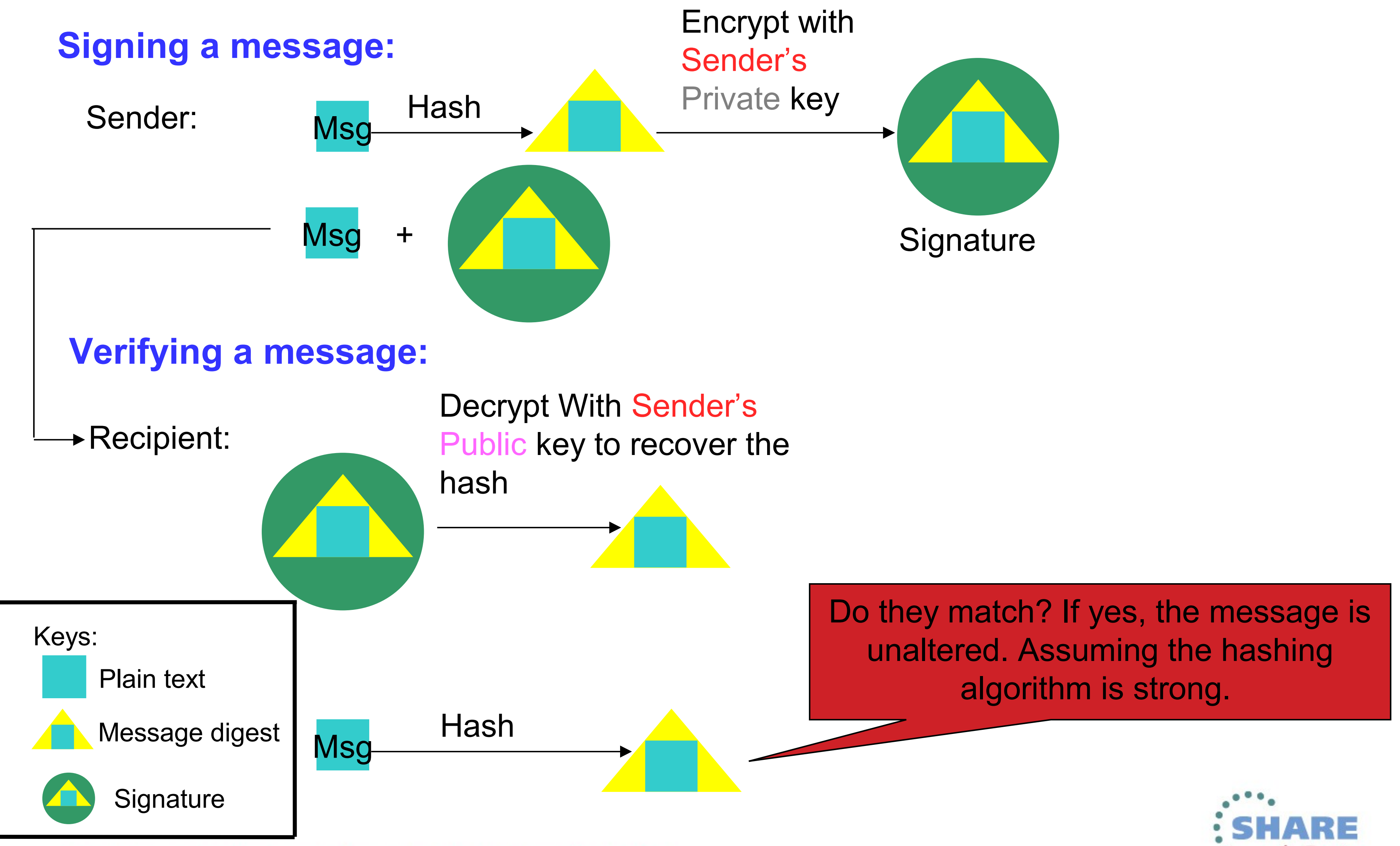
Encrypting a message:



Decrypting a message:



Signing (for integrity and non repudiation)


Signing a message:
Sender:
Msg
Hash
Encrypt with Sender's Private key
Signature
Msg
+
Verifying a message:
Recipient:
Decrypt With Sender's Public key to recover the hash
Do they match? If yes, the message is unaltered. Assuming the hashing algorithm is strong.
Keys:
Plain text
Message digest
Signature
Msg
Hash

SHARE in Boston

What is a Digital Certificate?

A Digital Certificate is a digital document issued by a trusted third party which binds an end entity to a public key.

- **Digital document:**
  - Contents are organized according to ASN1 rules for X.509 certificates
  - Encoded in binary or base64 format
- **Trusted third party** aka **Certificate Authority** (CA):
  - The consumer of the digital certificate trusts that the CA has validated that the end entity is who they say they are before issuing and signing the certificate.
- **Binds the end entity to a public key:**
  - **End entity** - Any person or device that needs an electronic identity. Encoded in the certificate as the Subjects Distinguished Name (SDN). Can prove possession of the corresponding private key.
  - **Public key** - The shared half of the public / private key pair for asymmetric cryptography
  - **Digitally signed** by the CA

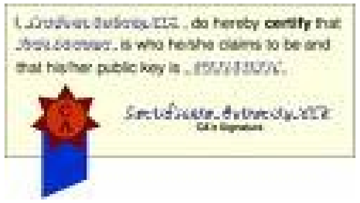
What is a Digital Certificate?

- Best way to think of it is as an **ID card**, like driver licenses or passport
- To **establish your identity** or credential to be used in electronic transactions
- Digital certificate technology has been in existence for over 20 years
- 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**
- 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.
- Generally a digital certificate provides:
  - Identity to a person or a server
  - Distribution of a public key

How is Digital Certificate used?

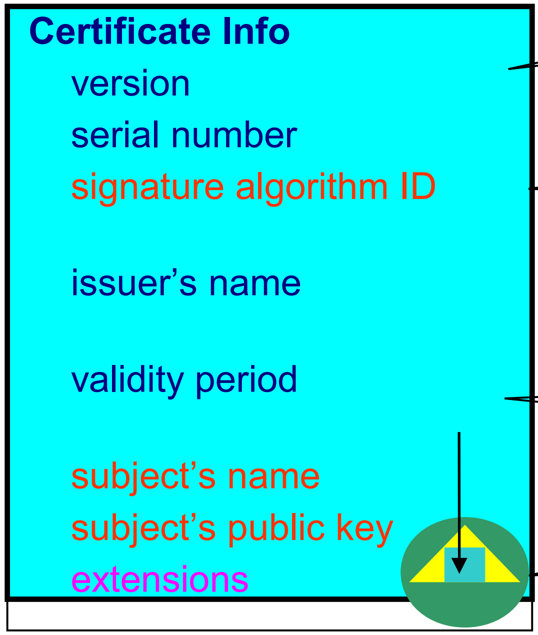
- **Prove Identity to a peer:**
  - Owner of the certificate can prove possession of the certificate's private key
  - Identity can be validated by checking it is signed by a trusted Certificate Authority
- **Prove authenticity of a digital document:**
  - Programs can be signed by code signing certificates
  - E-mail signatures
  - Certificates are signed by CA certificates
- **Establish a secure connection:**
  - Certificates contain a public key which allows protocols such as SSL and AT-TLS to exchange session keys

What is in a Digital Certificate?



I, [illegible], do hereby certify that [illegible], is who he/she claims to be and that his/her public key is [illegible].

[illegible]
CA's Signature



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

Version 1, 2, 3

This is the hash/encrypt algorithm used in the signature, eg. sha256RSA

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

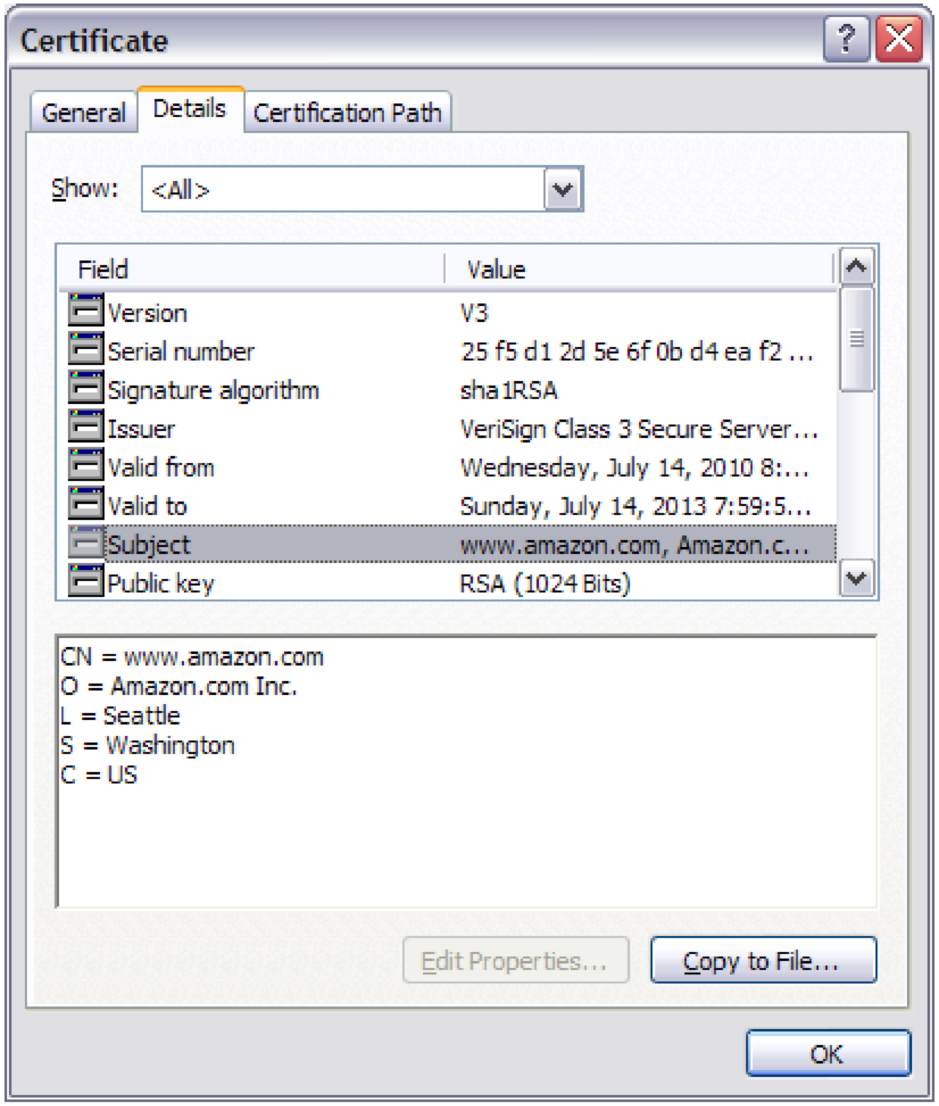
You can NOT change ANY of the certificate information!

SHARE in Boston

Extensions of a X.509 Digital Certificate

- **Adds additional definitions to a certificate and its identity information**
- 15+ extensions currently defined
- Top 7 extensions of interest:
  - **Authority Key Identifier** – Unique identifier of the signer
  - **Subject Key Identifier** – Unique identifier of the subject
  - **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
  - **Key Usage** – Defines how the public key can used
    - Digital Signature
    - Key Encipherment
    - Key Agreement
    - Data Encipherment
    - Certificate Signing
    - CRL signing
  - **Extended Key Usage** – Further defines how the public key can be used
    - *Server Auth*
    - *Code Signing*
    - *Client Auth*
    - *E-mail*
    - *OCSP Signing*
    - *Timestamping*

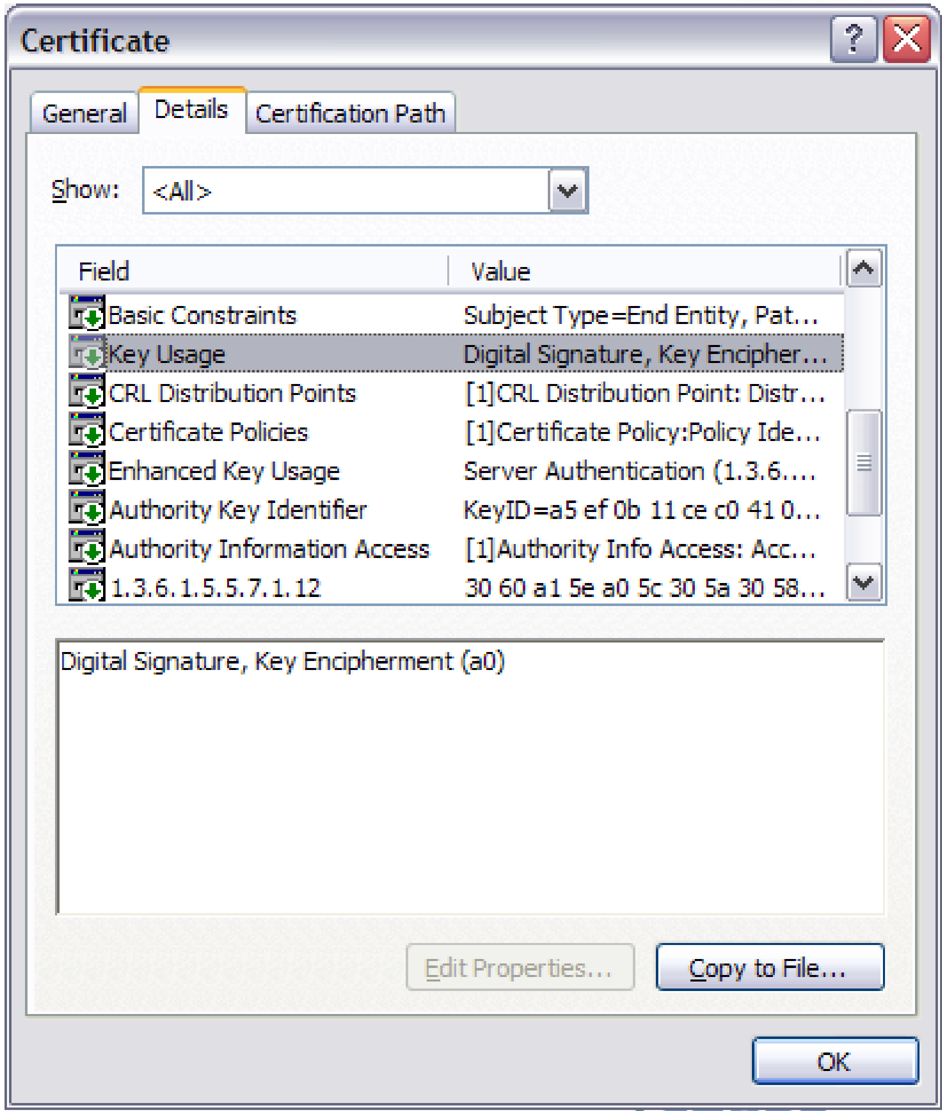
Example of a x.509 Digital Certificate



Show: <All>

Field	Value
Version	V3
Serial number	25 f5 d1 2d 5e 6f 0b d4 ea f2 ...
Signature algorithm	sha1RSA
Issuer	VeriSign Class 3 Secure Server...
Valid from	Wednesday, July 14, 2010 8:...
Valid to	Sunday, July 14, 2013 7:59:5...
Subject	www.amazon.com, Amazon.c...
Public key	RSA (1024 Bits)

```
CN = www.amazon.com
O = Amazon.com Inc.
L = Seattle
S = Washington
C = US
```



Show: <All>

Field	Value
Basic Constraints	Subject Type=End Entity, Pat...
Key Usage	Digital Signature, Key Encipher...
CRL Distribution Points	[1]CRL Distribution Point: Distr...
Certificate Policies	[1]Certificate Policy:Policy Ide...
Enhanced Key Usage	Server Authentication (1.3.6....
Authority Key Identifier	KeyID=a5 ef 0b 11 ce c0 41 0...
Authority Information Access	[1]Authority Info Access: Acc...
1.3.6.1.5.5.7.1.12	30 60 a1 5e a0 5c 30 5a 30 58...

Digital Signature, Key Encipherment (a0)

SHARE in Boston

Digital Certificates and Certificate stores

- Certificate must be placed in a **certificate store** before it can be used by an application, like communication Server or HTTP server for secure communication

- On z/OS, many components call System SSL APIs, which in turn call RACF **R_datalib** callable service to access the certificate store
  - Application → System SSL → R_DataLib

- Different names:
  - Certificate store = key ring = key file = key database

Types of Digital Certificates - Issuer

- **Self Signed**
  - Self-issued
  - Issuer and subject names identical
  - Signed by itself using associated private key
  - No trusted party involved – Must be implicitly trusted
  - Common uses for self signed certificates:
    - Root CA certificate
    - Single trusted certificate

- **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.
  - Common uses for signed certificates:
    - End entity certificates
    - Intermediate CA Certificates

Types of Digital Certificates - 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, EV Certificate…
- **Certificate Authority (CA) certificate**
  - Used to sign other certificates
  - Root CA: the top
  - Intermediate CA: signed by root CA or other intermediate CA

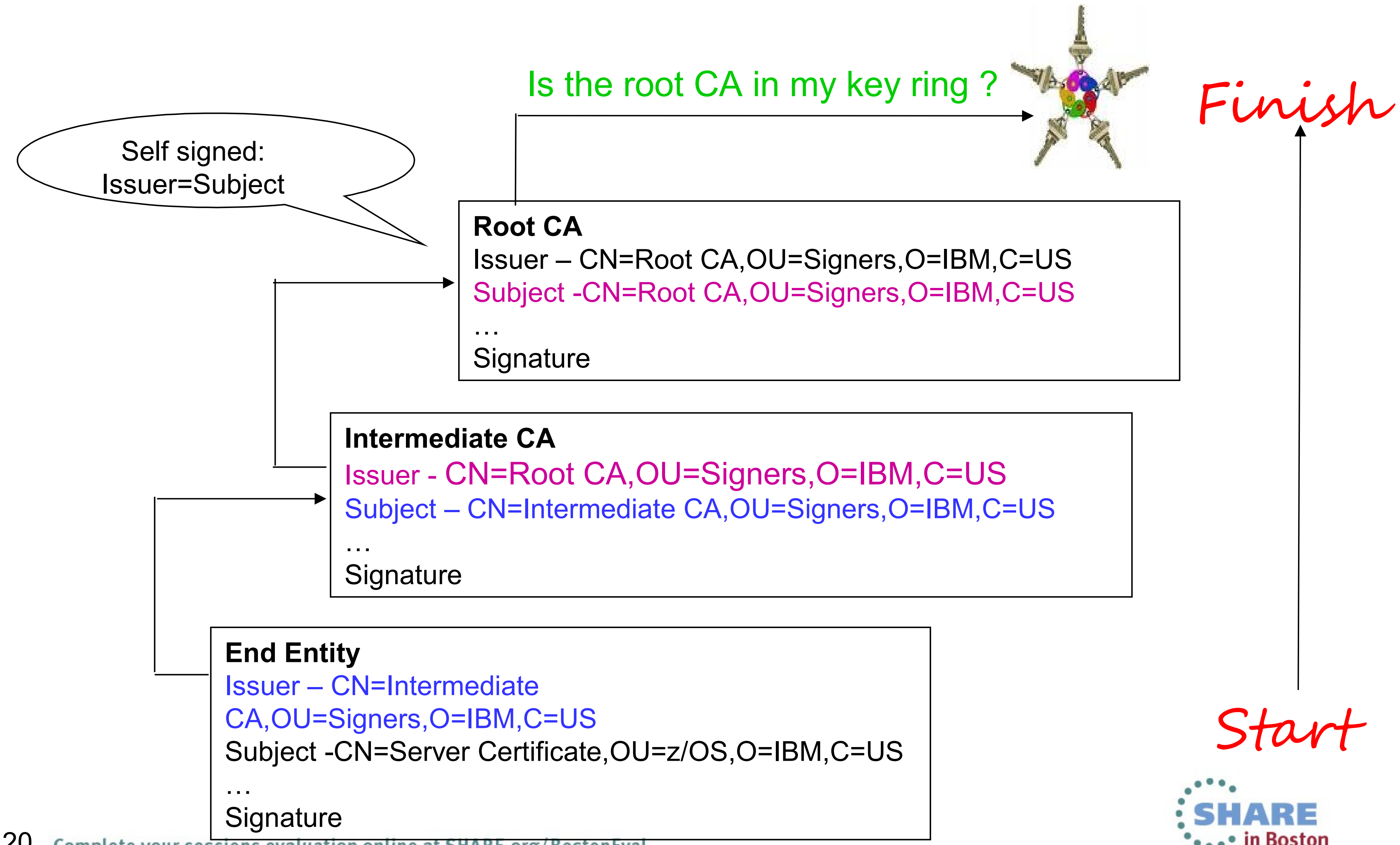
Digital Certificate Formats

- X.509 Digital Certificate can exist in many **different forms**
  - Single certificate
  - **PKCS Package** - (Public-Key Cryptographic Standards) – Developed by RSA
    - **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 is used to convert binary data to displayable text for easy cut and paste

Certificate Revocation

- Normally the lifetime of certificate is the defined **validity period**
- Revocation provides a means for a certificate to become **invalid prior to its validity end date**
- **Reasons for revocation:**
  - Private key associated with the certificate has been **compromised**
  - Certificates are being used for purpose other than what they are defined
- **CRL** – Certificate Revocation List:
  - List of certificates that should no longer be trusted
  - CRL Distribution Point extension in the X.509 certificate gives information about where to locate revocation information for the certificate.
- **OCSP** – Online Certificate Status Protocol:
  - Provides a query function for the revocation status of a certificate

Certificate Chain Validation



Root CA
Issuer – CN=Root CA,OU=Signers,O=IBM,C=US
Subject -CN=Root CA,OU=Signers,O=IBM,C=US
…
Signature

Intermediate CA
Issuer - CN=Root CA,OU=Signers,O=IBM,C=US
Subject – CN=Intermediate CA,OU=Signers,O=IBM,C=US
…
Signature

End Entity
Issuer – CN=Intermediate CA,OU=Signers,O=IBM,C=US
Subject -CN=Server Certificate,OU=z/OS,O=IBM,C=US
…
Signature

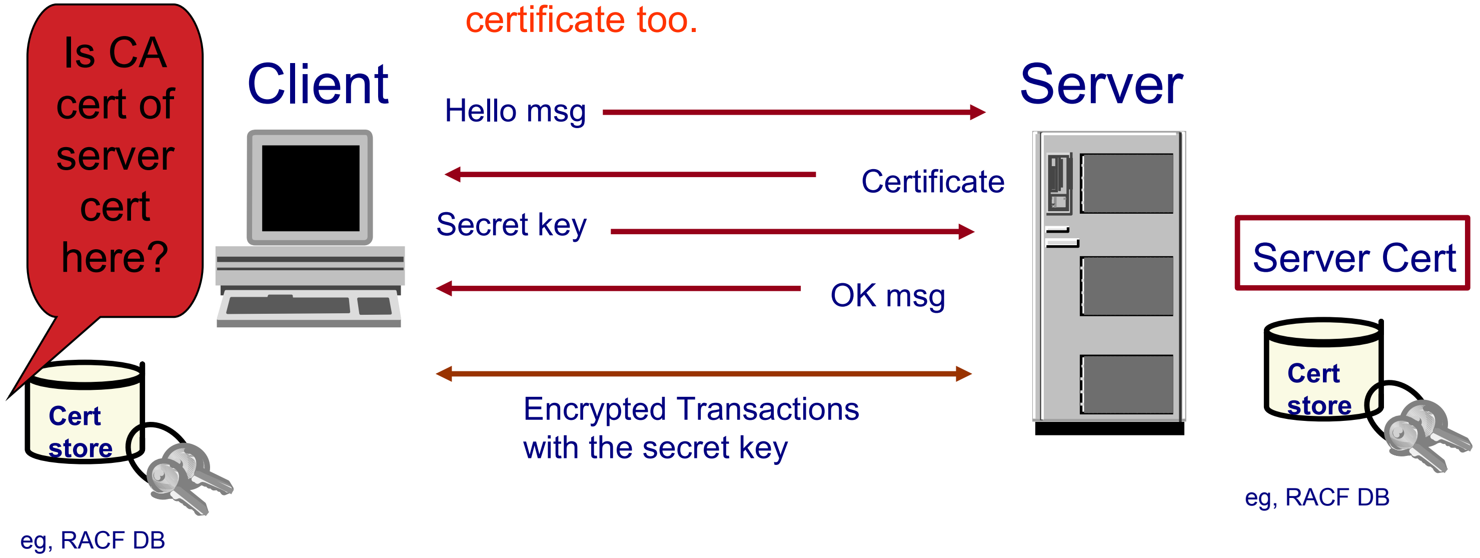
Certificate Validation

- **Signature chain validation:**
  - End Entity certificate signature is validated by signer's public key
  - Any intermediate CA certificates signatures are validated against their signer's public key
  - Root CA certificate is validated against it's own public key
  - Root CA certificate must be trusted
- **Validity period** – Check if the certificate has expired
- **Status** – Check if the certificate has been revoked:
  - **CRL** - Check if it is on a Certificate Revocation List
  - **OCSP** - Check with the CA which issued this certificate through the Online Certificate Status Protocol

Certificates 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.



Setup a certificate for SSL handshake

1. **Create a key ring (aka key file / certificate store)**
2. Install the **CA certificates** that will be used for SSL handshake
3. Generate a **certificate signing request** (also CSR)
   - Like an **application** to a certificate authority to obtain a signed digital certificate
   - Contains info about on the requestor
     - Identifying information, like **subject name**
     - **Public key** (may be generated before the request or generated at the same time as the request)
     - Other credentials or **proofs of identity** required by the certificate authority
     - 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

Setup a certificate for SSL handshake

4. 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.
5. Install the certificate to the **key ring**
6. **Permit the application** to access the key ring, the certificate and its associated private key
   - If it is a **RACF key ring**, use access control through <ring owner>.<ring name>.LST in the **RDATALIB** class
   - If it is a **key file**, permission is through the file **system's permission bits and password**

Certificate Life Cycle

- **To set up a certificate for secure traffic the first time is only the beginning**
- Must plan for the **certificate life cycle**
- Certificate expiration causes **application outage**
- Things to consider:
  - **How many** certificates are actively used in the system?
  - Certs **locally created** VS Certs by **external provider**
- 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?

Review

- **Cryptography**
- What are **Digital Certificates**
- Certificate **Types** and **Contents**
- Certificate **Formats**
- Certificate **Validation**
- Certificates and **SSL**
- Certificate **Life Cycle**

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

Questions?

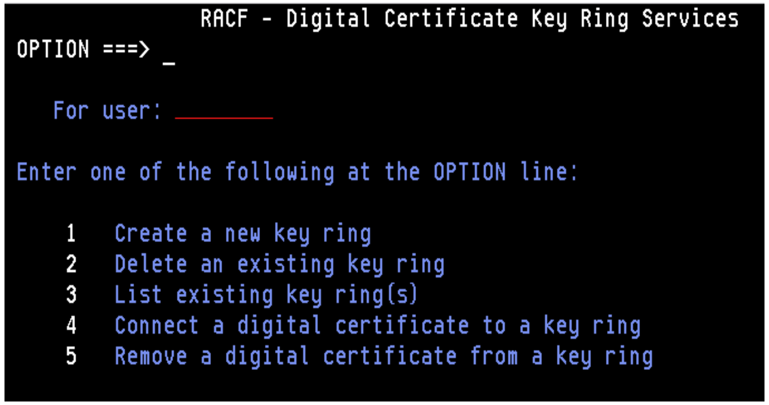
Ross Cooper
Session: 13651

Backup Slides:

- **RACDCERT** Overview & Commands
- RACF **Key Rings** and **Certificates**
- Certificate **Types** and **Contents**
- **PKI Services:** Full Certificate Authority

RACDCERT Overview

- RACDCERT is the primary administrative tool for managing digital certificates using RACF.
  - TSO command shipped as part of RACF
  - Command line interface with ISPF panels
  - Certificates and Rings are protected by RACF profiles
- Learn more:
  - RACF Command Language Reference

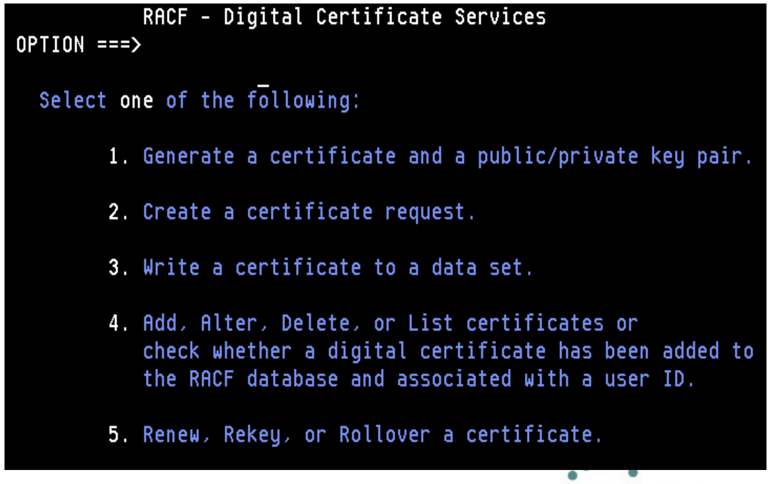


```
                RACF - Digital Certificate Key Ring Services
OPTION ===> _

   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
```



```
           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.
```

```
RACDCERT ID(FTPServer) GENCERT SUBJECTSDN(CN('Server
Certificate')OU('Production')O('IBM')L('Poughkeepsie') SP('New
York')C('US')) SIZE(1024) WITHLABEL('Server Certificate')
ALTNAME(DOMAIN('mycompany.com'))
RACDCERT ID(FTPServer) ADD('user1.svrcert') WITHLABEL('Server
Certificate')
RACDCERT ID(userid) EXPORT (LABEL('label-name')) DSN(output-dataset-
name) FORMAT(CERTDER | CERTB64 | PKCS7DER | PKCS7B64 |
PKCS12DER | PKCS12B64 ) PASSWORD('pkcs12-password')
```

RACDCERT Commands

- **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 **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 or a certificate and private key
  - RACDCERT **REKEY** – Renew certificate with new key pair
  - RACDCERT **ROLLOVER** – Finalize the REKEY process

RACDCERT Commands

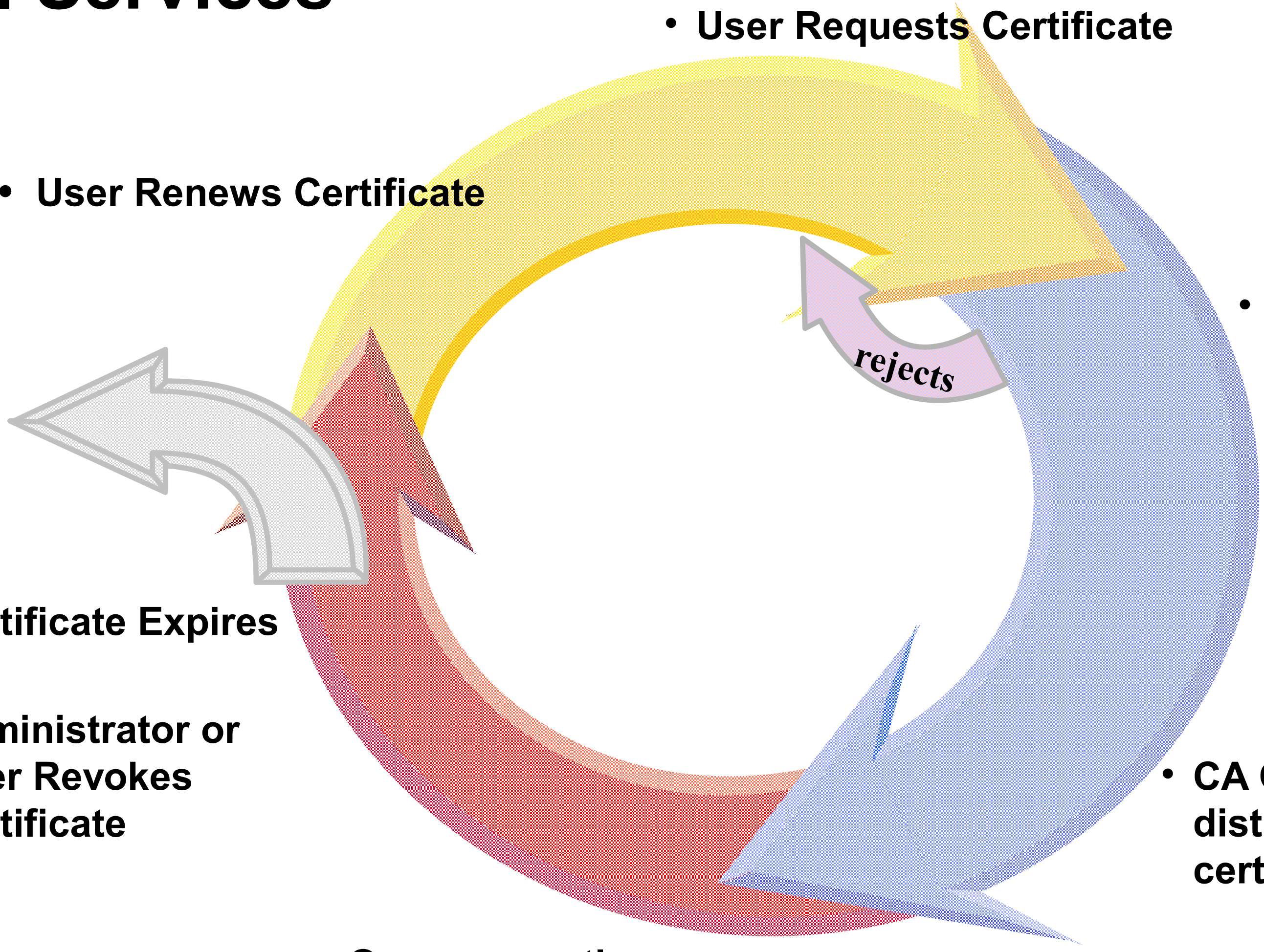
- **Certificate Ring Administration:**
  - **RACDCERT ADDRING – Create a key ring**
  - **RACDCERT CONNECT – Place a certificate in a key ring**
  - **RACDCERT REMOVE – Remove a certificate from a key ring**
  - **RACDCERT LISTRING – Display key ring information**
  - **RACDCERT DELRING – Delete a key ring**

- **Certificate Map 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

RACF Key Rings and certificates

- A key ring is a collection of certificates that **identify a networking trust relationship.**
- A certificate must be placed in a key ring before it can be used by middleware
- applications
- Key Ring Syntax for applications: **<user-id>/<ring-name>**
- **Types of Certificates in RACF:**
  - **User** – Directly Associated with one z/OS user ID (end entity)
  - **CERTAUTH** – Trusted CA certificate used to verify the peer entity's certificate.
  - **SITE** – Certificates associated with an off-platform server or other network identity. SITE certificates bypass the normal certificate chain validation. Private keys can be shared.
- **Key Rings contain Certificate Usage** – The usage assigned to a certificate when it is connected to a key ring indicates its intended purpose.
  - **PERSONAL** – Used to identify a local user or server application. Personal usage must be used to get access to the private key.
  - **CERTAUTH** – Used to verify the peer entity's certificate. Used to identify the local server's CA certificate.
  - **SITE** – Certificate associated with an off-platform server or other network identity. SITE certificates bypass the normal certificate chain validation.

Certificate Authority on z/OS: PKI Services



- User Requests Certificate
- Administrator Approves the request
- CA Generates and distributes certificate
- Owner uses the certificate
- Certificate Expires

Or

- Administrator or User Revokes Certificate
- User Renews Certificate

Certificate Authority on z/OS: PKI Services

- **PKI Services** provides full certificate life cycle management
- **Request**, **create**, **renew**, **revoke** certificates
  - Provides certificate status:
  - **Certificate Revocation List (CRL)**
  - **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 notifications** or renewal of expiring certificates