Using z/OS System SSL

Alyson Comer
IBM
comera@us.ibm.com

February 27, 2008
Session 5522

Trademarks

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

IBM®
MVS
RACF®
System z9
Z9
z/OS®

* Registered trademarks of IBM Corporation

The following are trademarks or registered trademarks of other companies.

Java and all Java-related trademarks and logos are trademarks of Sun Microsystems, Inc., in the United States and other countries.
Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.
Microsoft, Windows and Windows NT are registered trademarks of Microsoft Corporation.
UNIX is a registered trademark of The Open Group in the United States and other countries.
SET and Secure Electronic Transaction are trademarks owned by SET Secure Electronic Transaction LLC.

•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

- Introduction
- General SSL Overview
- What is System SSL
  - Certificates
    - How does System SSL use Certificates
    - Where can System SSL certificates reside
  - Enabling a Application to be protected through SSL
- Common Problem Areas
- System SSL use of hardware crypto
  - z800, z890, z900, z990, and z9
- Session Summary

What is SSL?

A well known and commonly used communication protocol that allows 2 communicating parties to communicate securely in a client/server relationship.

What is the SSL Protocol?

- Protocol is a set of rules governing how two applications will securely communicate over an open communications network. These rules are executed at the application level requiring the application to be designed to support SSL.

- SSL communications protocol was developed by Netscape® , Inc

- In 1999, Internet Engineering Task Force (IETF) standard RFC 2246 Transport Layer Protocol (TLS) V1.0 was defined.

- Most common use is between web browers and web servers but any kind of TCP/IP socket communication application is a candidate for SSL protection.

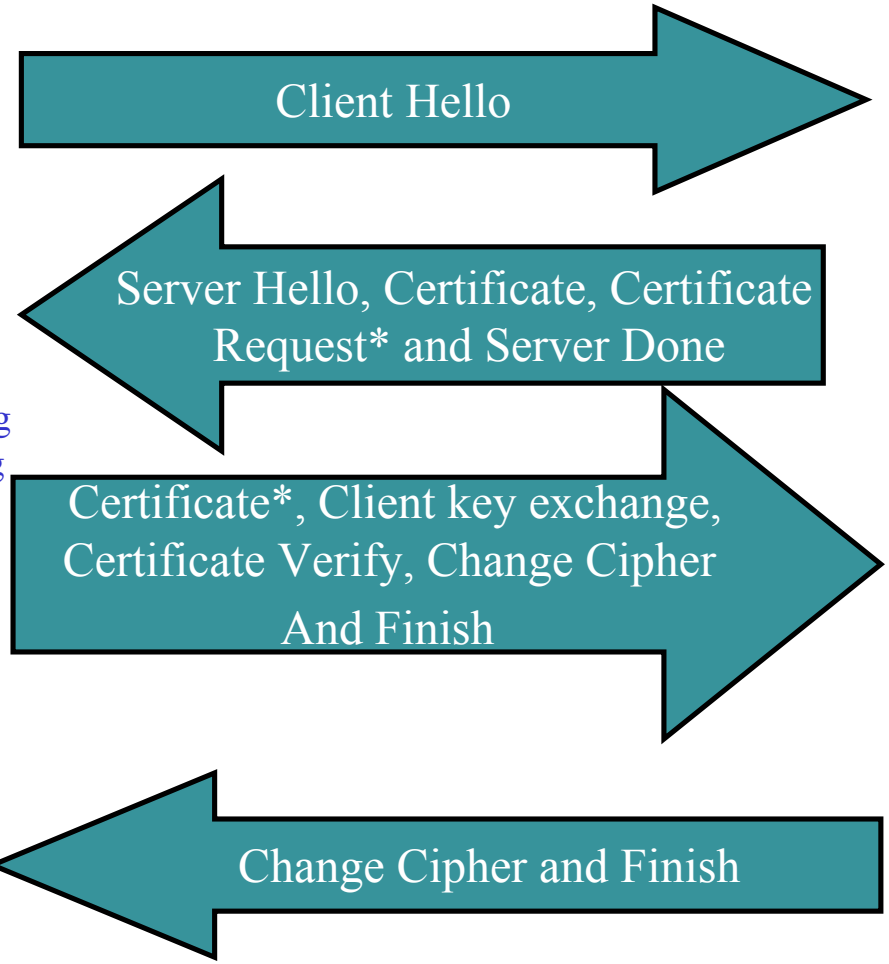
- Note: Throughout the presentation SSL refers to both SSL and TLS

How does SSL work?

Provides a way for applications to encrypt/decrypt data without having to have a previously agreed upon symmetric key. SSL provides each side information that is used to derive the same symmetric key. This information is security transferred during the "SSL Handshake" with the use of public key digital certificates.

- Digital certificates have two main purposes
  1. Contains information to securely identify someone
  2. Contains a public/private key pair known as asymmetric keys
     - –Key pairs work together to encrypt/decrypt information
     - –Public key is freely distributed and used by the sender to encrypt information
     - –Private key is used by the recipient to decrypt information
     - –The private key is never shared with anyone.

SSL Handshake



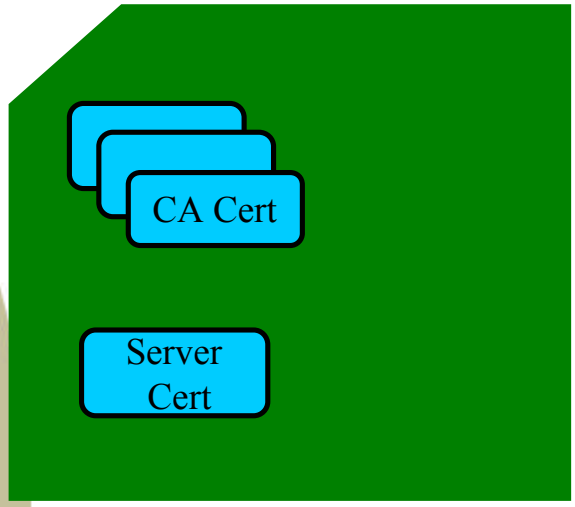
What is System SSL?

- An element of the z/OS base Cryptographic Services element that provides:
  - A certificate management utility, gskkyman, for managing certificates within a key database file as well as a suite of APIs to allow application writers the ability to write their own certificate management programs.
  - Applications a mechanism (suite of C/C++ POSIX callable application programming interfaces (APIS)) for applications to securely communicate over an open communications network (e.g. Internet
  - Although not part of the SSL protocol support, System SSL also contains a suite of APIs that allows for applications to build/read PKCS#7 messages.

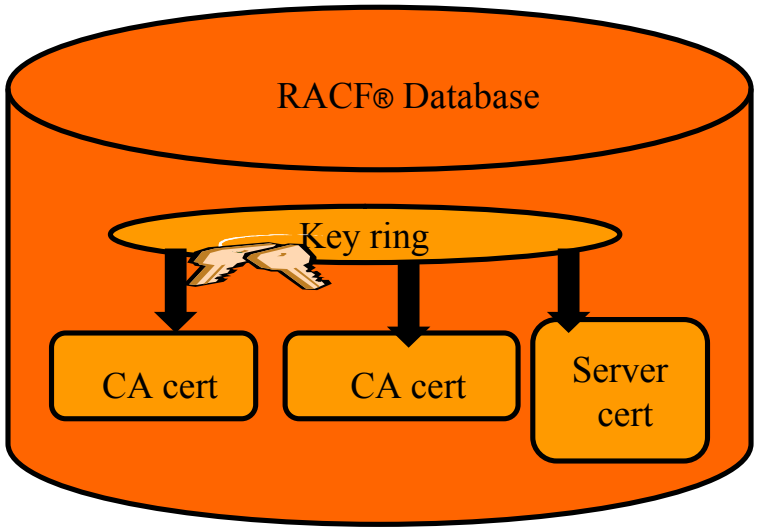
System SSL and Certificates

System SSL utilizes X.509 certificates managed through either the System SSL gskkyman utility or through the RACDCERT command.

GSKKYMAN



RACDCERT



Certificate Revocation Lists are stored in LDAP directories.

System SSL gskkyman

- Unix based utility executed from an OMVS environment

- Provides basic X.509 certificate management for certificates stored in a key database file
  - Creates/deletes HFS key database file
  - Store encrypted database password
  - Creates self-signed certificates (recommend not using in production)
  - Creates certificate requests (to be signed by trusted Certificate Authority)
  - Receives signed certificate requests
  - Setting default certificate
  - Imports/Exports certificates (with and without private keys)

Using a CA signed Server (Client) Certificate

- Server Application to use a certificate signed by a Certificate Authority
- Any certificate authority can be used as long as it supports X.509 V1, V2 or V3 certificates
  - For example
    - –z/OS PKI Services
- Steps:
  - Create a key database file and store its password
  - Create a new certificate request and send to CA
  - Receive CA certificate, if not already in the key database file
  - Receive signed certificate
  - Set as default (not required if application references the certificate by its label)
  - If client authentication, add client's signing certificate

Create key database file and associate a password with it

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

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.

Store key database file password to be used by an application

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): 10

Created file is called my.sth

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

Creating new certificate request continued

Certificate Type

1 - Certificate with 1024-bit RSA key
2 - Certificate with 2048-bit RSA key
3 - Certificate with 4096-bit RSA key
4 - Certificate with 1024-bit DSA key

Enter certificate type (press ENTER to return to menu): 1
Enter request file name (press ENTER to return to menu): certreq.arm ß **File to contain certificate request**
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

Creating new certificate request continued

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

Subject Alternate Name Type

1 - Directory name (DN)
2 - Domain name (DNS)
3 - E-mail address (SMTP)
4 - Network address (IP)
5 - Uniform resource identifier (URI)

Select subject alternate name type (press ENTER if name is complete): 2

Enter DNS name (press ENTER to return to menu): mycompany.com

Creating new certificate request continued

Contents of certreq.arm file:

-----BEGIN NEW CERTIFICATE REQUEST-----
MIIB3jCCAUcCAQAwczELMAkGA1UEBhMCVVMxETAPBgNVBAgTCE5ldyBZb3JrMREw
DwYDVQQHEwhFbmRpY290dDEMMAoGA1UEChMDSUJNMRMwEQYDVQQLEwpQcm9kdWN0
aW9uMRswGQYDVQQDExJTZXJ2ZXIgQ2VydGlmaWNhdGUwgZ8wDQYJKoZIhvcNAQEB
BQADgY0AMIGJAoGBAMTiaO7czZdi8IU+eCL23xtrqhXBqnksHBwdW8zeCjnqxq1l
ump9GY4Jw9Wyqp9a2J85bWJD06TaHhFALru5pgOl+jMOQTbB+wZoSOlbIrwoWl6l
pLx1cqJOn53mBmv6ruP/d055jjgKTczYhOa2JdhmfpAvf+C6tUkn7qMWlRzNAgMB
AAGgKzApBgkqhkiG9w0BCQ4xHDAaMBgGA1UdEQQRMA+CDW15Y29tcGFueS5jb20w
DQYJKoZIhvcNAQEFBQADgYEAAxCvLl4Cq+YVdJuHGnVr28ySnPz8E1uMT/k9Y6qM
EE+3Hiy2aD2mUREyeljehF5VNSbHwG5VCrFVVOtuVomeJgY8bYmlE45Z4oJoyqFG
HdQVUQO5E+W3UvKYv698KQTpl668BV51F3xlBwNx6K1PLl40i0fq8gFMfB8nP0KM
LOs=
-----END NEW CERTIFICATE REQUEST-----

Importing a signing Certificate Authority Certificate

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 Continued

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

Certificate imported.

Note: In order for the partner application to validate your certificate, it too must have a copy of the CA certificate.

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

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

File returned from CA

Marking a certificate as the default

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): 1

Marking a certificate as the default Cont'd

Key and Certificate List

Database:/tmp/my.kdb

1 - Server Certificate

0 - Return to selection menu

Enter label number (ENTER to return to selection menu, p for previous list): 1

Marking a certificate as the default Cont'd

Key and Certificate Menu

Label: New 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

Using RACF Key Rings for Certificates

Managed through the RACF RACDCERT Command

Certificates are group together into a “key ring”. Equivalent to a key database file.

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

Using RACF Key Rings for Certificates

RACDCERT ID(SUIMGTF) ADDRING(MyRACFKeyRing)

RACDCERT ID(SUIMGTF) 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(SUIMGTF) GENREQ(LABEL('Server Certificate')) DSN('suimgtf.highrisk.certreq')

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

RACDCERT ID(SUIMGTF) ADD('suimgtf.highrisk.signed') WITHLABEL('Server Certificate')

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

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

Using RACF Key Rings for Certificates

- In general, the server/client certificates must be owned by the application userid. Exception is SITE certificates
- Server/client certificates must be usage "personal" certificates
- Application ***must*** have read access to the IRR.DIGTCERT.LISTRING facility to read certificates connected to its key ring
- For server/client certificates to be shared (private keys), must be created as "site" certificates and applications needs "CONTROL" access to the IRR.DIGTCERT.GENCERT facility.
- To share key rings for certificate validation, the application userid must have “update” access to IRR.DIGTCERT.LISTRING

Using RACF Virtual Key Rings

- What is a Virtual Key Ring?
  - A set of all certificates associates with a given userid
- New in z/OS V1R8
- Three types
  - CERTAUTH - *AUTH*/*
    - Certificate Authority certificates owned by the CERTAUTH userid
  - SITE - *SITE*/*
    - General certificates owned by the SITE userid
  - PERSONAL - userid/*
    - Certificates owned by a particular user. i.e. COMERA

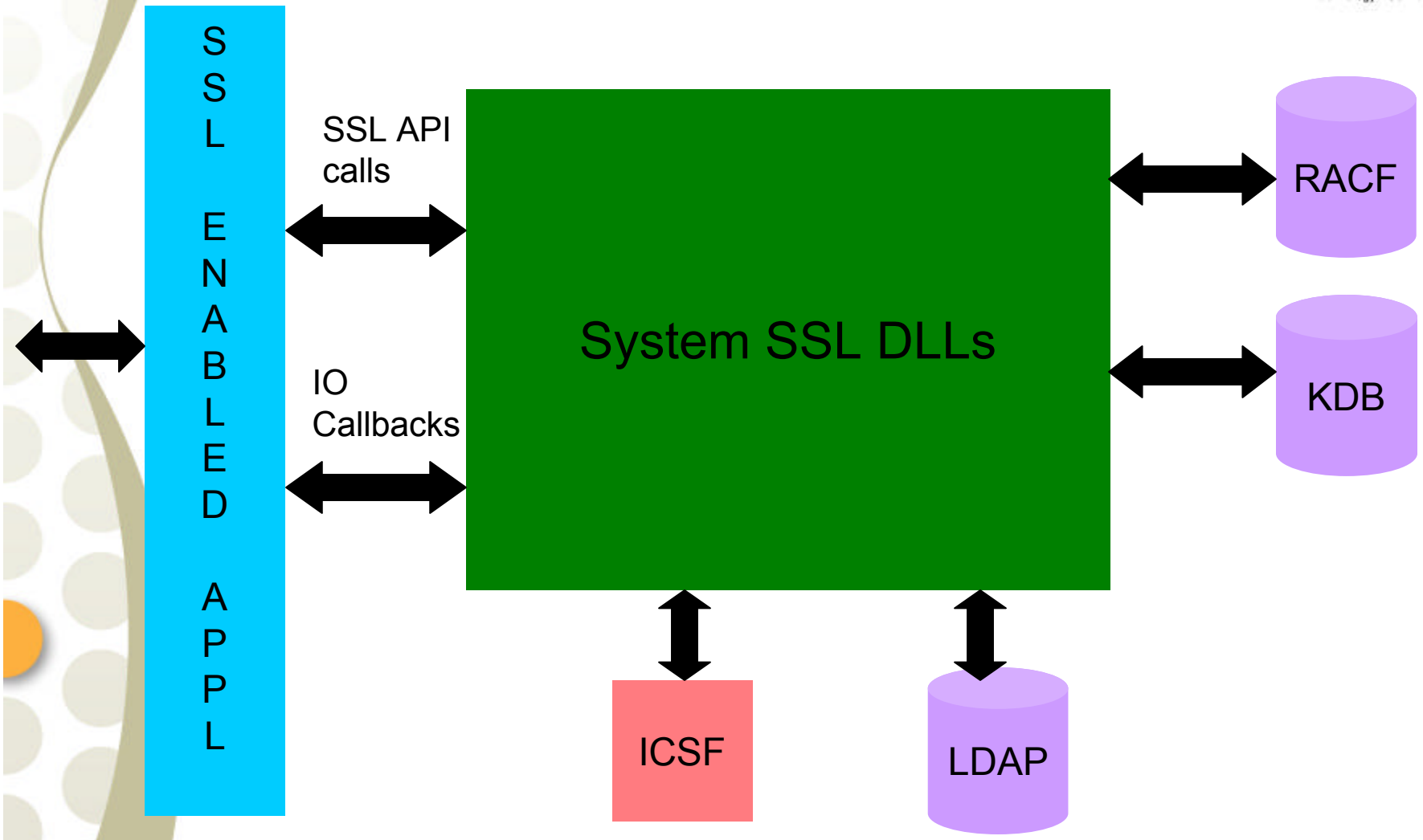
Using RACF Virtual Key Rings

- CERTAUTH and SITE virtual key rings will not allow access to the certificate private keys
- Application user ***must*** have read access to the IRR.DIGTCERT.LISTRING facility to read certificates connected to a CERTAUTH, SITE or its own virtual key ring.
- To share another user's PERSONAL virtual key ring, the application userid must have "update" access to IRR.DIGTCERT.LISTRING

Using a RACF Virtual Key Ring

- z/OS FTP client communicating with more than one FTP server performing server authentication only. Each FTP server is protected using a certificate from a different issuing Certificate Authority.
- Non-Virtual Key Rings for server authentication
  - Add certificate authority certificates to RACF database, connect to a key ring owned by the issuer of the client FTP, specify the RACF key ring in the ftp.data file for the user.
- Using a virtual CERTAUTH key ring
  - Add certificate authority certificates to RACF database, specify *AUTH*/* as the key ring in the ftp.data file for the user.
  - Additional server CA certificates can be added to RACF and no additional change is needed

System SSL Secure Communication APIs



System SSL Secure Communication APIs

Primarily 3 categories:

- SSL Environment Management
- SSL Connection Management
- Miscellaneous

SSL Environment Management

SSL environment consists of:

- Certificate repository definition
  - Key database file or SAF key ring
- Optional LDAP directory for CRLs
  - LDAP server name, port, userid and password
  - CRL caching timeout
- Session ID cache timeout
- Client authentication
- SSL/TLS protocols supported (SSL V2.0, SSL V3.0, TLS V1.0)
- List of ciphers supported (crypto algorithms)
- IO Callback routines (read,write, socket setting (blocking/nonblocking)

- N - number of secure connections

An application can have N number of concurrent environments.

SSL Environment Management APIs

- gsk_environment_open()
- Setting Environment Attributes
  - gsk_attribute_set_buffer()
  - gsk_attribute_set_enum()
  - gsk_attribute_set_numeric_value()
  - gsk_attribute_set_callback()
- gsk_environment_init()
- gsk_environment_close()

SSL Connection Management

A secure connection consists of the following attributes:

- Certificate label – (default)
- Supported ciphers (crypto algorithms)
- Supported protocols
  - *SSL V2.0, SSL V3.0 or TLS 1.0*
- File descriptor (socket)
- Client or Server

The secure connection involves the SSL handshake. SSL handshake results in each side of the secure connection to know what symmetric key is to be used for encrypting user data.

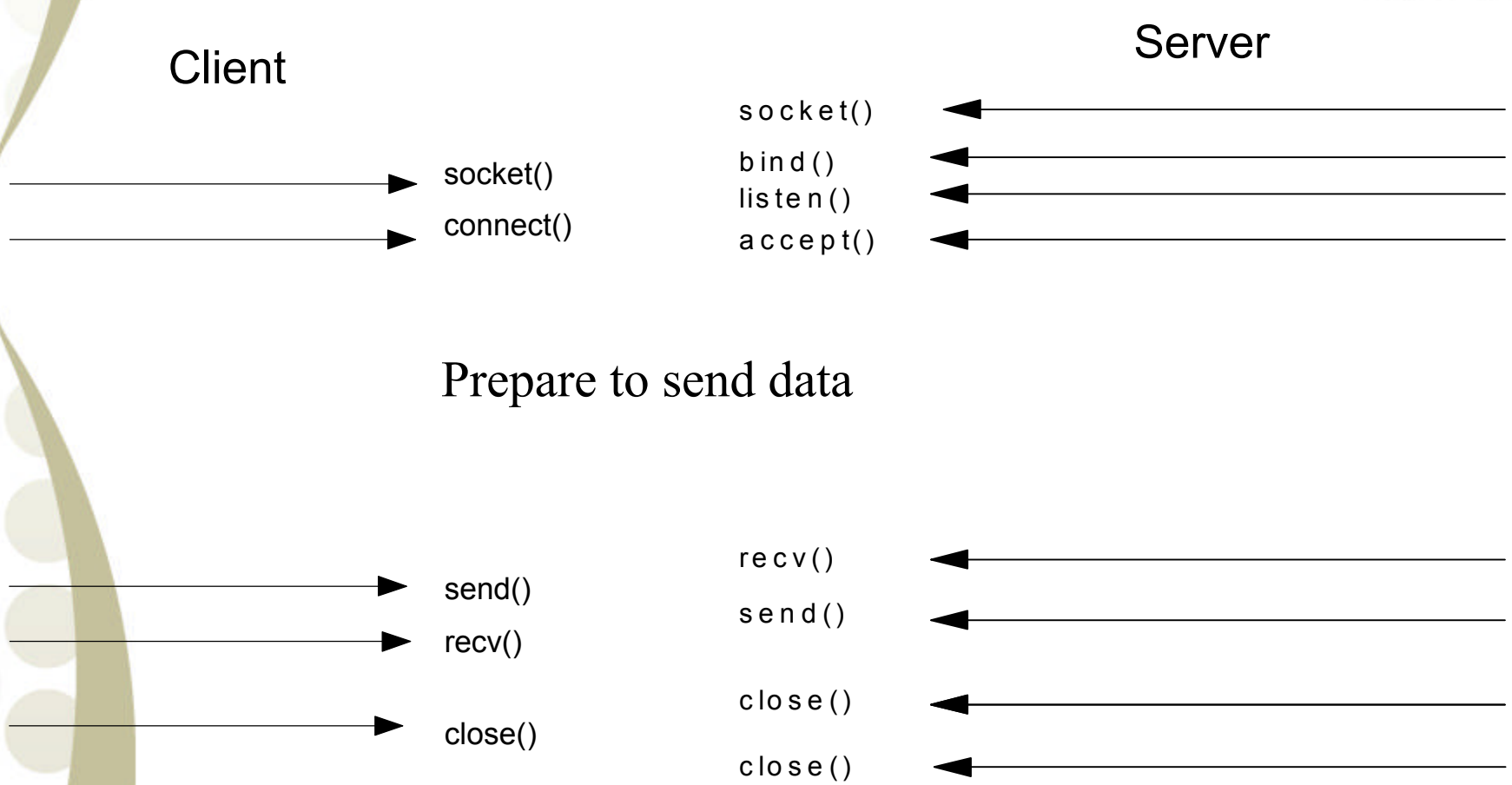
SSL Connection Management APIs

- gsk_secure_socket_open()
- Setting Connection Attributes
  - gsk_attribute_set_buffer()
  - gsk_attribute_set_enum()
  - gsk_attribute_set_numeric_value()
- gsk_secure_socket_init()
- gsk_secure_socket_read()
- gsk_secure_socket_write()
- gsk_secure_socket_shutdown()
- gsk_secure_socket_close()

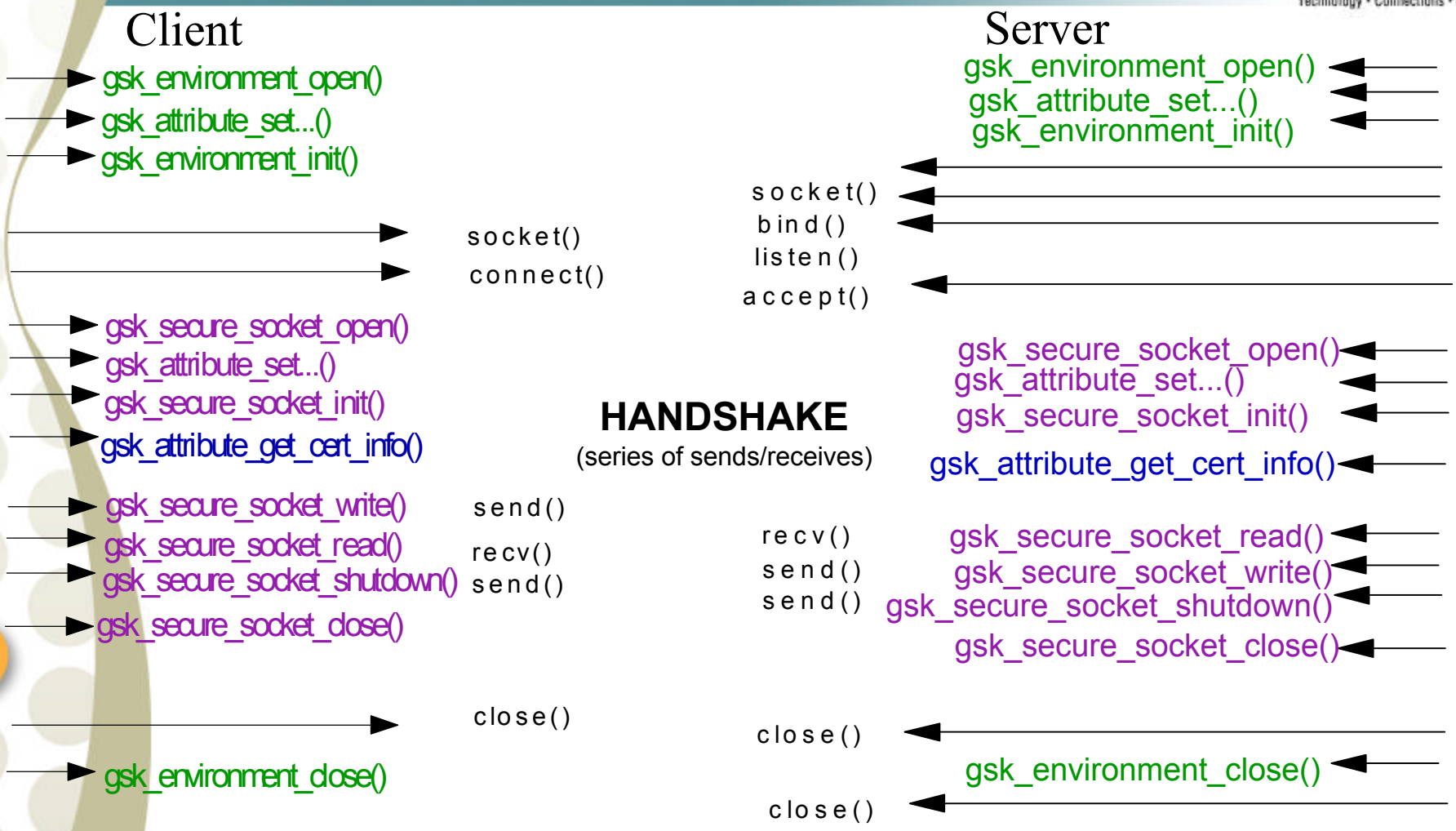
SSL Miscellaneous Management APIs

- gsk_attribute_get_buffer()
- gsk_attribute_get_enum()
- gsk_attribute_get_numeric_value()
- gsk_attribute_get_cert_info()
- gsk_get_cert_by_label()
- gsk_strerror()

Basic Socket Programming Model



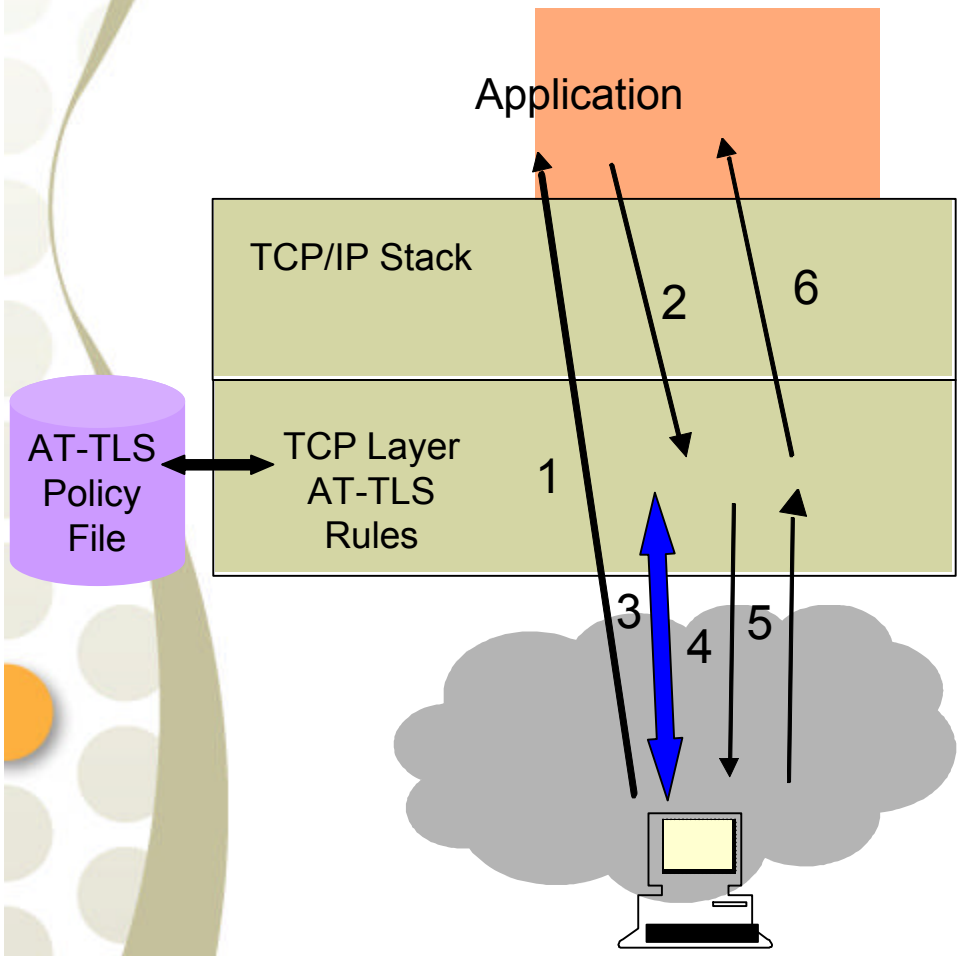
Basic Secure Socket Programming Model



Just a few words on AT-TLS

- AT-TLS Application Transparent TLS
- Introduced in z/OS V1R7
- A product of the z/OS Communication Server
- Provides SSL/TLS support at the TCP layer by using System SSL APIs
- Ideal for applications that do not require end to end security but security at the TCP/IP stack layer.
- Security Policy Files control the SSL/TLS protocol usage
  - Define the SSL environment and connection attribute values
  - Files can be either created using the IBM Configuration Assistant or manually
- Supports existing applications without change for basic SSL/TLS protection
- More advanced options (ie. Application requiring partners certificate for validation or first message of communication is not protected by SSL/TLS)
  - Application will need to program to the IOCTL SIOCTTLSCTL

Just a few words on AT-TLS



Configured AT-TLS Policy for the Application Server to use Transparent TLS:

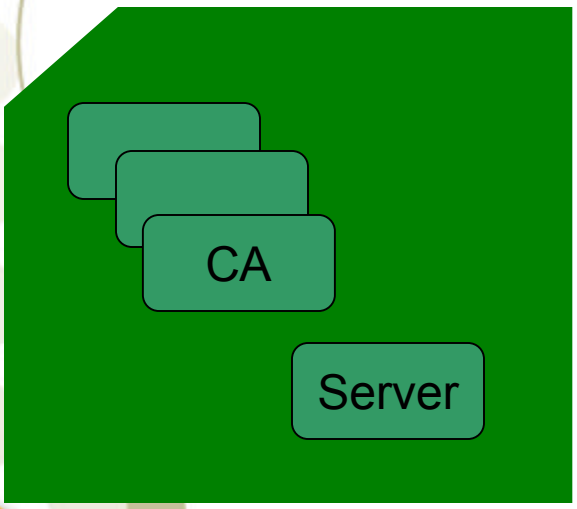
1. Client connects to server and connection becomes established
2. Server sends data in the clear and TCP layer queues it.
3. TCP layer invokes System SSL to perform SSL handshake under identity of the server.
4. TCP layer invokes System SSL to encrypt queued data and sends it to client.
5. Client sends encrypted data, TCP layer invokes System SSL to decrypt.
6. Server receives data in the clear.

Common Problem Areas (Gotchas)

Common Problem Areas

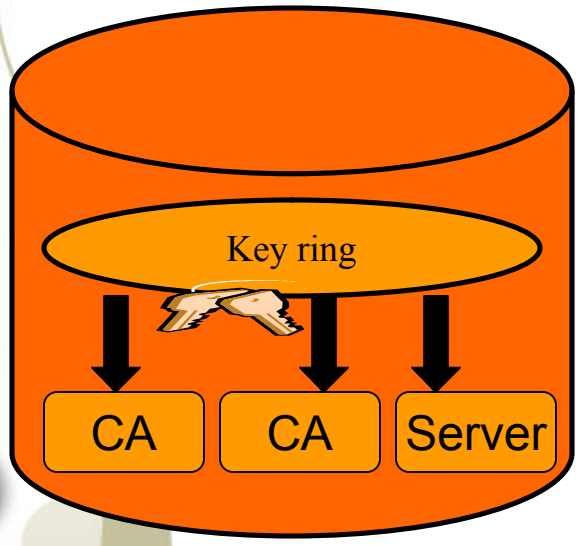
- Certificates
  - Using key database files
  - Using RACF key rings
- Common Programming Gotchas
- Knowing when hardware cryptography is being used

Using Key Database Files



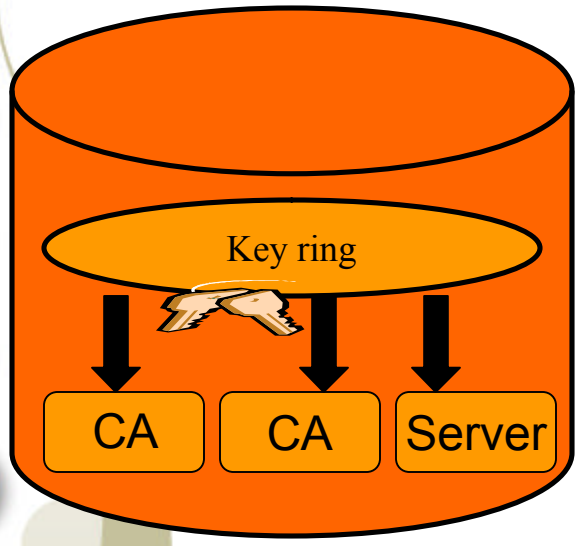
- Application must have read permission to key database file
- Not marking a certificate as the default
- Adding signed certificate requests – requires signing CA to be in database prior to receiving signed certificate.
- Creating self-signed certificate – care needs to be given to whether certificate is intended to be used as a server/client or certificate authority.
- Password or Stash file must be provided by the application using System SSL
  - Updating password and refreshing the key database stash file

Using RACF Key Ring Certificates



- Certificate Validation Failures
  - Default certificate authority certificates not marked TRUSTED (Note: Default CAs are not marked trusted)
  - Root Certificate not connected to key ring
- No Private Key
  - Private key associated with certificate stored in ICSF PKDS and ICSF started task not executing
  - Connecting user or server certificates to a key ring with a usage other than personal
  - Attempting to use a personal certificate own by another userid
- Sharing key rings among different userids
  - Certificate's private key not accessible for personal certificates and fails as a client or server certificate.
  - Using SITE certificates connected with usage SITE

Using RACF Key Ring Certificates



- Resource Access Failures
  - Application userid does not have READ/UPDATE access to the IRR.DIGTCERT.LISTRING
  - Using SITE certificates without appropriate access (CONTROL access to IRR.DIGTCERT.GENCERT)
- Processing Certificate Requests
  - Certificate authority not required to be in database prior to certificate being added.
  - Deleting certificate request and then receiving signed certificate request

Application programming Gotchas

- Application is responsible for I/O communication with partner application (blocking and non-blocking support)
- Application is responsible for enforcing client certificate was received when client authentication is requested
  - In V1R8, new environment attribute GSK_CLIENT_AUTH_ALERT will cause handshake to fail if client does not supply a certificate
- Application needs to inform partner when done sending data and secure connection is being closed
- POSIX LE environment

Using Hardware Crypto and Crypto Strengths

- Hardware crypto exploitation is transparent to the application but requires application supporters to be conscious of its use.
- Base System SSL FMID HCPT390 contains crypto up to 56-bit
- Security Level 3 System SSL FMID JCPT391 is required for 128-bit and greater crypto
- Larger the crypto key the greater the strength.
- Supported SSL crypto algorithms
  - Hash – MD5, SHA-1
  - Asymmetric – RSA, DSA
  - Symmetric – RC2, RC4, DES, 3DES and AES (128-bit & 256-bit)

Hardware Crypto Exploitation – z/OS V1R9

SSL Cipher	z800,z900 w/ CCFs RSA clear/encrypted key handshake operations in hardware	z890,z990,z9 – CPACF only RSA Clear key handshake operations in software	Z890,z990,z9 – CPACF and Crypto Express2 RSA clear key/encrypted key handshake operations in hardware
NULL/MD5	---	---	---
NULL/SHA-1	---	SHA-1	SHA-1
40-bit RC4/MD5 128-bit RC4/MD5 40-bit RC2/MD5	---	---	---
128-bit RC4/SHA-1 128-bit AES/SHA-1 256-bit AES/SHA-1	---	SHA-1 AES 128-bit (z9 only)	SHA-1 AES 128-bit (z9 only)
56-bit DES/SHA-1 168-bit 3DES/SHA-1	DES 3DES	DES/SHA-1 3DES/SHA-1	DES/SHA-1 3DES/SHA-1

How do I know hardware crypto is being used?

•Gsksrvr started task – Current available crypto used by System SSL through ICSF (Integrated Cryptographic Services Facility)

Query crypto operator command (f gsksrvr, d crypto)

Algorithm	Hardware	Software
DES	56	56
3DES	168	168
AES	128	256
RC2	--	128
RC4	--	128
RSA Encrypt	2048	4096
RSA Sign	2048	4096
DSS	--	1024

How do I know if hardware crypto is being used cont'd

- Diagnostic Environment Variable
  - GSK_SSL_HW_DETECT_MESSAGE
- SSL Product Trace
  - Environment Variables – GSK_TRACE and GSK_TRACE_FILE
  - Component Trace – Prior to z/OS V1R8 gsksrvr was required to be started prior to the application being traced.

How do I know hardware crypto is being used on z/OS V1R9

- GSK_SSL_HW_DETECT_MESSAGE=1
  - Written to standard out when the SSL DLL is being loaded
  - z800 or z900 without ICSF running
    System SSL: The crypto assist facility is not installed
    System SSL: ICSF services are not available
  - z800 or z900 with ICSF running
    System SSL: The crypto assist facility is not installed
    System SSL: ICSF FMID is HCR7740
    System SSL: CCF with a valid master key is available
    System SSL: DES CCF support is available
    System SSL: DES3 CCF support is available
    System SSL: PCI cryptographic coprocessor is not available
    System SSL: PCI cryptographic accelerator is available
    System SSL: Public key hardware support is available

How do I know hardware crypto is being used on z/OS V1R9

- Z890,z990, or z9 without ICSF running, CPACF available

  System SSL: SHA-1 crypto assist is available
  System SSL: DES crypto assist is available
  System SSL: DES3 crypto assist is available
  System SSL: ICSF services are not available

- Z890,z990, or z9 with ICSF running, CPACF available

  System SSL: SHA-1 crypto assist is available
  System SSL: DES crypto assist is available
  System SSL: DES3 crypto assist is available
  System SSL: ICSF FMID is HCR7740
  System SSL: PCI cryptographic accelerator is available
  System SSL: PCIX cryptographic coprocessor is available
  System SSL: Public key hardware support is available

How do I know if hardware crypto is being used on z/OS V1R9 through trace records

- Software Crypto Trace Records
  - Software DES encryption performed for n bytes
  - Software DES decryption performed for n bytes
  - Software 3DES encryption performed for n bytes
  - Software 3DES decryption performed for n bytes
  - Software RSA private key encryption performed
  - Software RSA private key decryption performed
  - Software signatures verified
  - Software signatures generated

How do I know if hardware crypto is being used on z/OS V1R9 through trace records

- **z800 and z900 – symmetric algorithms trace records**
  - CCF DES encryption performed for n bytes
  - CCF DES decryption performed for n bytes
  - CCF 3DES encryption performed for n bytes
  - CCF 3DES decryption performed for n bytes
- **Z890, z990, and z9 – symmetric algorithms trace records**
  - Clear key DES encryption performed for n bytes
  - Clear key DES decryption performed for n bytes
  - Clear key 3DES encryption performed for n bytes
  - Clear key 3DES decryption performed for n bytes
- **z800, z890, z900, z990, and z9 – asymmetric algorithms trace records**
  - Hardware RSA private key encryption performed
  - Hardware RSA private key decryption performed
  - Hardware signatures verified
  - Hardware signatures generated

How do I know if processing has switched from hardware to software

- In z/OS V1R9 notification support has been added
- Requires gsksrvr started task to be operational prior to the System SSL applications.
- Message sent to console and system log informing switch has occurred.
- Console Message:
  - GSK01051E *jobname/ASID* - Hardware encryption error. ICSF hardware encryption processing is unavailable
- System Log Message
  - GSK01052W *jobname/ASID* - Hardware encryption error. *algorithm* encryption processing switched to software

Publication/Reference Information

SC24-5901 - z/OS Cryptographic Services System Secure Sockets Layer Programming

SA22-7687 – z/OS Security Server RACF Command Language Reference

SSL C++ sample client/server applications
/usr/lpp/gskssl/examples/

SC31-8775 – z/OS Communication Server IP Configuration Guide

SC31-8776 – z/OS Communication Server IP Configuration Reference

Questions ?