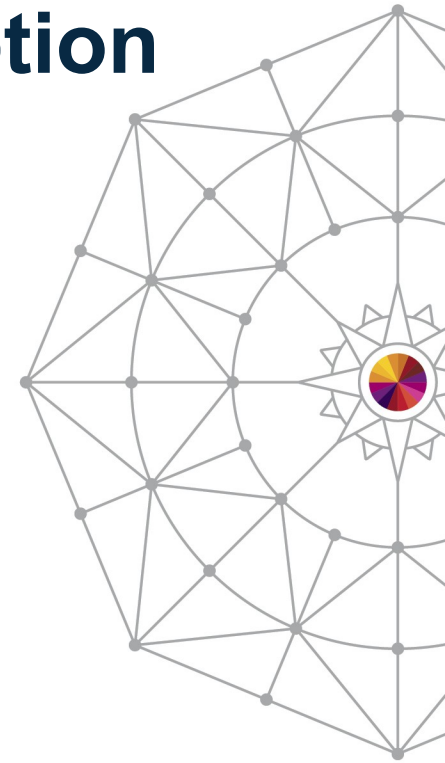
Sharing Secrets using Encryption Facility - Hands-on Lab

Steven R. Hart, CISSP®
IBM

Wednesday, August 6, 2014: 8:30 AM-9:30 AM
Session Number 15795



#SHAREorg

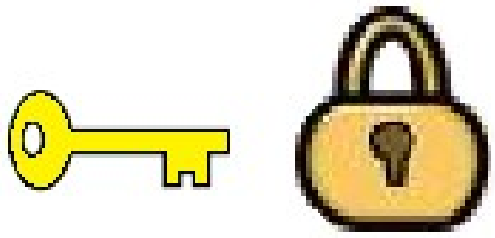
Topics

- Product Summary
- OpenPGP Support
- Digital Certificates
- Keystores
- Using zAAPs
- JZOS Batch Launcher
- Exchanging Encrypted Data
- Recent Enhancements

Encryption Facility for z/OS

- Encryption Facility for z/OS is a host based software solution for data encryption

- Encryption Facility for z/OS provides services for:
  - Public-key based encryption
  - Passphrase-based encryption
  - Modification detection of encrypted data
  - Compression of packaged data before encryption
  - Importing and exporting of OpenPGP certificates
    - Binary or ASCII armor format
  - Digital signatures of data



Encryption Facility for z/OS…

- Encryption Facility encrypts sensitive data for secure archival and business partner exchange using either a public key or passphrase

- Encrypted data is written to disk or other removable media
- Data can be encrypted in binary or text mode format.
  - Text mode translates the input data to UTF-8 or a local code page.
- Separately licensed product that does NOT come with ICSF or the z/OS base
- Option between 2 encrypted message formats
  - IBM Proprietary Encrypted Message format
    - Provides high performance for z to z encrypted data exchange
  - OpenPGP Message format
    - Provides compatibility between z and distributed systems for encrypted data exchange

EF OpenPGP Support

- OpenPGP is a widely accepted, open standard for handling encryption of data files and messages. OpenPGP is:
  - A file format for exchanging encrypted data. It is not an encryption algorithm.
  - Is standardized by the IETF and has been adopted as the format for encrypted files by a wide-range of open-source and commercial products
  - Does not define how encryption keys should be managed
  - Allows a wide range of choices for key exchange and trust model
  - Provides additional flexibility and interoperability for tape exchanges with external business partners and vendors
  - Allows customers to exchange an encrypted, compressed, and/or digitally signed file with business partners who have an installed OpenPGP client running on z/OS and other operating systems

EF OpenPGP Support…

- Provides at a minimum, support for all Mandatory/Must Do's identified in the OpenPGP standard (RFC 4880).
- This includes support for the following:

- **Public key encryption of session key**
- **Passphrase base encryption of session key**
  - string-2-key specification completed implemented
- **Digital signatures of data**
- **Compression of data**
- **Importing/exporting OpenPGP certificates**
- **RSA, ElGamal, and DSA key generation**
- **Use of partial data packets**
- **ASCII Armor for OpenPGP certificates**
- **Asymmetric Encryption Algorithms**
  - RSA
  - ElGamal

- **Symmetric Encryption Algorithms**
  - Triple DES
  - AES 128 bit keys
  - AES 192 bit keys
  - AES 256 bit keys
  - Blowfish
- **Compression Algorithms**
  - ZIP
  - ZLIB
- **Digest/Hash Algorithms**
  - SHA-1
  - MD5
  - MD2
  - SHA-256
  - SHA-384
  - SHA-512
- **Digital Signature Algorithms**
  - DSA w/ SHA1
  - RSA w/ all the supported hashes above

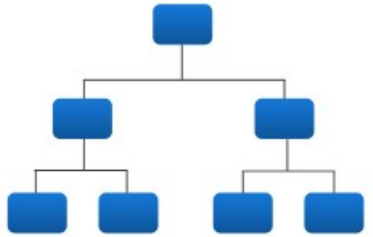
Legend/Notes:

GREEN: Functions that can use ICSF/H/W crypto. H/W crypto requires the correct environment and may require a Crypto coprocessor to be installed.

Digital Certificates

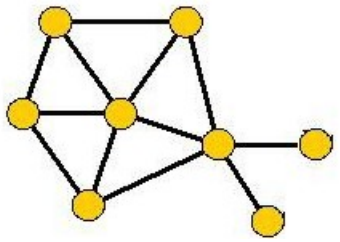
- Encryption Facility can use public-keys to encrypt data
- Public-keys are exchanged between business partners using digital certificates
- Digital certificates contain a public key, information to identify who the key belongs to, and a digital signature to authenticate and bind together the public key with the identity information.
- Digital Signatures are made by taking a hash of message, in this case the digital certificate, and then encrypting the hash value with the signers private key.
- Only the signer has the private key that was used to create the signature. Business partners must have the associated public key in order to verify the signature.

X.509 vs. OpenPGP Certificates



- X.509 uses a hierarchical authentication model
  - Each X.509 certificate contains 1 digital signature, either self signed or signed by a CA
  - A root Certificate Authority (CA) is established and trusted as a self signed certificate
  - Other certificates are signed by a CA within the hierarchy

- OpenPGP uses a decentralized authentication model
  - Each OpenPGP certificate can be self signed and can contain multiple signatures from other keys
  - OpenPGP sub-keys are all signed by the Primary key to bind together the Primary and sub-keys
  - Encryption Facility for z/OS provides an option to sign your OpenPGP certificates with a CA.



X.509 Certificates

- Encryption Facility can use both X.509 and OpenPGP certificates to encrypt data.

- X.509 Certificates use LDAP-style Distingished Names:
  "CN=Name, OU=Dept, O=Company, C=Country"

- Typical X.509 Certificates contain:
  - Version: The certificate version.
  - Serial Number: Unique identifier for the certificate.
  - Issuer: The entity that verified the information and issued the certificate.
  - Valid-From: The start date.
  - Valid-To: The expiration date.
  - Subject: The person, or entity identified.
  - Public Key: The public key.
  - Extensions: Used to store additional information.
  - Signature Algorithm: The algorithm used to create the signature.
  - Signature: The digital signature from the issuer.

OpenPGP Certificates

- OpenPGP Certificates use User Ids that can consist of a name, a comment, and an email address:

  “Name (Comment) <user@host>”

- Typical OpenPGP Certificates contain:
  - Public key packet: The primary public key for this certificate, creation date, expiration date, packet version, algorithm
  - User ID packet: Name, comment, and an email address.
  - Signature packets: Self signature binding the user ID to the Primary Key, signatures made by other keys, algorithm, Key ID, version number, creation date, signature class
    - Various attribute sub-packets such as preferred algorithms sub-packets, key flags, features and key server preferences.
  - Subkeys: Public sub-keys, version, algorithm, creation date, expiration date
  - Subkey Signatures: Signatures made by the Primary Key to bind the Subkeys to Primary Key.
    - Various attribute sub-packets such as preferred algorithms sub-packets, key flags, features and key server preferences.

Encryption Facility and ICSF

- What is the difference between EF and ICSF?

- ICSF (Integrated Cryptographic Service Facility) provides an Application Programmers Interface (API) for z/OS to access Cryptographic Features and the Cryptographic Key Data Sets

- In order to encrypt data with ICSF you need to:
  - Setup a key to be used for encryption
  - Write an application, utility, or script that calls ICSFs services to encrypt your data with a given key
  - Format the output encrypted data into a message that could be understandable to someone for decryption

- Encryption Facility is an application that can leverage ICSF's API's and easily perform all steps in the previous bullet

EF configured with ICSF and RACF

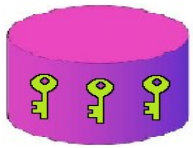
- EF configured with ICSF
  - When configured with ICSF, EF can generate new RSA key pairs in ICSFs PKDS
  - EF can also use pre-existing RSA key pairs in ICSF PKDS
  - EF can use and generate clear keys and secure keys (with the use of cryptographic coprocessors) in ICSFs PKDS
- EF configured with RACF
  - EF can use pre-existing RSA key pairs connected to a RACF Keyring
  - RACDCERT must be used to generate the RSA key pairs as EF only has read access of RACF keyrings
- EF configured with RACF and ICSF
  - EF can use pre-existing RSA key pairs connected to a RACF Keyring that point to private keys protected in ICSFs PKDS
  - RACDCERT must be used to generate the RSA key pairs as EF only has read access of RACF keyrings

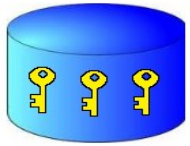
EF configured without ICSF and RACF

- EF configured without ICSF and RACF
  - EF can use the Java Cryptographic Extension (JCE) and a Java keystore within Unix System Services to perform software based cryptography.

Key Storage

- OpenPGP keyring
  - Keeps OpenPGP certificates
  - Keeps public keys
  - Resides in the Unix System Services file system





- Native Key store
  - Protected by either password or hardware master key depending on key store type, keeps private keys, can also keep public keys

z/OS	Java Keystore (JCEKS)	RACF Keyring (JCERACFKS)	ICSF PKDS (JCECCAKS)	RACF and ICSF (JCECCARACFKS)
Secure-key processing			✓	✓
Integration with existing audit and access control		✓		✓

Encryption Facilities use of Specialty Engines

- zAAP processors are specialty processors designed to lower overall computing cost by offloading certain types of workloads, such as Java workloads.
- The Encryption Facility for z/OS OpenPGP Support is implemented in Java.
- If a zAAP processor is available on the system, Encryption Facilities OpenPGP jobs will be off-loaded to those processors.
- When called from Java using the JCE HW provider, ICSF code remains zIIP and zAAP eligible.

Using JZOS with Encryption Facility

- The Encryption Facility OpenPGP support can be invoked from either a USS command prompt or from a batch job using JZOS.
- JZOS is a facility within IBM Java that provides the ability to launch Java applications from JCL.
- JZOS invocation samples are provided with the Encryption Facility V1.2 product and within the User's Guide.
- The samples consist of three different files:
  1) Procedure in PROCLIB
  2) Shell script to configure environment variables
  3) Batch job that calls the sample procedure in PROCLIB

Sample JCL that uses the Java batch program and environment script to encrypt data with EF

```
/*
//JAVA3 EXEC PROC=CSDJZSVM,VERSION='50'
//STDENV DD DSN=<HLQ>.JZOS.JCL(CSDSMPEN),DISP=SHR
//*
//DDDEF DD DSN=HLQ.EFR2.ENC.OUT2,
// DISP=(NEW,CATLG),
// DCB=(RECFM=VB,LRECL=32756,BLKSIZE=32760),
// UNIT=SYSALLDA,
// SPACE=(CYL,(5,1))
//*
//MAINARGS DD *
-homedir /etc/encryptionfacility/
-o 'DD:DDDEF'
-rA rsa_md2_4096
-keystore /var/encryptionfacility/keystores/encrdecr/keystore_jceks
-keystore-type JCEKS
-keystore-password password
-key-password password
-t 'UTF-8'
-e '//HLQ.EFR2.INPUT(CLRTXT)'
/*
```

EF OpenPGP Command Syntax

All Encryption Facility for OpenPGP commands have the following syntax.

-homedir must appear before all the options, and all the options must appear before the commands:

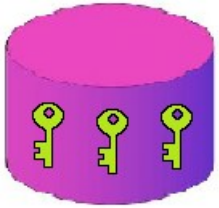
com.ibm.encryptionfacility.EFOpenPGP [-*homedir* name] | [*options*] *commands* [*arguments*]

where:

- ***homedir*** name is the name of the configuration file ibmef.config that contains specified options to use with the command.
- ***options*** is the name of one or more options to use on the command line and always starts with -. This option value overrides values in the configuration file.
- ***commands*** is the name of one or more commands and always starts with -.
- ***arguments*** specifies one or more targets of the command, for example, file name, certificate, alias, and so forth.

Using EF to Exchange Encrypted Data

z/OS



OpenPGP Keyring contains public keys

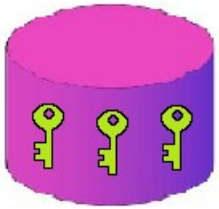
Either public keys, via digital certificates, or a passphrase is exchanged between business partners.

Each side can then encrypt data using their business partners public key or the passphrase. A symmetric session key is actually used to encrypt the data. When using public keys, the session key is randomly generated and wrapped by each public key separately. When using passphrase, the passphrase is used to generate the session key using an S2K algorithm.

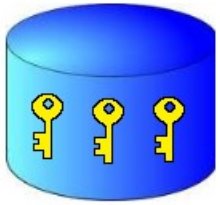
Each side can now send their partner encrypted data.

Each receiving side can then decrypt the data using their private key or the passphrase.

z/OS or Distributed System



OpenPGP Keyring



Private keystore

Recent Enhancements to the z/OS Encryption Facility

RFC 4880 Compatibility

- RFC 4880 is the Internet Standard for OpenPGP Message Format
  - http://www.ietf.org/rfc/rfc4880.txt
- EF's original OpenPGP support was based on RFC 2440
  - http://www.ietf.org/rfc/rfc2440.txt
- RFC 4880 replaces RFC 2440 making it obsolete
- The majority of OpenPGP products are RFC 4880 compliant
- In order to remain compatible with other OpenPGP products Encryption Facility has been upgraded it's OpenPGP support to RFC 4880

z/OS Encryption Facility -- Recent Enhancements

Speculative Key ID Support

- Encryption Facility has been enhanced to include Speculative Key ID support as described in RFC 4880.

- Speculative Key ID support allows the user to zero out the Key ID fields in their OpenPGP messages.

- By default encrypted messages contain the Key IDs of the public keys used to wrap the session key. When Key IDs are included in the encrypted message, an OpenPGP implementation can easily find the associated private key in the keystore.
  - The private key is then used to unwrap the session key, and then the session key is used to decrypt the encrypted data packet.

- With Speculative Key ID support the Key IDs of the public keys are zeroed out.
  - Zeroing out the Key IDs removes the risk of the Key IDs being intercepted by an unauthorized user performing traffic analysis.

- When the Key IDs are zeroed out, an OpenPGP implementation must try to decrypt the message with configured private keys in order to determine the associated private key.

z/OS Encryption Facility -- Recent Enhancements…

Batch Key Generation and Batch Public Key Export

- Encryption Facility OpenPGP support has been enhanced to allow for Batch Key Generation and Batch Public Key Export.

- In the original implementation of Encryption Facilities OpenPGP support, the generate and export commands were executed manually from a command line.
  - Once invoked the user was presented with a series of questions on the command line prompt.

- With the addition of batch key generation and batch public key export, the generate and export commands can now be executed as a batch job
  - No longer requires interactive answers to questions on the command line.

Recent EF Enhancements…

Symmetrically Encrypted Integrity Protected Data Packet

- Support has been added to allow for multiple public key encrypted session key packets preceding a Symmetrically Encrypted Integrity Protected Data Packet.
- The original Encryption Facility OpenPGP implementation only supported 1 public key encrypted session key packet preceding a Symmetrically Encrypted Integrity Protected Data Packet.
- This new support allows multiple recipients to be specified when using the Symmetrically Encrypted Integrity Protected Data Packet.
- This support allows for digitial signatures generated by OpenPGP certificate sub-keys.

Support for Notation Data Sub-packets containing raw binary data

- Support has been added for Notation Data Sub-packets containing 'name' and 'value' data sections in raw binary format.
- The original Encryption Facility OpenPGP implementation only supported Notation Data Sub-packets containing 'name' and 'value' data sections in UTF-8 human-readable text format.

EF zEDC Support Overview

- zEnterprise Data Compression (zEDC) is a new compression acceleration capability that is available beginning with z/OS V2.1.
- The new zEDC capability includes a new I/O feature on the zEC12 or zBC12 known as the zEDC Express.
- zEDC enables you to do hardware based data compression using the industry standard zlib library.
- The IBM Encryption Facility for z/OS product has been updated so that zEnterprise Data Compression (zEDC) can be used for compression of OpenPGP messages
- Requires zEDC Express feature available on the system and IBM 31-bit SDK for z/OS, Java Technology Edition, Version 7 SR 7 or later.

EF zEDC Performance Boost

- Internal testing has shown that IBM Encryption Facility environments using existing software compression have received the following performance boost when configured with zEDC:

  - Up to 70% reduction in processor time
  - Up to 60% reduction in elapsed time

Disclaimer: Results based on internal controlled measurements using IBM Encryption Facility for files containing public domain books. Results may vary by customer based on individual workload, data, configuration and software levels.

EF zEDC Support – Added Features

Support for Hardware Compression of OpenPGP Messages

- Encryption Facility for z/OS® supports data compression in the OpenPGP message format
  - When using the passphrase based encryption (-c) command, public key encryption (-e) command, and sign (-s) command.
  - The Encryption Facility decrypt (-d) command and verify (-v) command support decompression of data in the OpenPGP message format.
- The -z/COMPRESSION command option is used to turn on compression when using the -c, -e, or -s commands.
- A compression algorithm name may be specified by using the -compress-name/COMPRESS_NAME command option.
  - Supported compression algorithms include ZIP and ZLIB which are provided by the IBM Java Software Development Kit (SDK).
- The -d and -v commands do not require a command option for compressed data. These commands automatically decompress data in the OpenPGP message format.

EF zEDC Support – Added Features

Hardware Compression of OpenPGP Messages continued

- zEDC requires a minimum input buffer size for compression and decompression.
- If the input data is smaller than the minimum threshold, the data is processed using traditional software-based compression and decompression.
- Default thresholds are 4K for compression and 16K for decompression.
  - These values may be overridden.
- If the input data is large enough, EF will use 324 KB input buffers for compression and 64 KB input buffers for decompression.

EF zEDC Support – Added Features

Compress Only Command

- Encryption Facility has been enhanced to provide a new command, -compress, that outputs a compressed only OpenPGP message.

- Previously Encryption Facility only performed compression before it produced an encrypted or signed OpenPGP message.

- With this new option Encryption Facility can be used to simply compress data in the OpenPGP message format without having to also encrypt or sign the data.

References & Publications

- SA23-2229 IBM Encryption Facility for z/OS: Planning and Customizing
- SA23-1349 IBM Encryption Facility for z/OS: User's Guide
- SA23-2230 IBM Encryption Facility for z/OS: Using Encryption Facility for OpenPGP
- Publications may be downloaded from the z/OS Internet Library
  - http://www-03.ibm.com/systems/z/os/zos/library/bkserv/v2r1pdf/#CSD