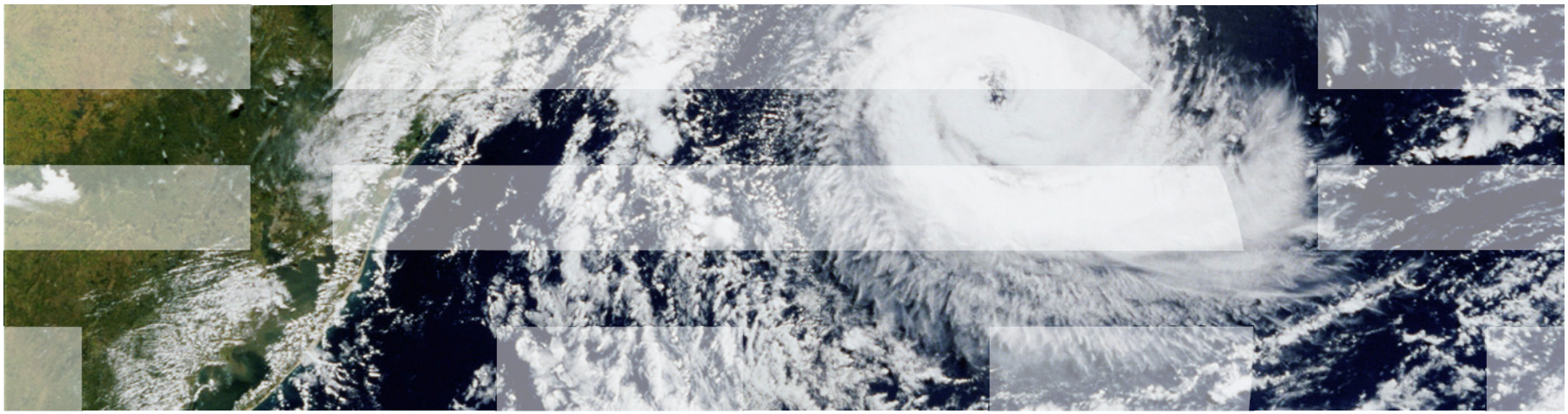
Exploiting z/VSE's OpenSSL support

Ingo Franzki



Trademarks

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

Not all common law marks used by IBM are listed on this page. Failure of a mark to appear does not mean that IBM does not use the mark nor does it mean that the product is not actively marketed or is not significant within its relevant market.

Those trademarks followed by ® are registered trademarks of IBM in the United States; all others are trademarks or common law marks of IBM in the United States.

For a complete list of IBM Trademarks, see www.ibm.com/legal/copytrade.shtml:

*, AS/400®, e business(logo)®, DBE, ESCO, eServer, FICON, IBM®, IBM (logo)®, iSeries®, MVS, OS/390®, pSeries®, RS/6000®, S/30, VM/ESA®, VSE/ESA, WebSphere®, xSeries®, z/OS®, zSeries®, z/VM®, System i, System i5, System p, System p5, System x, System z, System z9®, BladeCenter®

The following are trademarks or registered trademarks of other companies.

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.
Cell Broadband Engine is a trademark of Sony Computer Entertainment, Inc. in the United States, other countries, or both and is used under license therefrom.
Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.
Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.
Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.
UNIX is a registered trademark of The Open Group in the United States and other countries.
Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.
ITIL is a registered trademark, and a registered community trademark of the Office of Government Commerce, and is registered in the U.S. Patent and Trademark Office.
IT Infrastructure Library is a registered trademark of the Central Computer and Telecommunications Agency, which is now part of the Office of Government Commerce.

* 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.

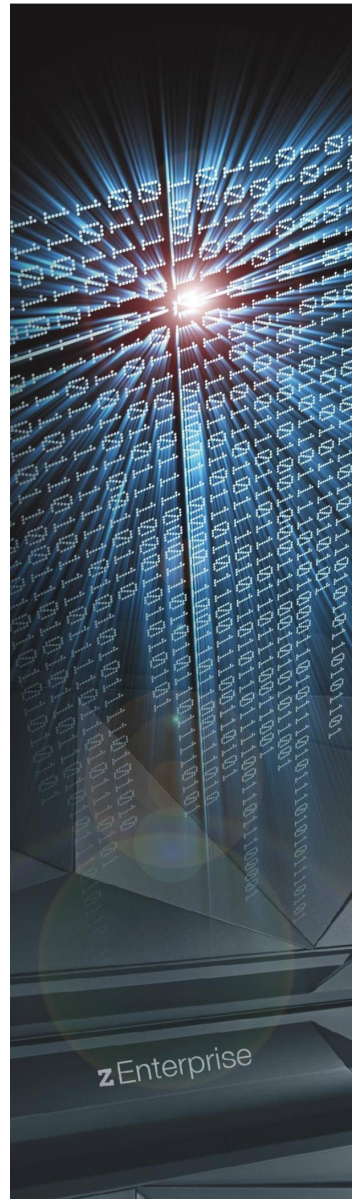
Prices subject to change without notice. Contact your IBM representative or Business Partner for the most current pricing in your geography.

Notice Regarding Specialty Engines (e.g., zIIPs, zAAPs and IFLs):

- Any information contained in this document regarding Specialty Engines ("SEs") and SE eligible workloads provides only general descriptions of the types and portions of workloads that are eligible for execution on Specialty Engines (e.g., zIIPs, zAAPs, and IFLs). IBM authorizes customers to use IBM SE only to execute the processing of Eligible Workloads of specific Programs expressly authorized by IBM as specified in the “Authorized Use Table for IBM Machines” provided at http://www.ibm.com/systems/support/machine_warranties/machine_code/aut.html (“AUT”).
- No other workload processing is authorized for execution on an SE.
- IBM offers SEs at a lower price than General Processors/Central Processors because customers are authorized to use SEs only to process certain types and/or amounts of workloads as specified by IBM in the AUT.

Agenda

- **Introduction**
- **Cryptography basics**
  - Encryption algorithms
  - Encryption keys
  - Diffie-Hellman versus RSA
  - Elliptic Curve Cryptography
- **Using cryptography with z/VSE**
  - Full tape encryption
  - Encryption Facility for z/VSE
  - SSL/TLS
  - Hardware cryptography support on IBM Z
  - OpenSSL
  - What's new with z/VSE V6.2

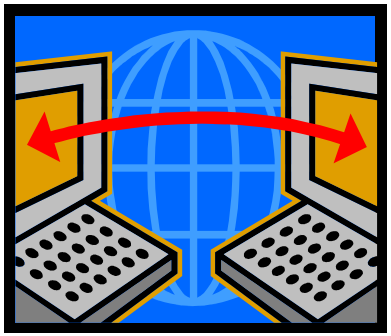


Why secure VSE ?

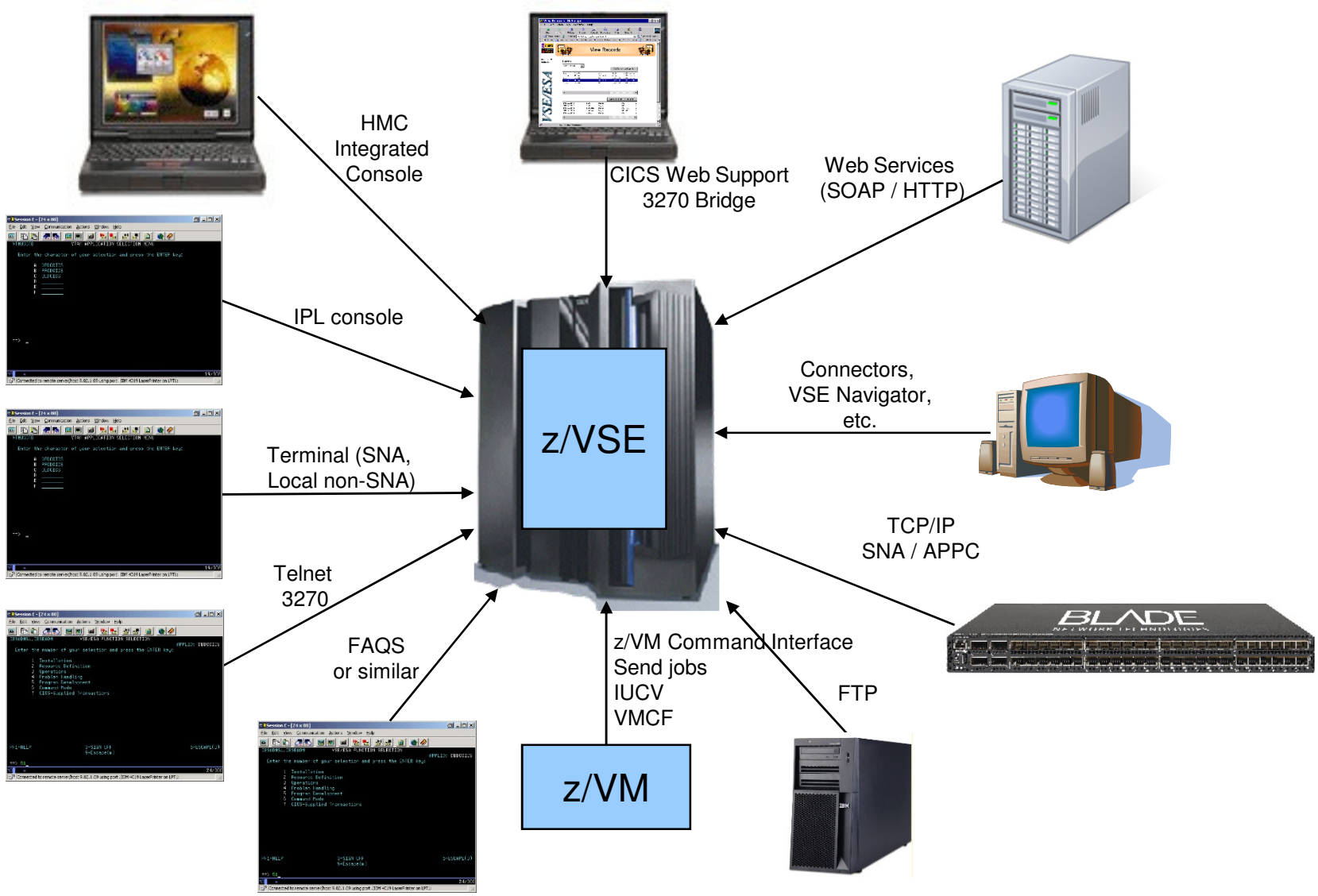
- **Prevent unauthorized access to VSE and data**
  - Keep secret data secret
  - Data modification by unauthorized users

- **Prevent users from damaging the VSE system (maybe by accident)**
  - Deletion of members or entries
  - Submission of jobs

- **Prevent unauthorized remote access to VSE**
  - Today most computers are part of a network
  - Theoretically every system in the network could connect to your VSE system
  - FTP allows to access production data
    - VSAM
    - POWER entries (listings)

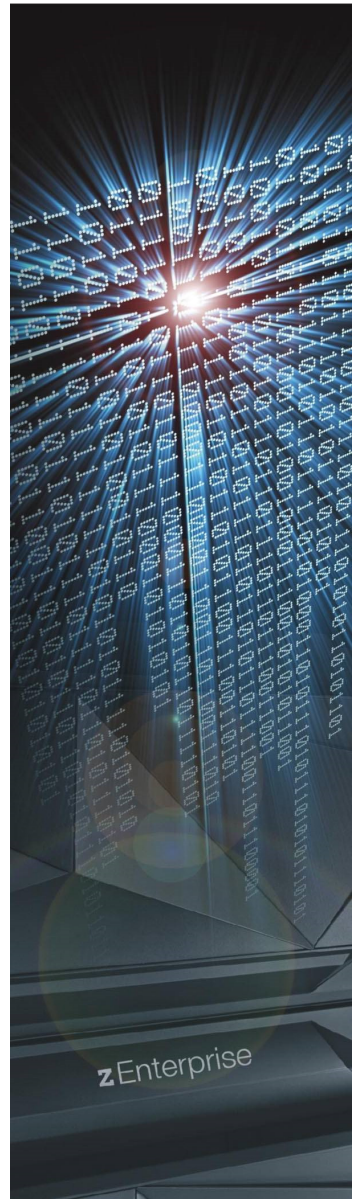


Ways into your z/VSE system – Are you securing them all?



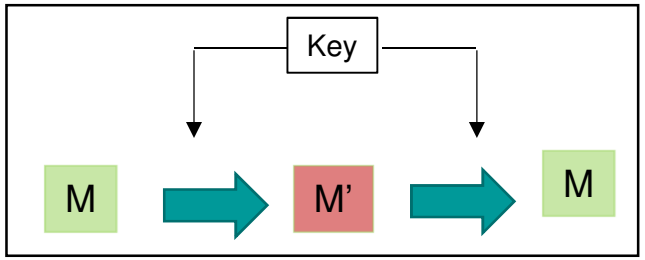
Agenda

- **Introduction**
- **Cryptography basics**
  - Encryption algorithms
  - Encryption keys
  - Diffie-Hellman versus RSA
  - Elliptic Curve Cryptography
- **Using cryptography with z/VSE**
  - Full tape encryption
  - Encryption Facility for z/VSE
  - SSL/TLS
  - SecureFTP
  - Hardware cryptography support on IBM Z
  - OpenSSL
  - What's new with z/VSE V6.2

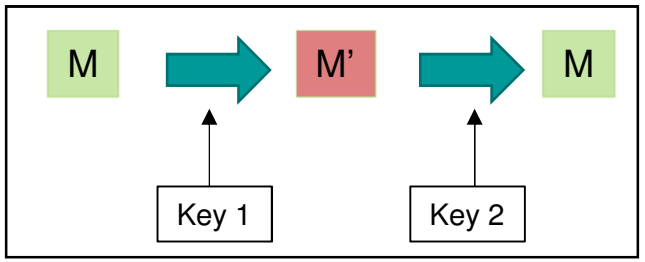


Encryption basics

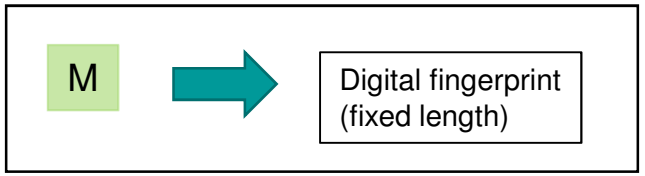
- **Symmetric encryption**
  - The same key is used to encrypt and decrypt
  - Example: RC4, DES, 3DES, AES



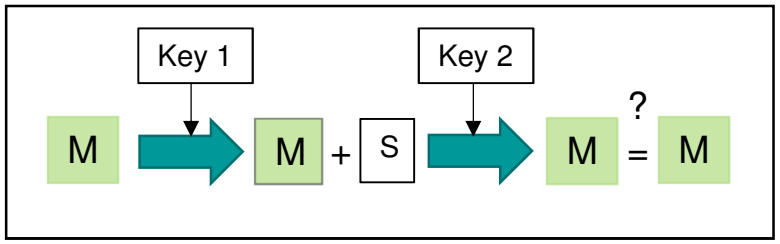
- **Asymmetric encryption**
  - One key is used for encryption, another key is used for decryption (public and private keys)
  - Example: RSA, Elliptic Curve Cryptography



- **Hash Algorithms**
  - A digital fingerprint of a text
  - Example: MD5, SHA



- **Signatures**
  - To create a digital signature asymmetric algorithms are used, mainly RSA



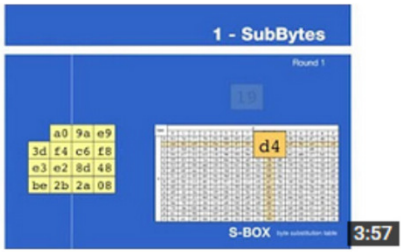
Different kinds of encryption keys

- **Keys that consist of numbers which are based on mathematical algorithms (asymmetric algorithms)**
  - RSA, Example: public key = (23,143), private key = (47,143)
    - Encryption of the number 7: 7^{23} mod(143) = 2
    - Decryption: 2^{47} mod(143) = 7

In this example, one could easily 'guess' the private key of 47 (i.e. brut force).

In reality this is done using much longer numbers, e.g. numbers of 4096 bits length

- **Keys that consist of random bit patterns (symmetric algorithms)**
  - The key consist of a bit pattern of fixed length, e.g.
    - 16 Bytes = 128 bit results in $2^{128} = 3{,}4*10^{38}$ possibilities
    - 32 Bytes = 256 bit results $2^{256} = 1{,}1*10^{77}$ possibilities
  - Example: Youtube: https://www.youtube.com/watch?v=evjFwDRTmV0



Animation of RIJNDEAL CIPHER : AES Encryption algorithm

HowTo
1 year ago • 2,497 views
This **animation** is made by Mr. Enrique Zabala. This is verison 4 made for CrypTool. This video is made for students so that they ...

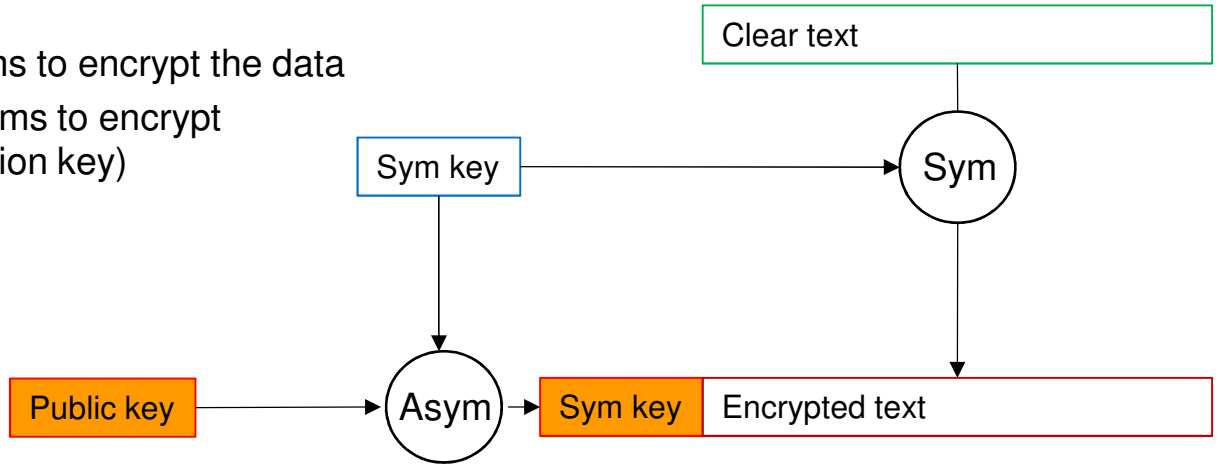
Encryption key sizes

… and its security level

RSA	ECDH	Symmetric	Hash	Security (bits)
		RC4		<?
		DES	MD5	<?
			SHA-1	<80
1024	160			80
2048	224	TDES	SHA-224	112
3072	256	AES-128	SHA-256	128
4096				
7680	384	AES-192	SHA-384	192
15360	512	AES-256	SHA-512	256

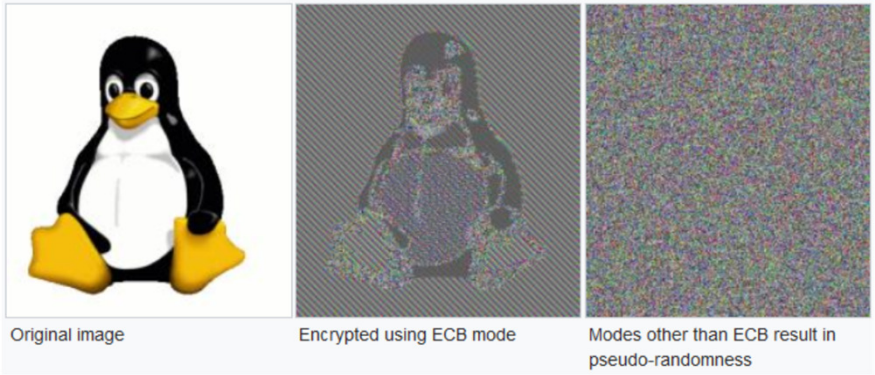
Why all this different encryption algorithms?

- **Asymmetric algorithms**
  - Are slower by factors than symmetric algorithms
  - Used to uniquely identify a communication partner
  - Can only encrypt a certain number of bytes
- **Symmetric algorithms**
  - Based on bit-shifting and logical computations (XOR, etc.)
  - Very fast
  - Can encrypt any numbers of bytes (usually in blocks of 8 or 16 bytes)
- **Idea:**
  - Use symmetric algorithms to encrypt the data
  - Use asymmetric algorithms to encrypt the symmetric key (session key)



Encryption modes (chaining)

- **ECB (Electronic Codebook)**
  - Each data block is encrypted separately



Source: Wikipedia

- **CBC (Cipher Block Chaining)**
  - The result of the encryption of one data block is fed into the encryption of the next data block

- **GCM (Galois Counter Mode)**
  - Encryption and generation of a hash (digital fingerprint) in one step
  - Most current and securest mode

- **Others**
  - CFB - Cipher Feedback
  - OFB - Output Feedback
  - XTS - XEX-based tweaked-codebook mode with ciphertext stealing
  - ...

SSL/TLS Connection establishment and key exchange

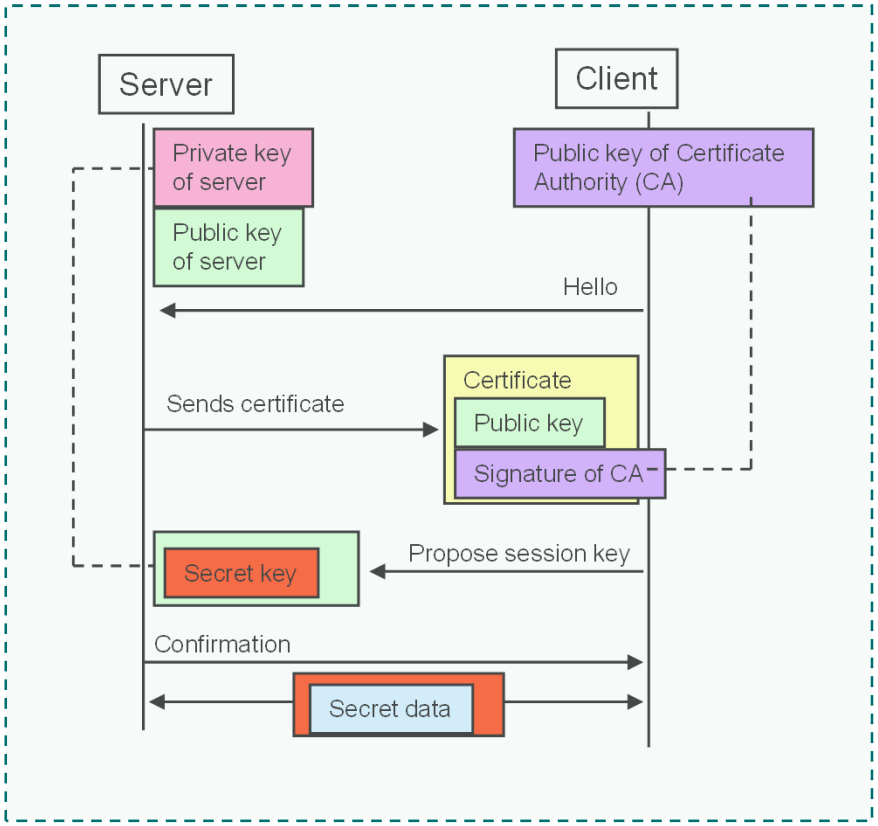
- **RSA-based:**
  - Commonly used
  - Long-term attacks are possible, because the session key is sent (encrypted) over the line

- **Diffie-Hellman based:**
  - Usage increases
  - Needs up to 30% more CPU
  - Long-term, attacks are NOT possible (forward secrecy), because the session key is not sent over the line
  - Usually used in combination with Elliptic Curve Cryptography (ECC)
  - https://www.youtube.com/watch?v=3QnD2c4Xovk

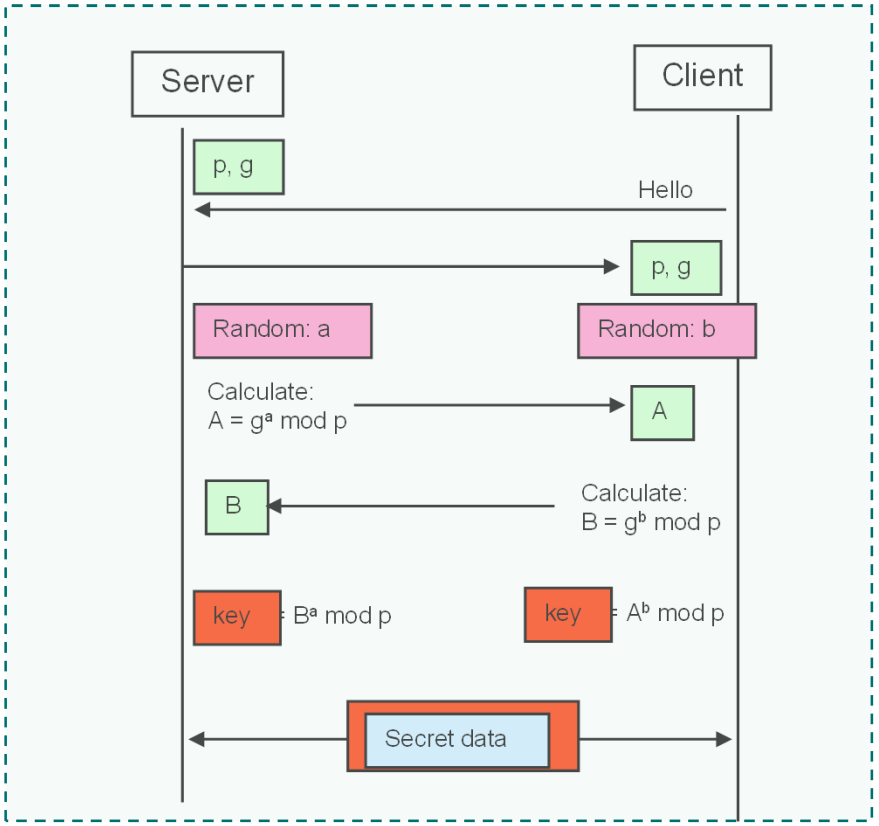
Public Key Cryptography: Diffie-Hellman Key Exchange (short version)
Art of the Problem
4 years ago • 366,437 views
Diffie-Hellman key exchange was one of the earliest practical implementations of key exchange within the field of cryptography.

Diffie-Hellman versus RSA

RSA



Diffie-Hellman



DH: session key does not go over the line

Diffie-Hellman versus RSA

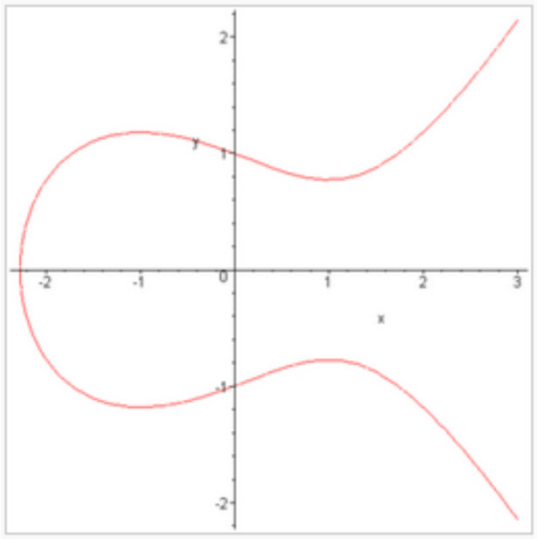
- **Diffie-Hellman**
  - Provides “forward secrecy”, because session key is not part of the session data
  - Needs up to 30% more CPU
  - Does not provide authentication, i.e. normally used together with RSA
  - Often used together with Elliptic Curve Cryptography (ECC) for better performance
  - Refer to Wikipedia / Youtube:
    - https://en.wikipedia.org/wiki/Diffie%E2%80%93Hellman_key_exchange
    - https://en.wikipedia.org/wiki/Elliptic_curve_Diffie%E2%80%93Hellman
    - https://www.youtube.com/watch?v=YEBfamv-_do

- **Special SSL/TLS cipher suites use Diffie-Hellman and Elliptic Curve**
  - DHE-RSA cipher suites: use Diffie-Hellman with RSA
  - ECDHE-RSA cipher suites: use Diffie-Hellman with ECC and RSA

DHE-RSA and ECDHE-RSA is supported on z/VSE with OpenSSL 1.0.1e or later

Some more info on Elliptic Curve Cryptography (ECC) …

- **Elliptic Curves**
  - Described through $y^2 = x^3 + ax + b$
  - Mathematical calculation based on points on the curve



- **Prime Curves (NIST)**
- **Brainpool curves**
  - Are being researched and provided by an working group of German governmental institutions and companies, including the German BSI (equivalent to U.S. NIST)
  - Are supported with OpenSSL 1.0.2 (and Java)
  - EC keys based on Brainpool curves are supported by Keyman/VSE
  - Refer to
    - http://www.ecc-brainpool.org/ (German website)
    - https://en.wikipedia.org/wiki/Elliptic_curve_cryptography#Implementation

- **CEX4C, CEX5C, CEX6C**
  - Provide ECC acceleration
  - z/VSE 6.2: Added Hardware acceleration for ECC in z/VSE and OpenSSL

Recommendations

- **Symmetric encryption:**
  - RC4 (Ron‘s Code 4), from the 80‘s, Stream cipher → Insecure
  - DES, 3DES (Data Encryption Standard), 1977, Block cipher → also treated as insecure nowadays
  - AES (Advanced Encryption Standard), 2000, Block cipher → Recommended (AES-128/256)

- **Asymmetric encryption:**
  - RSA (Rivest, Shamir, Adleman), 1977, → Use key sizes >= 2048 bits
  - ECC (Elliptic Curve Cryptography) (from the 80’s) → Use in combination with RSA

- **Hash Algorithms („digital fingerprint“)**
  - MD5 (Message Digest 5) → Insecure
  - SHA-1 (Secure Hash Algorithm, 2001) → no longer considered secure
  - SHA-2 (224, 256, 384, 512), 2002 → Recommended hash algorithm
  - SHA-3, standardized 2015 -> Successor of SHA-2, may not be available in applications

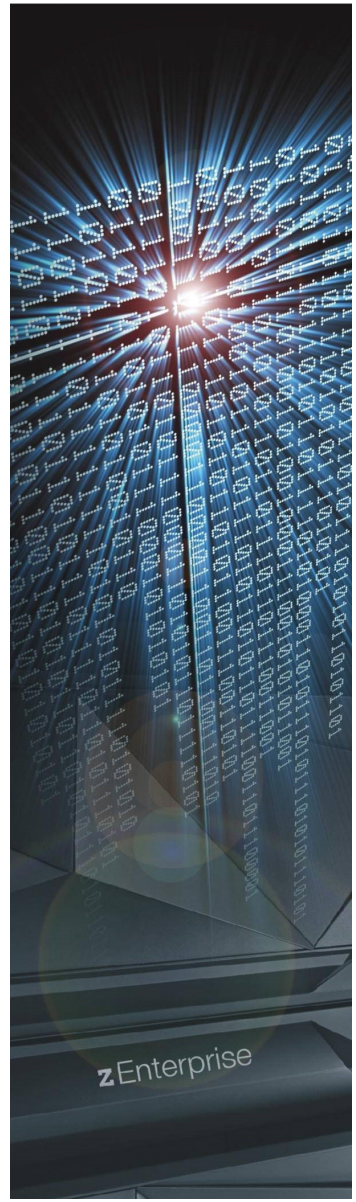
- **SSL/TLS protocol versions**
  - SSL 3.0 → Do not use this anymore
  - TLS 1.0 / 1.1 → May be used if TLS 1.2 is not available
  - TLS 1.2 → Recommended

What's coming next?

- **TLS 1.3**
  - First draft from 2016
  - Removes all deprecated and insecure algorithms
  - Key exchange only using Diffie-Hellmann, preferable with Elliptic-Curve
  - Data encryption with AES-GCM only
  - Already available in:
    - Google Chrome 56 (needs manual activation)
    - Firefox 52 (TLS 1.3 is activated per default)
    - OpenSSL TLS 1.3 support currently under development

Agenda

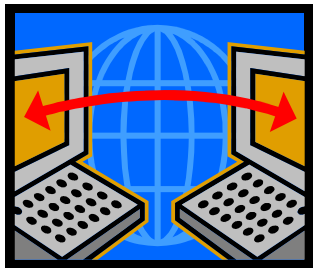
- **Introduction**
- **Cryptography basics**
  - Encryption algorithms
  - Encryption keys
  - Diffie-Hellman versus RSA
  - Elliptic Curve Cryptography
- **Using cryptography with z/VSE**
  - Full tape encryption
  - Encryption Facility for z/VSE
  - SSL/TLS
  - Hardware cryptography support on IBM Z
  - OpenSSL
  - What's new with z/VSE V6.2



Using cryptography with z/VSE

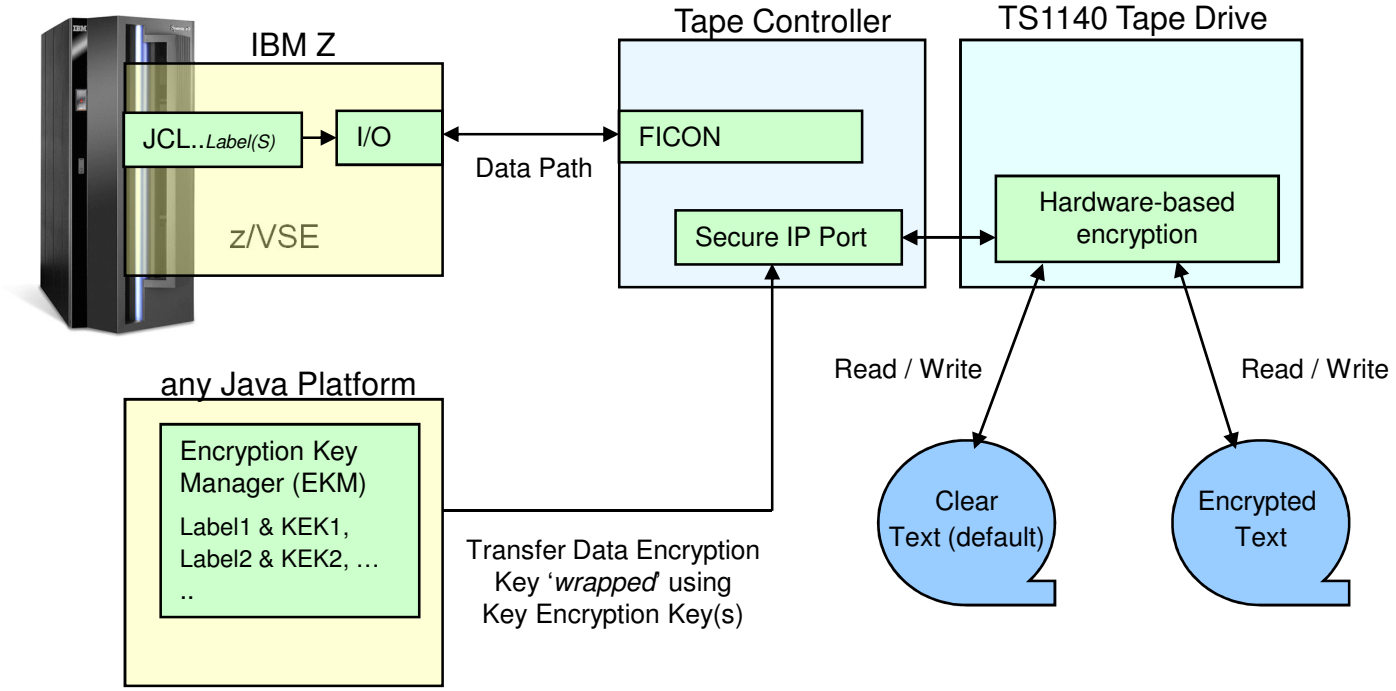
Main areas of cryptography:

- **Encryption of data transmitted over network connections (Data in flight)**
  - SSL/TLS, HTTPS
  - SecureFTP



- **Encryption of data stored on disk or tape (Data at rest)**
  - Encryption of backups or archives
  - Exchange of encrypted and/or signed data with customers or business partners
  - TS1140 Encrypting Tape Drive
  - Encryption Facility for z/VSE

IBM Tape Encryption – TS1140



encryption mode (03=write)

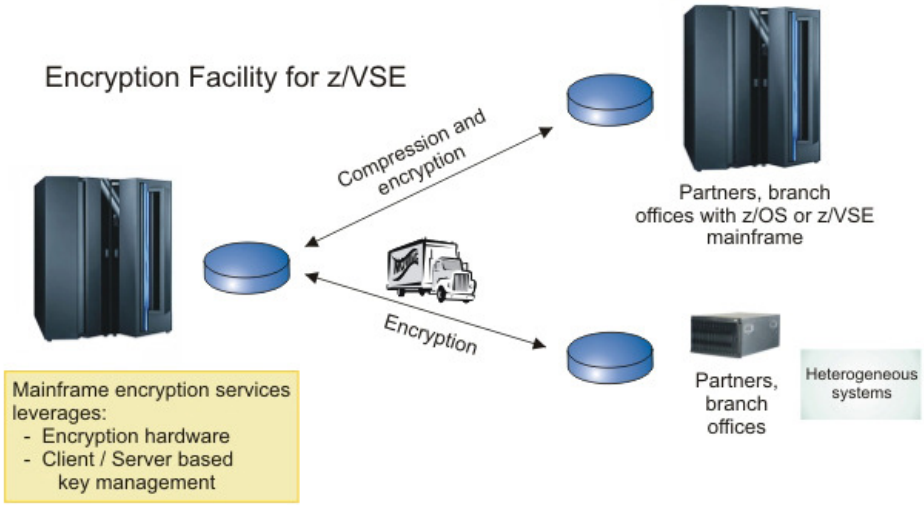
```
// JOB ENCRYPT
// ASSGN SYS005,480,03
// KEKL UNIT=480,KEKL1='MYKEKL1',KEM1=L,KEKL2='MYKEKL2',KEM2=L
// EXEC LIBR
  BACKUP LIB=PRD2 TAPE=SYS005
/*
/&
```

encoding mechanism (L=Label, H=Hash)

key label1 (name of the 1. KEK-key in EKM)

Encryption Facility for z/VSE

- Secure business and customer data
- Address regulatory requirements
- Protect data from loss and inadvertent or deliberate compromise
- Enable sharing of sensitive information across platforms with partners, vendors, and customers
- Enable decrypting and encrypting of data to be exchanged between z/VSE and non-z/VSE platforms



- The Encryption Facility for z/VSE is packaged as an optional, priced feature of VSE Central Functions V8.1 (5686-CF8-40).

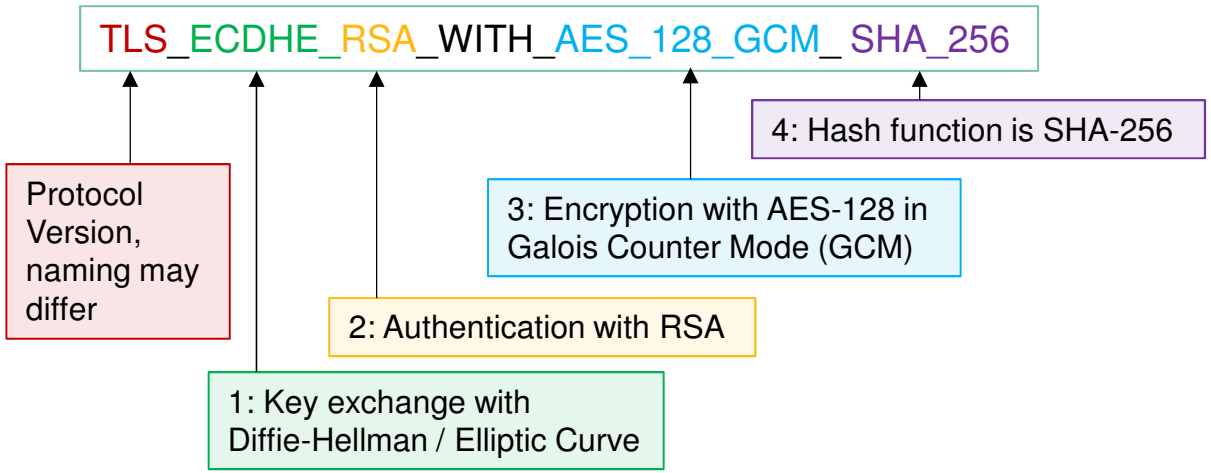
- The Encryption Facility for z/VSE V1.1 uses IBM Z data format

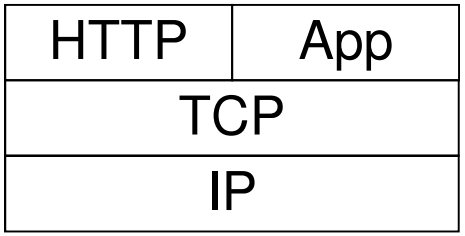
- The Encryption Facility for z/VSE V1.2 uses the standard OpenPGP data format
  - PGP stands for „Pretty Good Privacy“, invented by Phil Zimmermann in 1991
  - Open Standard, described in RFCs 2440 and 4880
  - Compatible with Encryption Facility for z/OS V1.2 and many other OpenPGP implementations

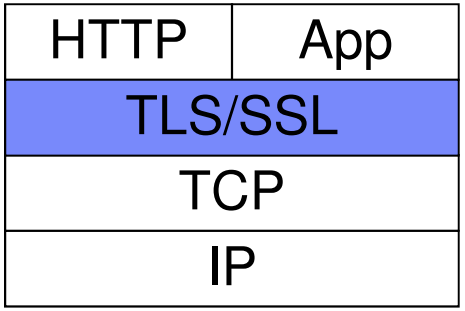
Transport Layer Security – Encrypted data transfer over a network

Formerly named SSL – Secure Socket Layer

- **TLS/SSL provides a communication channel with message integrity, authentication, and confidentiality**
- **TLS/SSL is a widely used protocol**
  - Secure HTTP (HTTPS) is used very often in the Internet
- **TLS/SSL uses a TCP connection to transfer encrypted messages**
  - Uses asymmetric cryptography for session initiating
  - Uses symmetric cryptography for data encryption
- **Cipher suites defines the algorithms used:**
  - For key exchange
  - For encryption
  - For hash algorithm







All together builds an SSL/TLS cipher suite

Example: maps.google.de

TLS_ECDHE_RSA_WITH_AES_128_GCM_ SHA_256

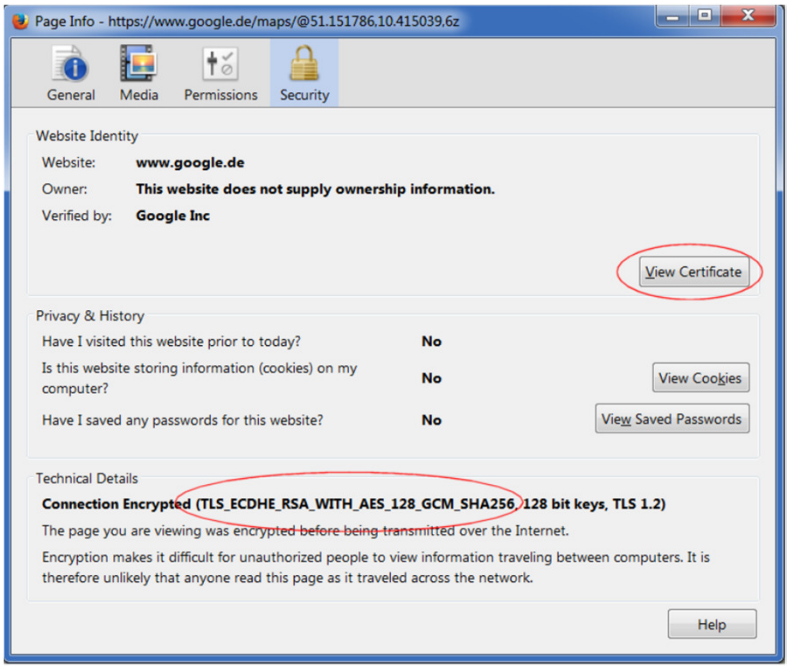
4: Hash function is SHA-256

3: Encryption with AES-128 in Galois Counter Mode (GCM)

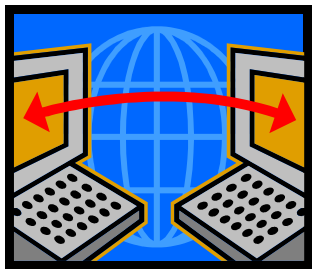
2: Authentication with RSA

1: Key exchange with Diffie-Hellman / Elliptic Curve

Protocol Version, naming may differ



SecureFTP



- **The FTP protocol provides a easy and straight forward protocol for transferring files between systems on different platforms**
  - Many installations rely on it to efficiently transmit critical files that can contain vital information such as customer names, credit card account numbers, social security numbers, corporate secrets and other sensitive information
  - FTP protocol transmits data without any authentication, privacy or integrity

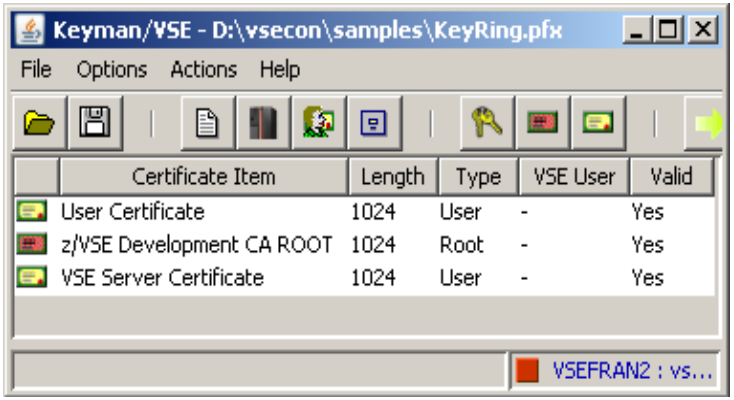
- **SecureFTP provides user authentication, privacy and integrity by using RSA digitally signed certificates, data encryption and secure hash functions**
  - SecureFTP is integrated into TCP/IP for VSE with z/VSE V4.1 or later (at no additional charge) or offered as separately priced product by CSI

- **How to setup Secure FTP with VSE:**
  ftp://ftp.software.ibm.com/eserver/zseries/zos/vse/pdf3/How_to_setup_SecureFTP_with_VSE.pdf

Key & Certificate Management

Cryptography uses Keys and Certificates

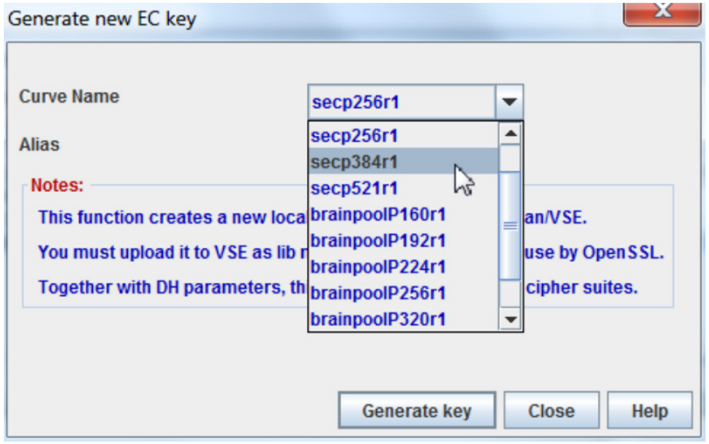
- **Key Management is not trivial**
  - Key must often be kept secure for a very long time
  - You must be able to associate the encrypted data with the corresponding key(s)
  - Encrypted data and the corresponding key(s) must be strictly separated

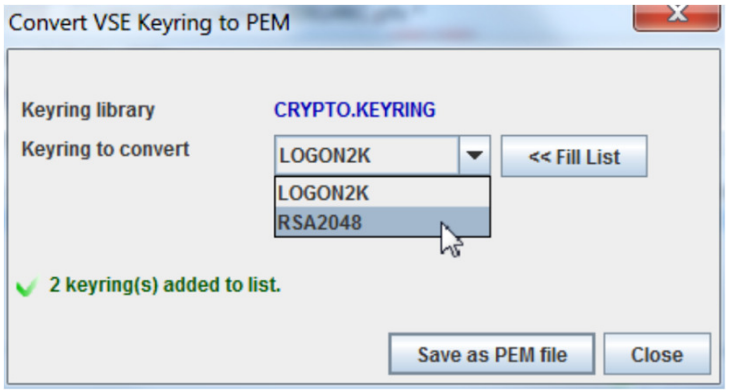


- **Keyman/VSE**
  - Creation of RSA keys and digital certificates
  - Upload of keys and certificates to VSE
  - Creation of PKCS#12 keyring files (use with Java-based connector or import into a Web browser)
  - Download from VSE Homepage https://www.ibm.com/it-infrastructure/z/zvse-downloads

Keyman/VSE updates

- **Elliptic Curve Crypto support**
  - Create and upload Elliptic-Curve (EC) key pairs.
- **SHA-256 support**
  - Support SHA-256 signatures in certificates.
  - This may require additional 1.5F zaps on TCP/IP for VSE.
- **Convert CSI keyrings to OpenSSL PEM**
  - Use existing PRVK, CERT, and ROOT members on VSE and build an equivalent PEM file for OpenSSL

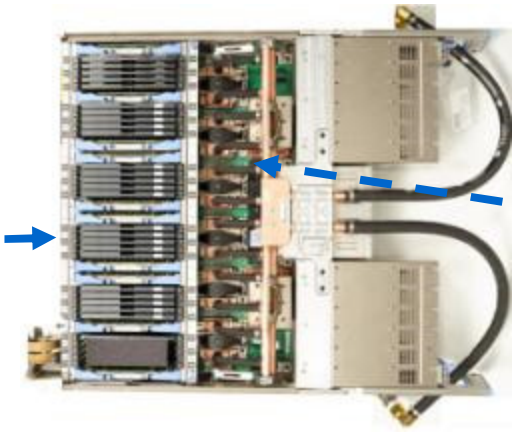




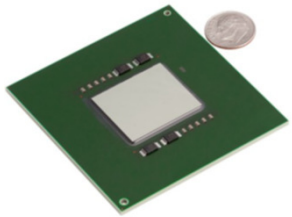
Hardware Crypto Support on IBM Z

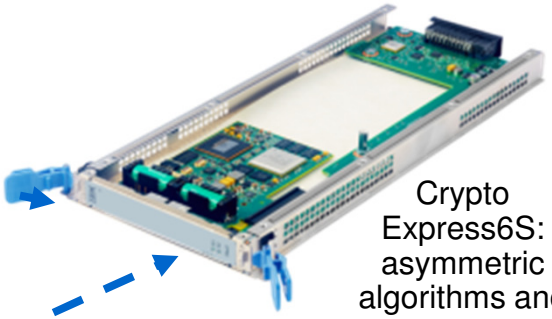
PCIe I/O drawers

CPC Drawer



CPACF Chip: symmetric algorithms and hashing





Crypto Express6S: asymmetric algorithms and secure key

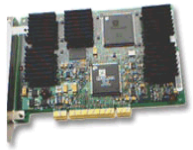
TKE for management of the card: secure key entry and EP11

Hardware Crypto Support on IBM Z

z800, z900

PCICC(*)

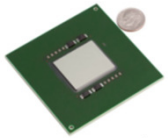
PCICA



z890, z990

PCIXCC

CPACF



z9, z10

CEX2C CEX2A
configurable

PRNG,
protected-key CPACF

z196, z114

CEX3C CEX3A
configurable

CFB-mode

zEC12,
zBC12

CEX4C

CEX4A

CEX4P

z13, z13s

CEX5C

CEX5A

CEX5P

z14, z14s

CEX6C

CEX6A

CEX6P

CCA Coprocessors
- RSA acceleration
- RSA key generation
- Random numbers

Accelerators
- RSA acceleration

EP11 Coprocessor
- PKCS#11
- Not exploited by VSE

On-board
crypto chip
- Symmetric encryption
- Hashing (SHA)

(*) PCICC was never supported by VSE

Crypto Express6S

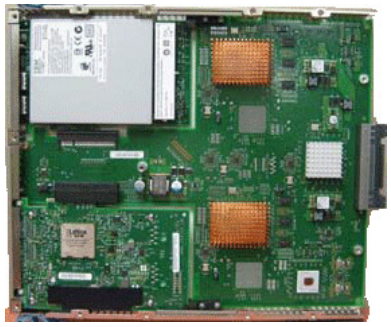
- Exclusive to IBM z14
- One-port card, i.e. one AP (adjunct processor) per physical card
  - 2 cards min, 16 cards max per machine
- Seneca I/O cage (the 'S' in the name)
- Can be configured in one of three ways:
  - CEX6A: Accelerator
  - CEX6C: IBM Common Cryptographic Architecture (CCA) coprocessor
  - CEX6P: IBM Enterprise Public Key Cryptography Standards (PKCS) #11 (EP11) coprocessor
- Form factor comparison CEX3 / CEX6S:

Support for Crypto Express6S

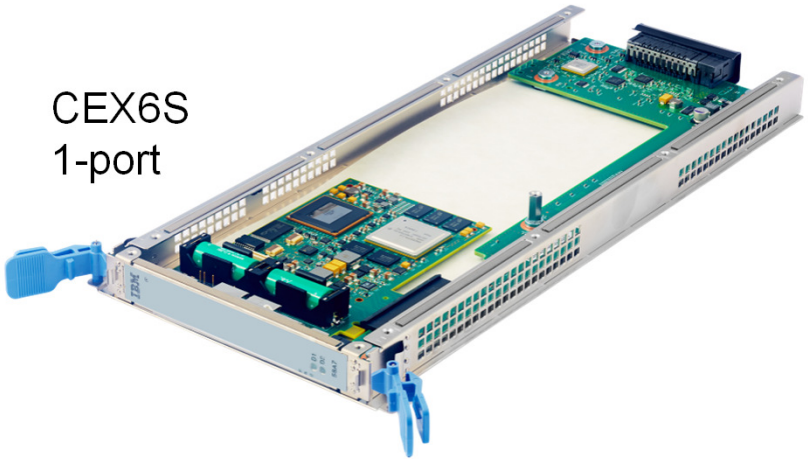
- Included in z/VSE 6.2 GA version
- APAR DY47715 for z/VSE 5.2
- APAR DY47716 for z/VSE 6.1

NEW!

CEX3
2-port



CEX6S
1-port



z/VSE Hardware Configuration

- **z/VSE hardware configuration not necessary for crypto hardware**
  - No IOCDS definition in VSE
  - No device type
  - No ADD statement
  - You may have to define the devices in the HMC (LPAR) or z/VM directory
- **Use of crypto hardware is transparent to end users and applications**
  - But use of crypto hardware can be disabled via option
- **How to setup cryptographic hardware for VSE:**
  - https://www.ibm.com/support/knowledgecenter/SSB27H_6.2.0/technical_articles_and_whitepapers.html

```
FB 0095 1J054I FOUND A CRYPTO EXPRESS5S CARD AT AP 0
FB 0095 1J054I FOUND A CRYPTO EXPRESS5S CARD AT AP 3
FB 0095 1J005I HARDWARE CRYPTO DEVICE DRIVER INITIALIZED SUCCESSFULLY.
FB 0095 1J006I USING AP QUEUE 79
```

OpenSSL Support

- **What is OpenSSL?**
  - OpenSSL is an Open Source project providing an SSL/TLS implementation and key management utilities
  - Available for most Unix-style operating systems, MAC, Windows, and IBM System i (OS/400)
  - For details on OpenSSL refer to http://www.openssl.org/

- **What is available on z/VSE?**
  - OpenSSL 1.0.2h runtime library
  - New component: z/VSE cryptographic services, 5686-VS6-17-562
  - Software implementations for all algorithms with all key lengths
  - Hardware Crypto Support (Crypto Express cards and CPACF)
  - Programming APIs:
    - OS390 / z/OS compatible SSL API (gsk_initialize(), gsk_secure_soc_init(), etc.)
    - Subset of the OpenSSL API (LE/C)

OpenSSL Exploitation

- **CICS TS for z/VSE V2.2: OpenSSL for CICS Web Support**
  - Enable OpenSSL via // SETPARM BPX$GSK="IJBGSKOS"

- **z/VSE Connectors**
  - **z/VSE Connector Server** (LE/C)
  - **z/VSE Script Server & Client** (LE/C)
  - **z/VSE Database Connector/DBCLI** (EZA)
  - **z/VSE HTTP Client** (LE/C)
  - **z/VSE SOAP Engine**
    - As server: OpenSSL for CICS Web Support
    - As client: z/VSE HTTP Client (LE/C)
  - **z/VSE REST Engine**
    - As server: OpenSSL for CICS Web Support
    - As client: z/VSE HTTP Client (LE/C)
  - **z/VSE LDAP Client and Signon support** (LE/C)
  - **z/VSE VTAPE Tape Data Handler** (LE/C)

- **IPv6/VSE product**
  - **SSL Proxy Server** (BSTTPRXY)
  - **Automatic TLS Facility** (BSTTATLS)

- **Applications using the LE/C socket interface:** enable via LE/C socket API multiplexer
- **Applications using the EZA socket interface:** enable via EZA socket API multiplexer

News with z/VSE 6.2

- **OpenSSL component of z/VSE enhancements:**
  - The OpenSSL component of z/VSE (z/VSE Cryptographic Services) was upgraded to benefit from newer SSL/TLS functions
  - The OpenSSL component transparently uses hardware acceleration for Elliptic Curve Cryptography (ECC), if available

- **CICS TS V2.2 security enhancements:**
  - OpenSSL support for CICS Web Support will give clients more flexibility and allow them to take advantage of the OpenSSL security.
  - Enable OpenSSL via // SETPARM BPX$GSK="IJBGSKOS"

- **EZA API enhancements:**
  - The EZA 'Multiplexer' and the EZA OpenSSL support will simplify the use of the EZA interface with any TCP/IP stack and allow to transparently use OpenSSL with EZA SSL-applications

- **VTAPE enhancements:**
  - Clients can use SSL/TLS connections for remote VTAPEs (virtual tapes) to protect sensitive data during network transfer

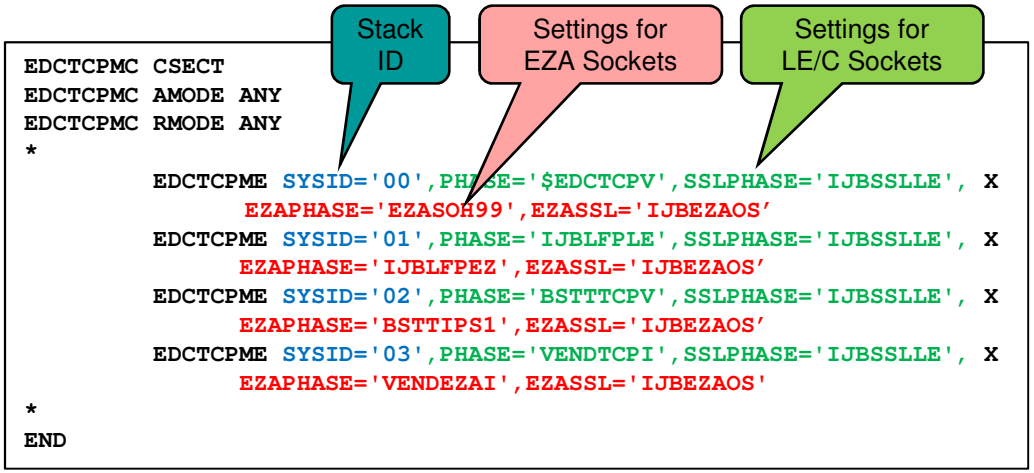
z/VSE 6.2: EZA Multiplexer

- **With the EZASOKET and EZASMI interfaces you can specify which socket interface module to the TCP/IP partition is to be used**
  - Default: EZASOH99 (for TCP/IP for z/VSE)

- **Select the EZA socket interface routine:**
  - Via JCL statement: **`// SETPARM [SYSTEM,]EZA$PHA='phasename'`**
  - Via parameter ADSNAME on the EZAAPI/EZASOKET INITAPI call

- **The EZA Multiplexer can be used to ease the correct setup of socket interface modules for the corresponding stack IDs**
  - The multiplexer allows you to perform a **one time setup** and to assign the corresponding socket interface modules to the stack IDs
  - The use of the multiplexer is transparent for your application

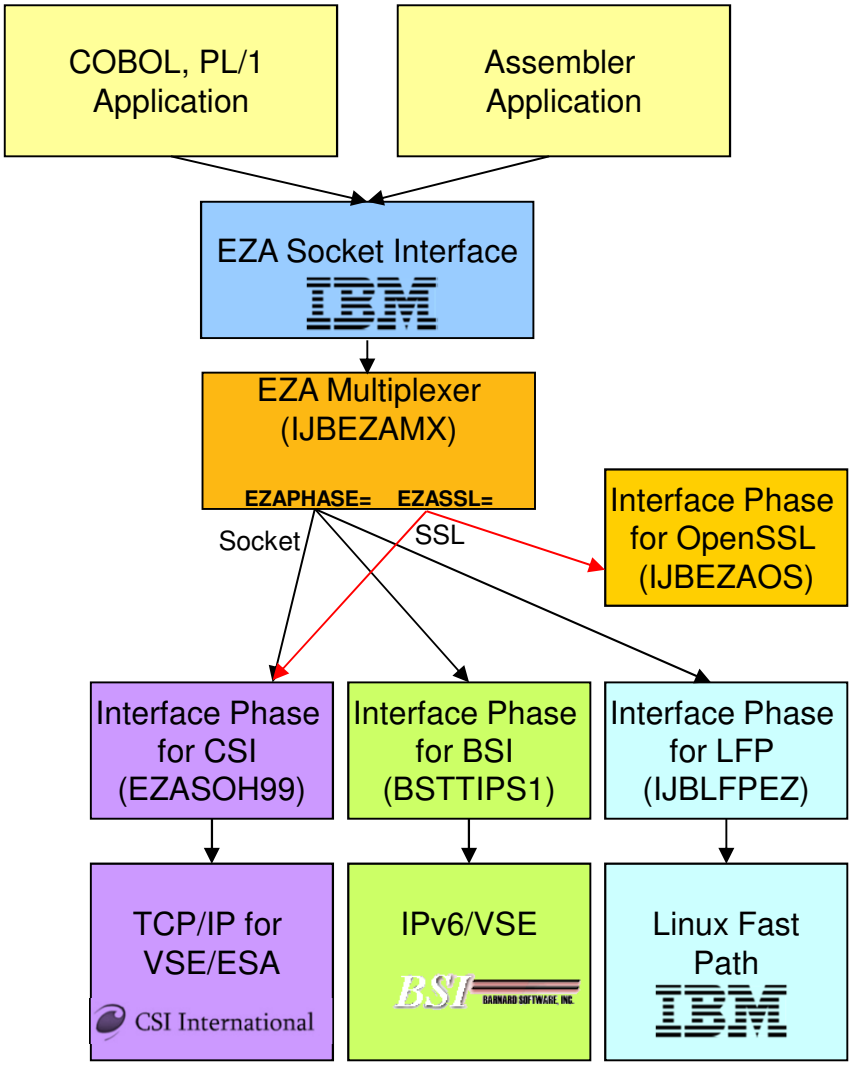
- **Setup:**
  - Select **IJBEZAMX** as EZA interface phase for all applications (via SETPARM SYSTEM)
  - Configure Multiplexer via **EDCTCPMC** in ICCF library 62
    - Same configuration as for LE/C Socket API Multiplexer
    - Additional parameters now allow to specify EZA interfaces



z/VSE 6.2: EZA OpenSSL Support

- **Besides the EZA socket interface routine, the EZA Multiplexer also allows you to specify an alternative EZA SSL interface routine**
  - Default: The same as the EZA socket interface routine
- **The new EZA SSL interface routine IJBEZAOS provides an interface to z/VSE's OpenSSL implementation**
- **The use of an alternative EZA SSL interface routine is transparent for your application**
  - OpenSSL uses different key and certificate formats (e.g .PEM instead of .PRVK, .ROOT, .CERT)

→ This makes z/VSE's OpenSSL support available for non-LE/C applications (i.e. COBOL, PL/1, HLASM)



Questions ?