

Securing and Encrypting Network Traffic with z/OS Communications Server and Policy Agent

Security Workshop

x.509 Certificates for z/OS Communications Server



IBM Washington System Center
IBM Technical Sales Support

Trademarks

- **The following are Registered Trademarks of the International Business Machines Corporation in the United States and/or other countries.**
 - IBM
 - z/OS
 - **The following are trademarks or registered trademarks of other companies.**
 - Microsoft is a registered trademark of Microsoft Corporation in the United States and other countries.
 - All other products may be trademarks or registered trademarks of their respective companies.
 - Refer to www.ibm.com/legal for further legal information.
-
- OSA-Express Features
 - There are many different types of OSA-Express features. In this document where OSA is mentioned it refers to an OSA port on any of the OSA-Express features unless a specific OSA-Express feature is mentioned.

Agenda

- Encryption
- Encryption Keys and Certificates
- Digital Key Storage
- Certificate Usage and RACF Commands
- Certificate Creation
- Host Name Check

Encryption



Confidentiality and Privacy

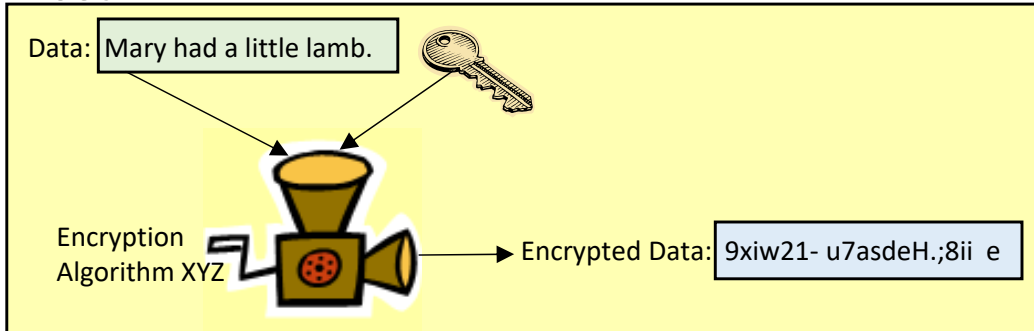
- We encrypt data to make it:
 - Confidential
 - Private
- Confidentiality and Privacy are required by Security Mandates for certain types of data:
 - Data Payload itself (i.e., Credit Card Data with Payment Card Industry - PCI - Mandates)
 - Passwords and optionally Userids (PCI, NIST, etc.)
 - Encryption Keys for maintaining Data Privacy
- Cryptographic Hardware and Software can be validated according to standards listed in FIPS140-2. See the following URLs:
 - <http://csrc.nist.gov/groups/STM/cmvp/documents/140-1/1401val2014.htm>
 - <http://csrc.nist.gov/publications/fips/fips140-2/fips1402annexa.pdf>

Encryption Keys and Certificates

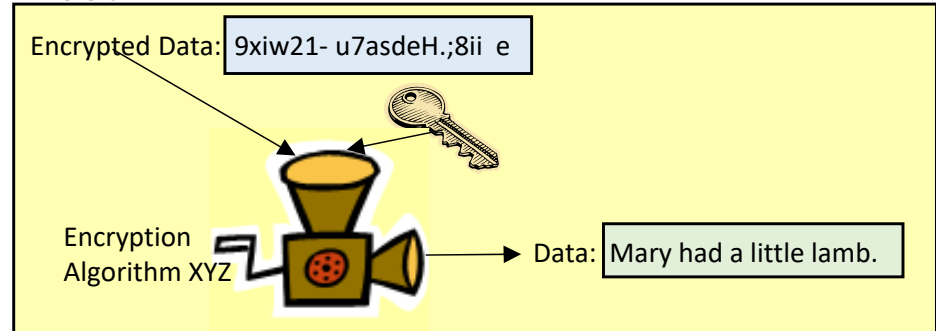


Symmetric Keys

Host A

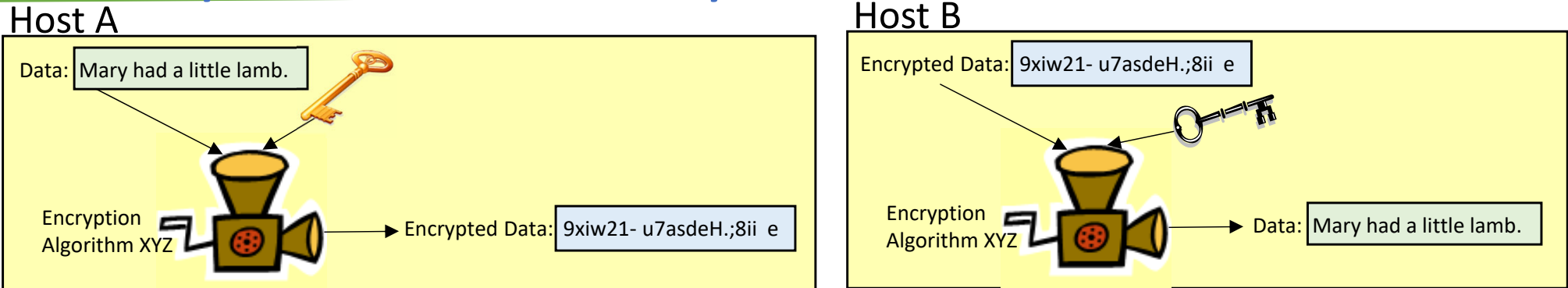


Host B



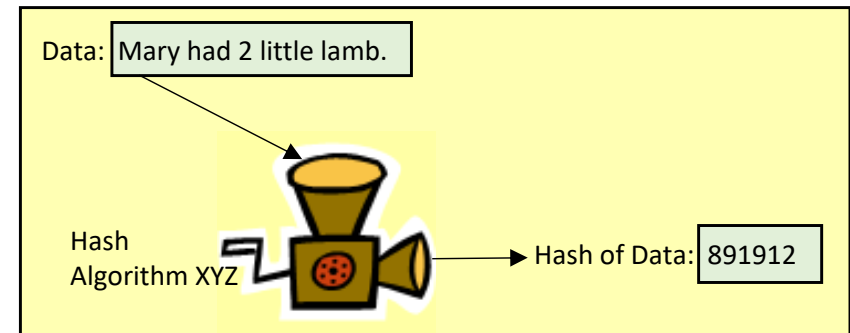
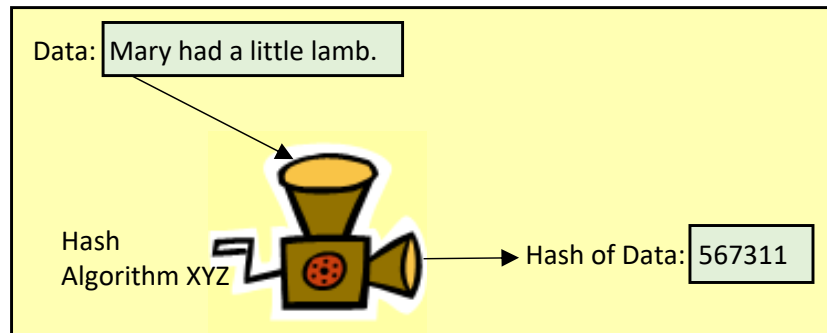
- Symmetric Keys are identical on both sides of the connection.
 - Host A has a piece of data and an encryption key.
 - Those are input to an encryption algorithm.
 - An encrypted piece of data is output.
 - Host A sends the encrypted data to Host B.
 - Host B inputs the encrypted data and the same key into the same encryption algorithm.
 - The output data is identical to the original data.
- The algorithms are publicly available.
 - Cryptographic algorithms are mentioned but algorithm description and details are outside the scope of these presentations. Web searches provide information on algorithms.
 - Also referred to as cipher suites.
- The key must be kept secret so that only Host A and Host B know what it is.
 - One partner creates the key.
 - How does that host securely send the key to the partner host?
- Symmetric keys are more efficient than asymmetric keys.
 - Less overhead for encryption.

Asymmetric Keys



- Asymmetric Keys are not identical but are mathematically related.
 - Data encrypted with one of the keys in the key pair can only be unencrypted with the other key.
 - Cannot be unencrypted with the first key!
 - But knowing one key, in no way helps guess what the second key is.
- Public/Private Key Pair
 - One of keys is kept by the owner and never shared. – The Private Key
 - The other key is given to everyone. – The Public Key
- Authentication
 - Host A inputs a piece of data and their private key into an encryption algorithm and sends the encrypted data to Host B.
 - Host B inputs the encrypted data and the public key into the same encryption algorithm and obtains the original data.
 - Everyone can read the message because everyone has the public key.
 - The message had to come from Host A because they are the only one with the private key.
 - Host A has been authenticated as the originator of the message. - Authentication
 - Host A cannot repudiate that they created the message. – Non-Repudiation
- Privacy
 - Host B inputs a piece of data and Host A's public key into an encryption algorithm and sends the encrypted data to Host A.
 - Host A inputs the encrypted data and their private key into the same encryption algorithm and obtains the original data.
 - The message was kept secret when it was sent over the network. Only Host A could read the message. – Data Privacy
- Public/Private Asymmetric Cryptography requires a starting point.
 - Host B has to trust that the public key they have is associated with Host A's private key.
 - How is this trust established?

Hash Algorithm



- Encryption
 - An encrypted message is the same length as the original message.
- Hash
 - Hash is a one direction operation. A hash value can not be changed back into the original message.
 - A hash is a fix length output value (smaller than the original message) that is not unique to the input but rare enough to be an added verification that the data has not been changed in transit.
 - Data Integrity Verification
- Encryption and Hash Usage Together
 - Host A takes the data message and creates a Hash value from it.
 - Host A appends the Hash value to the data message and then uses the Session key to encrypt the whole thing.
 - Host A sends the encrypted message to Host B.
 - Host B unencrypts the data using the Session key.
 - Host B takes the data message and creates a Hash value from it, using the same algorithm.
 - Host B compares the created Hash value to the one sent with the data message. If the two values are the same, then the message has not been changed in transit.
- Digital Signature
 - Sometimes it doesn't matter if the message is viewed by others. Sometimes it is only important to prove that the message hasn't been changed and to prove it came from the person who claims it came from.
 - Host A takes the data message and creates a Hash value from it.
 - Host A encrypts the Hash value using their Private key.
 - Host A appends the encrypted Hash value to the unencrypted data message.
 - Host A sends the unencrypted data message with the encrypted Hash value to Host B.
 - Host B uses Host A's Public key to unencrypt the Hash value, thereby proving the message came from Host A.
 - Host B takes the data message and creates a Hash value from it, using the same algorithm.
 - Host B compares the created Hash value to the one sent. If the two values are the same, then the message has not been changed in transit.

x.509 Digital Certificate

- X.509 is a standard format.
- Public/Private Key Pair Creation
 - An X.509 Certificate is created and issued along with the key pair.
 - Certificate establishes the credentials of the entity.
 - The public key is always attached to the certificate.
- Signed
 - Each certificate is signed by a Certificate Authority (CA) private key.
- Certificate Fields (some of them)
 - Serial Number – unique on the creation system
 - Algorithm – the algorithm that was used to sign the certificate
 - Issuer – CA that created the certificate (or CA certificate chain)
 - Does the partner need the immediate CA certificate or the full certificate chain for authentication?
 - Usually just the immediate CA certificate but it depends upon the application.
 - Valid Dates – Date range that certificate is valid
 - Subject Distinguished Name – the full name (ie. International Business Machines Incorporated)
 - Extensions – Optional fields that may be used for authentication
 - Altname (nickname) (ie. IBM)
 - IP Address
 - Domain Name
 - Fully Qualified Host Name



Certificate Hierarchy

CA Certificate and Key Pair



Server Certificate and Key Pair

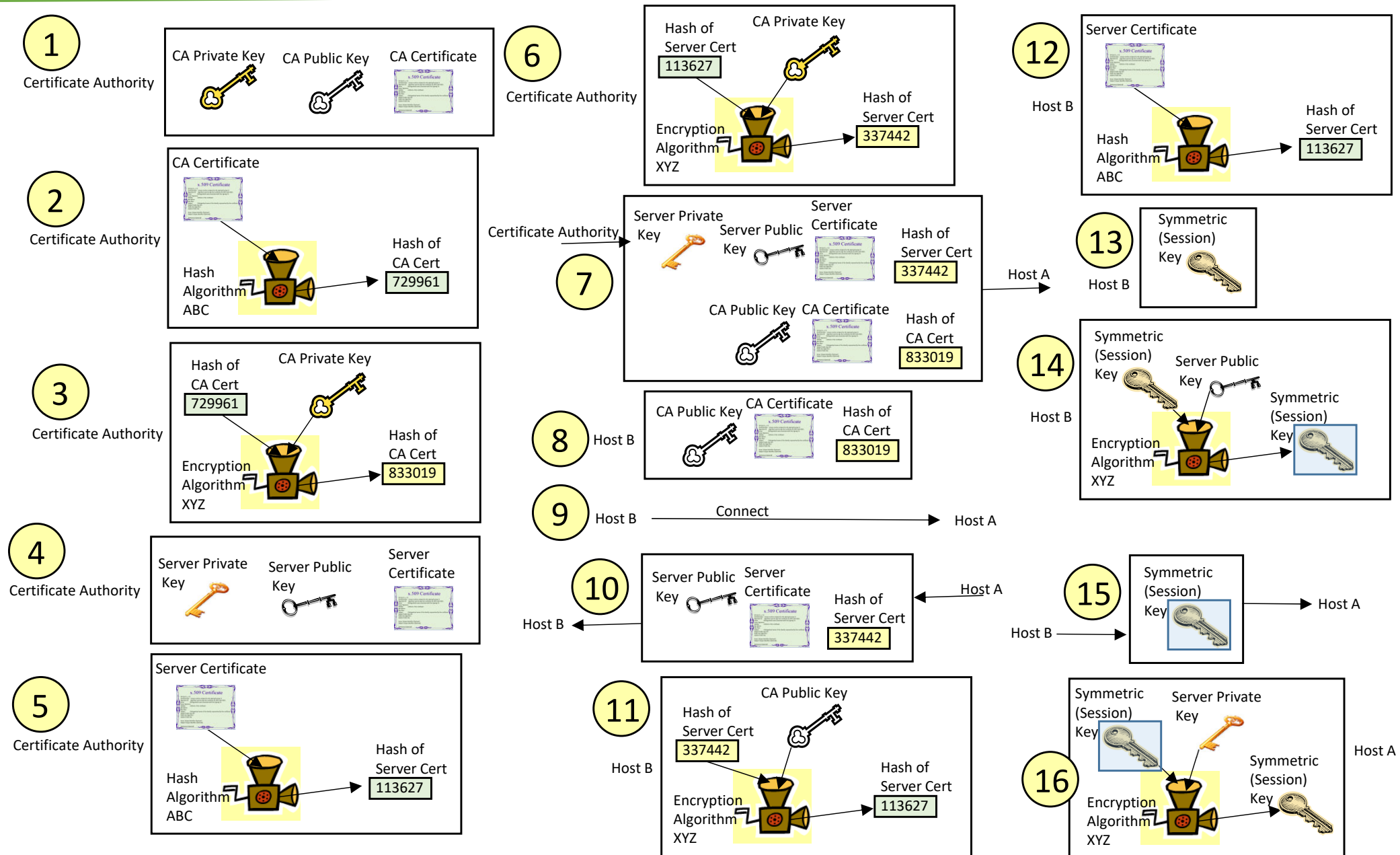


- Certificate Authority (CA) creates a CA certificate and associates it with a public/private key pair.
- CA signs the CA certificate with the private key.
 - Self-signed certificate.
- CA creates a certificate for an end entity and associates it with a public/private key pair.
 - End entity certificate is often referred to as a server certificate even though it could be used by a server or client.
- CA signs the Server certificate with the CA private key.
- As listed above there may be a single CA certificate, or the first CA certificate could have been used to create a second CA certificate that could have been used to create a server certificate, etc.
 - The hierarchy of the certificates is also referred to as a certificate chain.

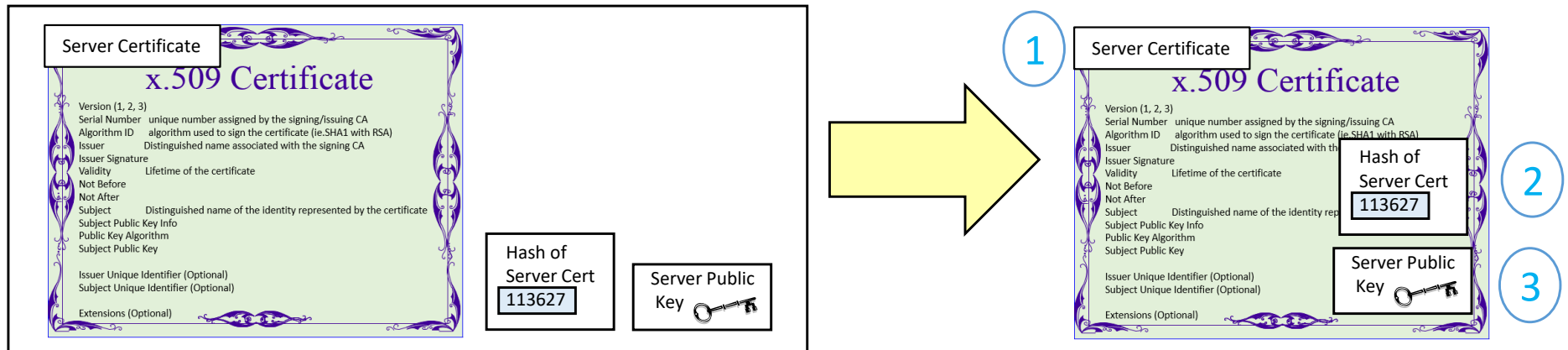
Asymmetric Keys Starting Point

- Well Known Certificate Authority (CA)
 - VeriSign
 - Thawte
 - etc.
- Starting Point - Well Known CA certificates are preloaded:
 - Internet Browsers:
 - Internet Explorer
 - Firefox
 - etc.
 - TN3270 Clients:
 - Pcomm
 - Host On-Demand
 - Rocket BlueZone
 - etc.
 - z/OS RACF
 - etc.

Asymmetric Keys and Certificate Usage



Certificate



- From this point forward in this presentation “Certificate” will refer to:
 1. The text document that contains the end entity’s information (referred to so far as the certificate)
 2. The Hash of the certificate signed by the CA’s private key
 3. The end entity’s public key. The three are bound together.

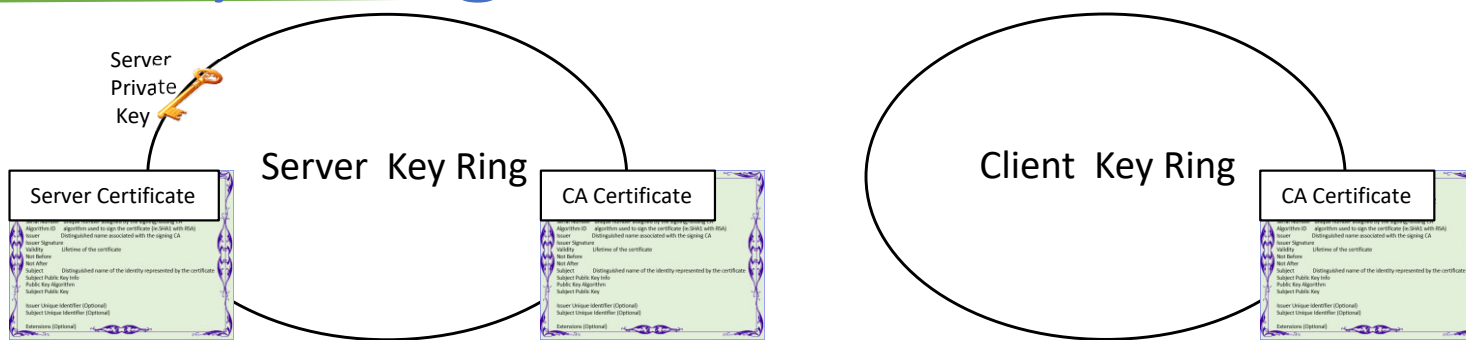
Digital Key Storage



Different Types of Digital Keys

- Key Functions:
 - Authentication and Non-Repudiation
 - Encryption
 - Data Integrity Verification through Hashing
- Key Types:
 - Symmetric and Asymmetric
- Common Networking Protocols that Use Keys
 - TLS
 - IPSec (Virtual Private Networks or "VPN")
 - Secured Shell ("SSH")
- Where Can Keys Be Stored? How Are They Distributed?
 - In a Configuration File (Definition File) - Password
 - OSPF in OMPRoute (Dynamic Routing Protocol)
 - OSPF uses a "password" that is used to build an MD5 Hashing value -- not really a key, but used for authentication of partner.
 - Manual Tunnels for VPNs (Symmetric Keys)
 - Dynamic Tunnel with Pre-Shared Keys (Symmetric Keys)
 - In a Generated Key File and "known hosts file"
 - SSH (Uses Asymmetric, Public/Private Keys to generate Session Key)
 - Bound to an x.509 Certificate: Public Asymmetric Key
 - With x.509 Certificates the implementation is called Public Key Infrastructure (PKI)
 - In a Key Ring or Key Database: Private and Public Asymmetric Key
 - In Cryptographic Hardware (Master Key)
 - Not recommended to store Private key in hardware, but rather in PKCS (Public-Key Cryptography Standard) dataset for ICSF.

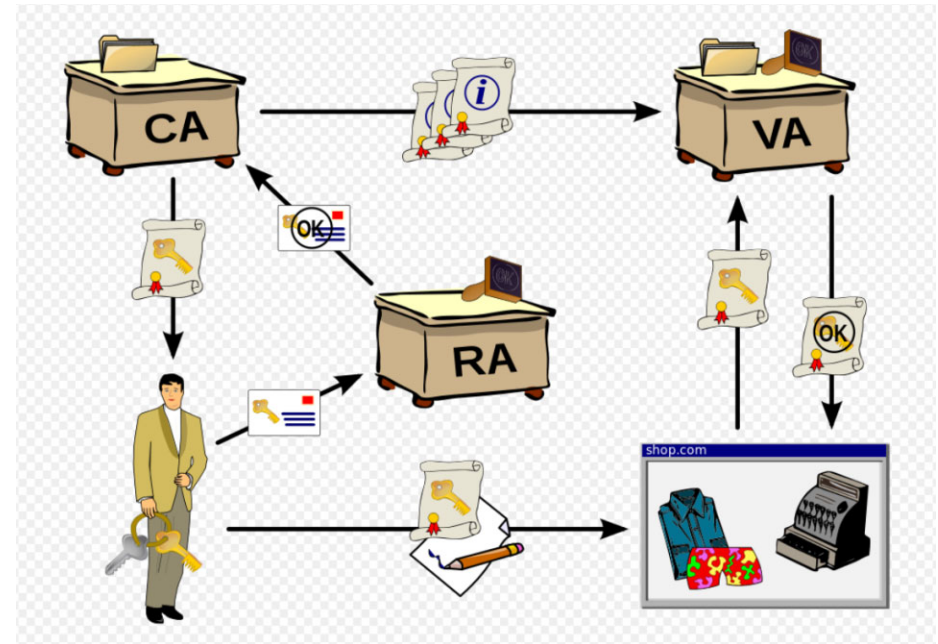
Key Ring



- Certificates and Keys may be stored in Key Repositories
 - Key Ring, Key Database, on a card, on a chip, or in an area of storage on a computer platform
 - IPsec only supports RACF Key Ring repository
 - AT-TLS supports both RACF Key Ring and Unix gskkyman repository
 - Unix gskkyman is outside the scope of this course.
- Key rings are defined in RACF
 - Provides access to some of the certificates and associated private keys loaded in the RACF database.
 - Certificate must be “marked” (defined) as TRUSTED.
 - Certificates with date ranges outside of the current date are created as UNTRUSTED.
 - In order to use a certificate the current date must be in the date range defined in the certificate.
 - RACF is the preferred certificate storage for z/OS Communications Server applications
 - NOTE: IKE with RSA Signature Mode requires a SAF Keyring; other key repositories are not supported.
- Keep Track of certificate Expiration Dates
 - RACF ICETOOL at www.racf.co.uk
 - Unload the RACF database and then run the icetool to produce a report on certificate expiration dates.
 - Health Check RACF_CERTIFICATE_EXPIRATION
 - Identifies expired and near expired certificates.
 - PKI Services
 - See next page.

z/OS Cryptographic Services PKI Services

- Public key infrastructure (PKI) provides applications with a framework for performing the following types of security-related activities:
 - Authenticate all parties that engage in electronic transactions
 - Authorize access to sensitive systems and repositories
 - Verify the author of each message through its digital signature
 - Encrypt the content of all communications
 - z/OS Cryptographic Services PKI Services
 - Allows you to use z/OS to establish a PKI infrastructure and serve as a certificate authority
 - Allows you to use z/OS to issue and administer digital certificates
 - PKI Services application may request and obtain certificates through their Web browsers
 - Authorized PKI administrators approve, modify, or reject these requests through their Web browsers
 - Allows automatic approval for certificate requests from certain users and, to provide additional authentication, add host IDs, such as RACF user IDs, to certificates you issue for certain users
 - Allows creation of certificates for browsers, servers, and other purposes
 - Allows automatic certificate expiration notification
 - Allows certificate renewal via web browser
 - Supports certificate revocation lists
 - Supports the delivery of certificates through the Secure Sockets Layer (SSL) for use with applications that are accessed from a Web browser or Web server
 - Supports the delivery of certificates that support the Internet Protocol Security standard (IPSEC) for use with secure VPN applications or IPSEC-enabled devices
- The diagram illustrates a Public Key Infrastructure (PKI) system with three main components: CA (Certificate Authority), RA (Registration Authority), and VA (Validation Authority). The CA is shown as a desk with a folder and a document labeled 'CA'. The RA is shown as a desk with a computer monitor and a document labeled 'RA'. The VA is shown as a desk with a folder and a document labeled 'VA'. A person in a suit is shown interacting with the RA. The RA sends a request to the CA, which approves it (indicated by a document with 'OK' and a red stamp). The CA then issues a certificate to the person. The person uses the certificate to access a website (shop.com) via a web browser. The VA is also shown interacting with the website, likely for validation purposes.
- From Wikipedia <http://en.wikipedia.org>



From Wikipedia <http://en.wikipedia.org>

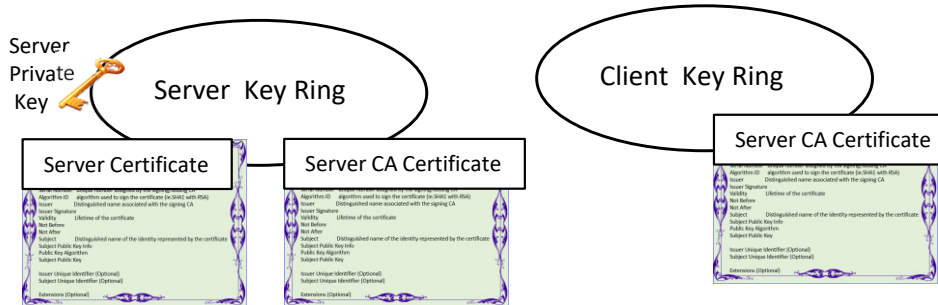
Certificate Usage and RACF Commands



Key Ring Contents

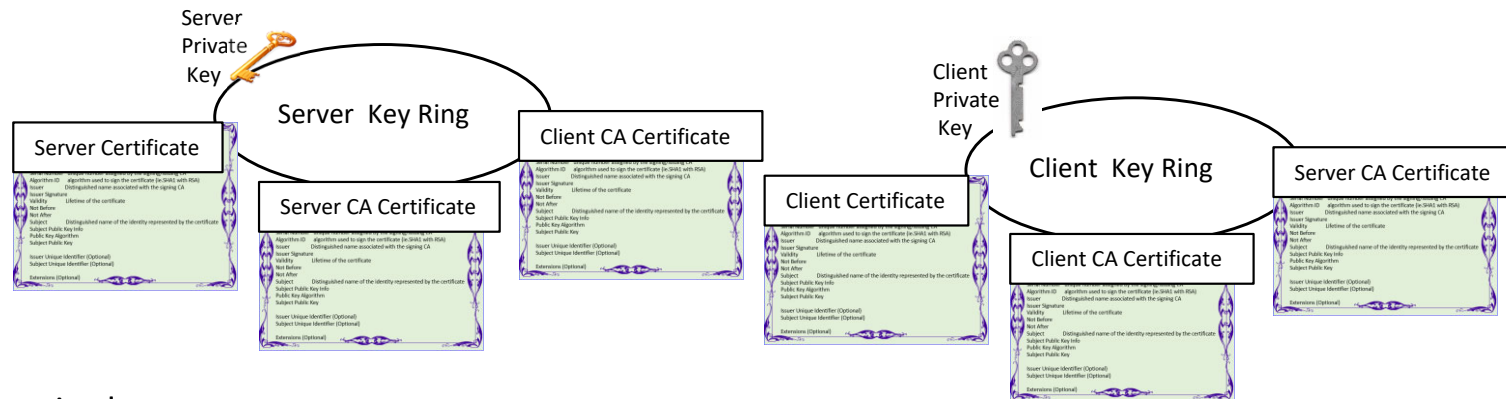
- AT-TLS Server Authentication is required.

- Server
 - Server Certificate
 - Server Private Key
 - Server CA Certificate
- Client
 - Server CA Certificate



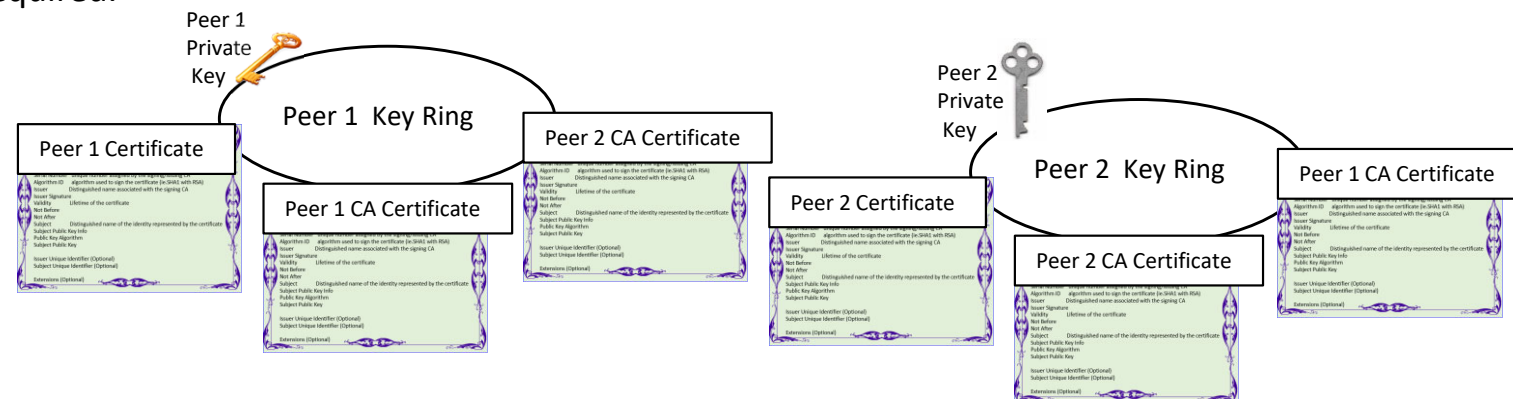
- AT-TLS Client Authentication is optional.

- Server
 - Server Certificate
 - Server Private Key
 - Server CA Certificate
 - Client CA Certificate
- Client
 - Client Certificate
 - Client Private Key
 - Client CA Certificate
 - Server CA Certificate



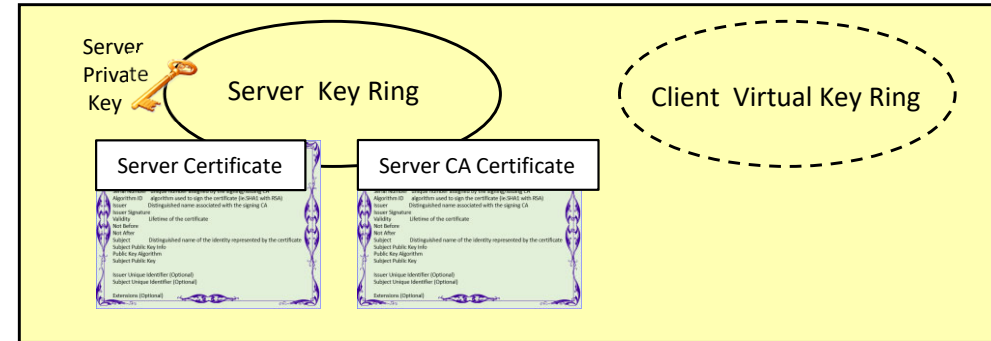
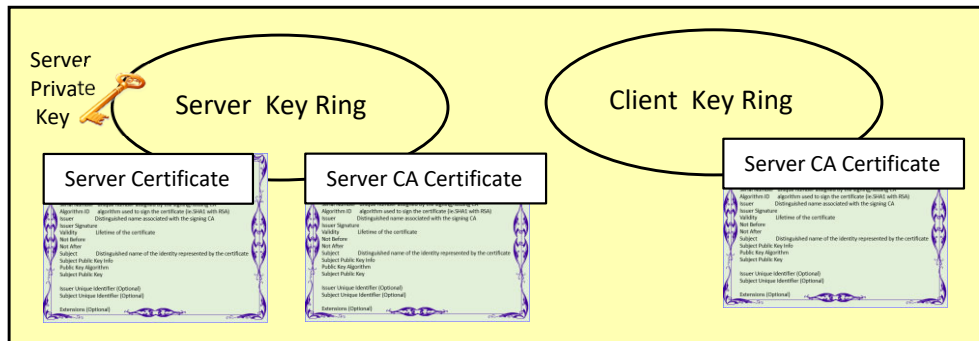
- IPsec Peer Authentication is required.

- Peer 1
 - Peer 1 Certificate
 - Peer 1 Private Key
 - Peer 1 CA Certificate
 - Peer 2 CA Certificate
- Peer 2
 - Peer 2 Certificate
 - Peer 2 Private Key
 - Peer 2 CA Certificate
 - Peer 1 CA Certificate



- Same certificates and key required as AT-TLS Server and Client authentication.

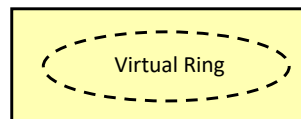
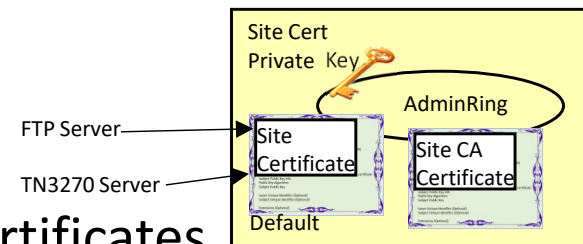
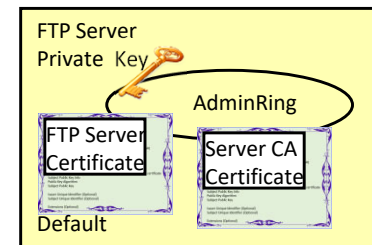
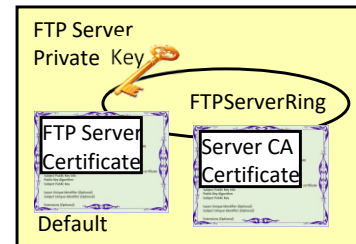
Key Ring and Certificate Types



- RACF Key Ring
 - Regular Key Ring
 - Key Ring is owned by a single User ID.
 - Virtual Key Ring
 - Access to all CA Certificates
 - Certificate must be marked as trusted.
 - Useful when ONLY CA Certificates are needed.
 - Client side when Server Only authentication is being used.
- RACF Certificates
 - When a certificate is connected to a key ring the private key is also attached if it is loaded in RACF.
 - Individual User or Personal Certificate (Server Certificate)
 - Only one owner
 - End Entity must “own” the certificate to use it.
 - Site Certificate
 - Supports multiple Owners
 - CA Certificate
 - Well-Known or Local CA

AT-TLS or IPsec Certificate Definition

- By default the DEFAULT certificate on the Key Ring is used.
 - To use a certificate that is not the default, the certificate Label Name must be specified.
 - The certificate Label Name is defined when the certificate is added to the certificate repository. Label Name is unique on the repository.
- Individual User or Personal Certificate (Server Certificate)
 - Certificate Must be Owned by User
 - Key Ring Owned by User
 - KEYRING FTPServerRing
 - Key Ring Not Owned by User
 - KEYRING ADMIN/AdminRing
 - Where userid ADMIN is the owner of Key Ring AdminRing
- Site Certificate
 - KEYRING ADMIN/AdminRing
 - Certificate shared by multiple applications.
- CA Certificate
 - Virtual Key Ring provides access ONLY CA certificates
 - KEYRING *AUTH*/*



Multiple Rules or Multiple Key Rings

- User ID must own certificate to use it.
 - Server or Client
- When Client Authentication is being used, is a separate key ring and a separate rule required for each client?
- User ID does not have to own the key ring.
 - Multiple Rules, one for each User ID
 - One rule specifies keyring FTPCL/FTPCLRing and label User21Cert
 - One rule specifies keyring FTPCL/FTPCLRing and label User31Cert
 - User ID FTPCL owns FTPCLRing keyring
 - When client User ID USER21 matches traffic then User21Cert is used.
 - When client User ID USER31 matches traffic then User31Cert is used.
 - Are you able to differentiate the traffic/rules?
 - Multiple Key Rings, all the same Key Ring name
 - keyring FTPClientRing
 - When User ID USER21 traffic matches the rule keyring FTPClientRing owned by User ID USER21 is used.
 - When User ID USER31 traffic matches the rule keyring FTPClientRing owned by User ID USER31 is used.

Certificate Packages

- Single Binary Certificate (Format is "CERTDER")
 - Coded as DER (Distinguished Encoding Rules), platform-independent format
 - Use: Used mostly for Certificate Requests, which are always DER-encoded and then base64-encoded, like base64-encoded certificates.
- PKCS#7 (binary package) (Format is "PKCS7DER")
 - One or more Certificates packaged together but not signed or encrypted
 - End Entity Certificate
 - Certificate Chain of Trust
 - Use: When the CA wants to deliver multiple certificates to a destination
- PKCS#12 (binary package) (Format is "PKCS12DER")
 - One or more Certificates packaged together, password-encrypted
 - End Entity Certificate
 - Certificate Chain of Trust (includes CA certificates if found)
 - Can Contain the Private Key if generated by the Certificate Authority
 - Use: When the CA wants to ship a package confidentially that contains the private key.
 - Use: To migrate certificates and keys from one platform to another.
- Base64-encoded Certificates (Format is "CERTB64" if no private key; PKCS12B64 if with private key and then it requires password-encryption)
 - Ascii, Text
 - Use: When making a Certificate Request in an email or if Certificate Management Facility on another platform requires ASCII format.

RACDCERT Command

- Define RACDCERT as an authorized TSO command in IKJTSOxx
- Define IRR.DIGTCERT.function resources using RDEFINE RACF command
 - function = ADD, ADDRING, ALTER, ALTPMAP, BIND, CONNECT, DELETE, DELMAP, DELRING, EXPORT, EXORTKEY, GENCERT, GENREQ, LIST, LISTMAP, LISTRING, MAP, REKEY, REMOVE, ROLLOVER
- To issue RACDCERT, user must have one of the following authorities
 - SPECIAL attribute
 - Sufficient authority to Facility class IRR.DIGTCERT.function resources
- Permit users to IRR.DIGCERT.function

Access Permission	Description
READ	control certificates for this user only
UPDATE	control certificates for other users too
CONTROL	control special certificates like CERTAUTH (Certificate Authority) certificates

- A SETROPTS command "refreshes" the controls for the certificate functions
- User executing this command requires the SPECIAL attribute

Some RACDCERT Commands

- DIGTCERT ADD - create a certificate using a data set
- DIGTCERT ADDRING - create a key ring
- DIGTCERT CONNECT - add certificate to key ring
- DIGTCERT DELETE – delete certificate
- DIGTCERT DELRING – delete key ring
- DIGTCERT EXPORT – write certificate to data set
- DIGTCERT GENCERT – create a certificate and optionally a public/private key pair
- DIGTCERT GENREQ – create a certificate request
- DIGTCERT LIST – list certificates
- DIGTCERT LISTRING – list key rings
- DIGTCERT REKEY - replicate (rekey) a digital certificate with a new public/private key pair
- DIGTCERT REMOVE – remove certificate from key ring
- DIGTCERT ROLLOVER - supersede one certificate (the source certificate) with another certificate (the target certificate)

RACF DIGTCERT Protection

- ADD and GENCERT

SIGNWITH	One's own certificate	Someone else's certificate	SITE or CERTAUTH certificate
SIGNWITH one's own certificate	READ authority to IRR.DIGTCERT.ADD and READ authority to IRR.DIGTCERT.GENCERT	UPDATE authority to IRR.DIGTCERT.ADD and READ authority to IRR.DIGTCERT.GENCERT	CONTROL authority to IRR.DIGTCERT.ADD and READ authority to IRR.DIGTCERT.GENCERT
SIGNWITH a SITE or CERTAUTH certificate	READ authority to IRR.DIGTCERT.ADD and CONTROL authority to IRR.DIGTCERT.GENCERT	UPDATE authority to IRR.DIGTCERT.ADD and CONTROL authority to IRR.DIGTCERT.GENCERT	CONTROL authority to IRR.DIGTCERT.ADD and CONTROL authority to IRR.DIGTCERT.GENCERT
SIGNWITH not specified	READ authority to IRR.DIGTCERT.ADD and READ authority to IRR.DIGTCERT.GENCERT	UPDATE authority to IRR.DIGTCERT.ADD and UPDATE authority to IRR.DIGTCERT.GENCERT	CONTROL authority to IRR.DIGTCERT.ADD and CONTROL authority to IRR.DIGTCERT.GENCERT

- ADD and ADDRING

- LIST and LISTRING

Function	READ	UPDATE	CONTROL
ADD	Add a certificate to one's own user ID.	Add a certificate for someone else.	Add a CERTAUTH or SITE certificate.
ADDRING	Create a key ring for one's own user ID.	Create a key ring for another user ID.	
LIST	List one's own certificate.	List the certificate of someone else.	List CERTAUTH or SITE certificates.
LISTRING	See one's own key ring.	See the key ring of someone else.	

RACF Granularity

- Access to all Key Rings
 - PERMIT IRR.DIGTCERT.LISTRING CLASS(FACILITY) ACCESS(UPDATE) ID(USER99)
- Access to a particular Key Ring
 - PERMIT USER99.USER99RING.LST CLASS(RDATALIB) ACCESS(READ) ID(USER99)
- Access to the Private Key
 - PERMIT USER99.USER99RING.LST CLASS(RDATALIB) ACCESS(UPDATE) ID(USER99)

Two z/OS Security Services manuals are critical to RACF functions and syntax:
z/OS Security Server RACF Security Administrator's Guide
z/OS Security Server RACF Command Language Reference

Certificate Creation



Generate Self Signed CA Cert

```
//GBGCAS21 JOB MSGCLASS=X,NOTIFY=&SYSUID
//GBGCAS21 EXEC PGM=IKJEFT01,DYNAMNBR=30,REGION=4096K
//SYSTSPRT DD SYSOUT=*
//SYSTSIN DD *
RACDCERT CERTAUTH GENCERT
        SUBJECTSDN (O('GBG')
                    CN('GBGCAS21.LABS.IBM.COM')
                    C('US'))
        ALTNAME (IP(192.168.20.102)
                DOMAIN('GBG.LABS.IBM.COM')
                EMAIL('ZOS@GBG.LABS.IBM.COM'))
        NOTBEFORE (DATE(2009-11-27))
        NOTAFTER (DATE(2010-06-27))
        KEYUSAGE (CERTSIGN)
        SIZE(1024)
        WITHLABEL('GBGCAS21 LABS Server CA')
```

Generate Server Certificate

```
//GBGSRV21 JOB MSGCLASS=X,NOTIFY=&SYSUID
//GBGSRV21 EXEC PGM=IKJEFT01,DYNAMNBR=30,REGION=4096K
//SYSTSPRT DD SYSOUT=*
//SYSTSIN DD *
RACDCERT ID(TN3270) GENCERT
    SUBJECTSDN (O('IBM')
                CN('TN3270.GBG.LABS.IBM.COM')
                C('US'))
    ALTNAME (EMAIL('TN3270@GBG.LABS.IBM.COM'))
    NOTBEFORE (DATE(2009-11-27))
    NOTAFTER (DATE(2010-06-09))
    WITHLABEL('TN3270 on MVS2')
    SIZE(1024)
    SIGNWITH(CERTAUTH
              LABEL('GBGCAS21 LABS Server CA'))
```

Export Certificate

```
//GBGX1221 JOB MSGCLASS=X,NOTIFY=&SYSUID  
//GBGX1221 EXEC PGM=IKJEFT01,DYNAMNBR=30,REGION=4096K  
//SYSTSPRT DD SYSOUT=*  
//SYSTSIN DD *  
RACDCERT ID(USER21) EXPORT(LABEL('USER21 on MVS2')) -  
    FORMAT(PKCS12DER) DSN('USER.CS.TEAM21.P12') PASSWORD('USERLABS')
```


Create Certificate with Request

1. Create Self Signed certificate as a place holder.

```
//GBGUPE21 JOB MSGCLASS=X,NOTIFY=&SYSUID
//GBGUPE21 EXEC PGM=IKJEFT01,DYNAMNBR=30,REGION=4096K
//SYSTSPRT DD SYSOUT=*
//SYSTSIN DD *
RACDCERT ID(USER11) GENCERT
            SUBJECTSDN (O('IBM')
                        CN('USER11.GBG.LABS.IBM.COM')
                        C('US'))
            ALTNAME (EMAIL('USER11@GBG.LABS.IBM.COM'))
            NOTBEFORE (DATE(2009-11-27))
            NOTAFTER (DATE(2010-06-09))
            WITHLABEL('USER11 on MVS1')
            SIZE(1024)
            SIGNWITH(LABEL('USER11 on MVS1'))
```



Create Cert with Request (cont.)

2. Create Certificate Request.

```
//GBGURQ21 JOB MSGCLASS=X,NOTIFY=&SYSUID
//GBGURQ21 EXEC PGM=IKJEFT01,DYNAMNBR=30,REGION=4096K
//SYSTSPRT DD SYSOUT=*
//SYSTSIN DD *
RACDCERT ID(USER11) GENREQ(LABEL('USER11 on MVS1'))
          DSN('USER11.CERT.REQ') -
```

3. FTP certificate request to Certificate Authority.

4. Sign certificate request.

```
//GBGUGN21 JOB MSGCLASS=X,NOTIFY=&SYSUID
//GBGUGN21 EXEC PGM=IKJEFT01,DYNAMNBR=30,REGION=4096K
//SYSTSPRT DD SYSOUT=*
//SYSTSIN DD *
RACDCERT ID(USER11) GENCERT('USER11.CERT.REQ') -
          NOTBEFORE(DATE(2011-01-07)) -
          NOTAFTER(DATE(2012-12-30)) -
          SIGNWITH(CERTAUTH LABEL('MVS1 LABS Certificate Authority'))
```

Add Certificate

5. Signed certificate request.

```
//GBGUAD21 JOB MSGCLASS=X,NOTIFY=&SYSUID
//GBGUAD21 EXEC PGM=IKJEFT01,DYNAMNBR=30,REGION=4096K
//SYSTSPRT DD SYSOUT=*
//SYSTSIN DD *
RACDCERT ID(USER11) ADD('USER11.CERT.B64')
```

Certificate with keys.

```
//GBGUAD21 JOB MSGCLASS=X,NOTIFY=&SYSUID
//GBGUAD21 EXEC PGM=IKJEFT01,DYNAMNBR=30,REGION=4096K
//SYSTSPRT DD SYSOUT=*
//SYSTSIN DD *
RACDCERT ID(USER11) ADD('USER11.CERT.P12') -
      TRUST WITHLABEL('USER11 on MVS1') -
      PASSWORD('USERLABS')
```

Rekey and Rollover Certificate

Rekey

```
//GBGURK21 JOB MSGCLASS=X,NOTIFY=&SYSUID
//GBGURK21 EXEC PGM=IKJEFT01,DYNAMNBR=30,REGION=4096K
//SYSTSPRT DD SYSOUT=*
//SYSTSIN DD *
RACDCERT ID(USER11) REKEY(LABEL('USER11 on MVS1'))
                SIZE(1024)
                NOTBEFORE(DATE(2009-11-27))
                NOTAFTER(DATE(2010-06-09))
                WITHLABEL('USER11 on ANY MVS'))
```

-
-
-
-

Rollover

```
//GBGURO21 JOB MSGCLASS=X,NOTIFY=&SYSUID
//GBGURO21 EXEC PGM=IKJEFT01,DYNAMNBR=30,REGION=4096K
//SYSTSPRT DD SYSOUT=*
//SYSTSIN DD *
RACDCERT ID(USER11) ROLLOVER(LABEL('USER11 on MVS1'))
                NEWLABEL('USER11 on ANY MVS'))
```

-

Host Name Check



Host Name Certificate Check

The image shows a 'Telnet3270' window with the 'Security Setup' tab selected. The 'Enable Security' checkbox is checked. Under 'Security Package', 'IBM Global Security Kit (GSKit)' is selected. The 'Security Protocol' dropdown is set to 'TLS1.0'. In the 'Server Authentication' section, the checkbox 'Check for Server Name and Certificate Name Match' is checked and highlighted with a black arrow. The 'Client Authentication' section has 'Send Personal Certificate to Server if it is Requested' unchecked. A pink callout box at the top right contains the text: 'PComm error message: Secure connection failed. The host name does not match the server certificate name.'

There is a problem with this website's security certificate.

The security certificate presented by this website was issued for a different website's address.

Security certificate problems may indicate an attempt to fool you or intercept any data you send to the server.

We recommend that you close this webpage and do not continue to this website.

- [Click here to close this webpage.](#)
- [Continue to this website \(not recommended\).](#)
- [More information](#)

End of Topic



End of Topic

