E56

z/VSE Security Exploitation with Crypto Hardware

Ingo Franzki – ifranzki@de.ibm.com

IBM System z Expo
September 17-21, 2007
San Antonio, TX

Trademarks

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

CICS*	IBM*	Virtual Image Facility
DB2*	IBM logo*	VM/ESA*
DB2 Connect	IMS	VSE/ESA
DB2 Universal Database	Intelligent Miner	VisualAge*
e-business logo*	Multiprise*	VTAM*
Enterprise Storage Server	MQSeries*	WebSphere*
HiperSockets	OS/390*	xSeries
	S/390*	z/Architecture
	SNAP/SHOT *	z/VM
		z/VSE
		zSeries

* Registered trademarks of IBM Corporation

The following are trademarks or registered trademarks of other companies.

LINUX is a registered trademark of Linus Torvalds

Tivoli is a trademark of Tivoli Systems Inc.

Java and all Java-related trademarks and logos are trademarks of Sun Microsystems, Inc., in the United States and other countries

UNIX is a registered trademark of The Open Group in the United States and other countries.

Microsoft, Windows and Windows NT are registered trademarks of Microsoft Corporation.

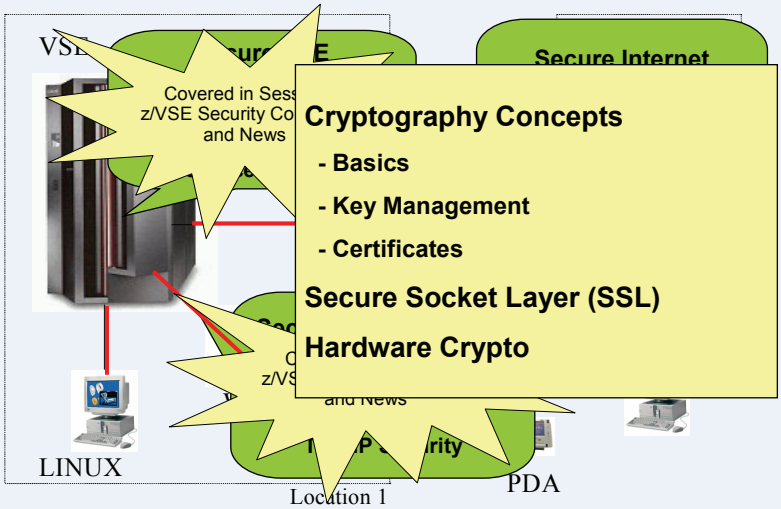
SET and Secure Electronic Transaction are trademarks owned by SET Secure Electronic Transaction LLC.

Intel is a registered trademark of Intel Corporation.

Security requirements

- Security requirements are increasing in today's world
  - Data security
  - Data integrity
  - Keep long-term data audit-save
- The number of attacks increase daily
  - Industrial spying
  - Security exploits, Denial-of-Service attacks
  - Spam, Phishing, …
- Not paying attention to security requirements can be very expensive
  - Your data is the heart of your company
  - Loosing your customer data is a disaster
  - You can loose customers
- IT Security gets more and more important
  - You need to consider the whole IT Environment not only single systems

Security in a heterogeneous environment



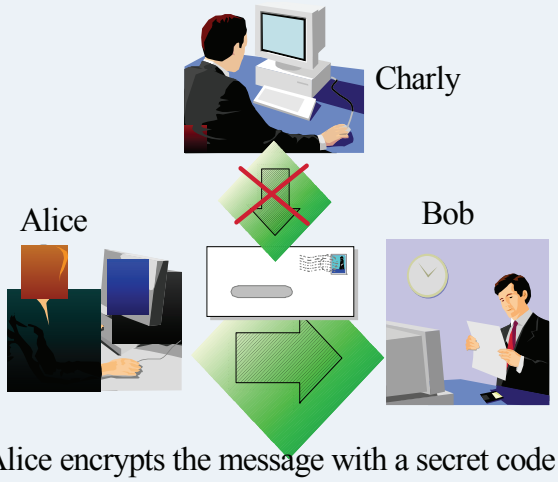
What can cryptograph do for you?

- 2 main areas
  - Encryption of data transmitted over TCP/IP connections
    - SSL, HTTPS
    - SecureFTP
  - Encryption of data stored on disk or tape
    - Encryption of backups or archives
    - Signing of data
    - Exchange of encrypted and/or signed data with customers or business partners

Why Cryptography ?

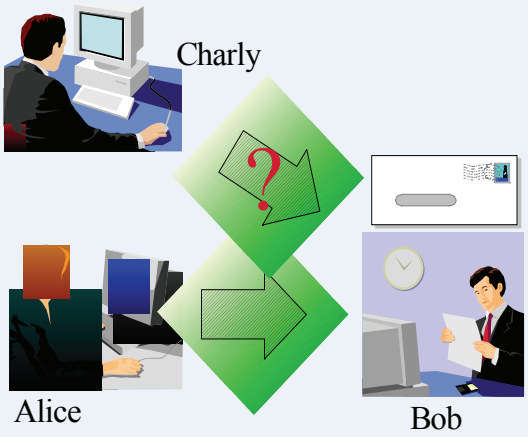
- Keeping secrets
  - Alice wants to send Bob confidential information,
  - Charly should not be able to read it.
- Proving identity
  - Bob receives a message from Alice. How he can be sure that it is really from Alice?
- Verifying information
  - Bob receives a message from Alice. How he can be sure that the content has not been modified?

Keeping Secrets



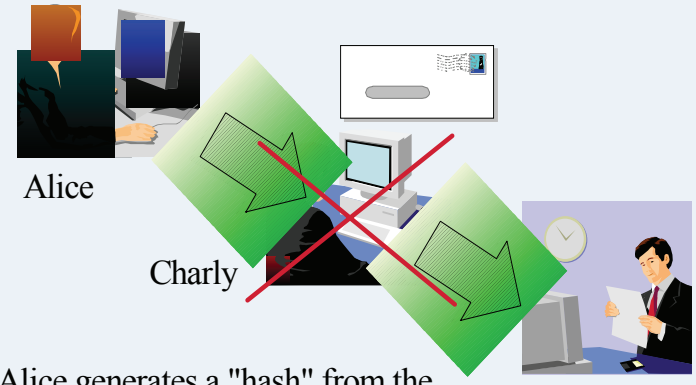
Alice encrypts the message with a secret code that only she and Bob knows

Proving Identity



Alice "signs" the message by attaching a secret phrase that only she and Bob knows

Verifying Information

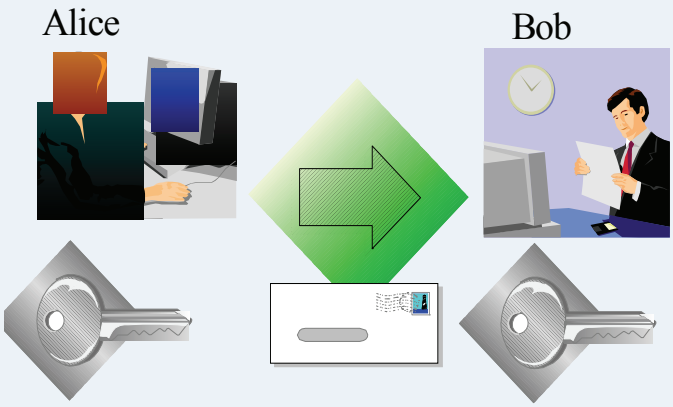


Alice generates a "hash" from the message using a secret code and attaches it to the message. Bob also generates the hash from the received message and compares it.

Secret Key Cryptography (symmetric)

- **Both parties know the same secret code (key)**
- **The key must be kept secret**
- **Encryption algorithm = mathematical transformation of the data with the key**
  - **DES** Data Encryption standard
  - **3DES** Triple strength DES
  - **AES** Advanced Encryption Standard
- **Typical key length: 40, 56, 128 or 256 bit**

Secret Key Cryptography - continued



Alice encrypts the message with the secret key and sends it to Bob. Bob decrypts the message with the secret key.

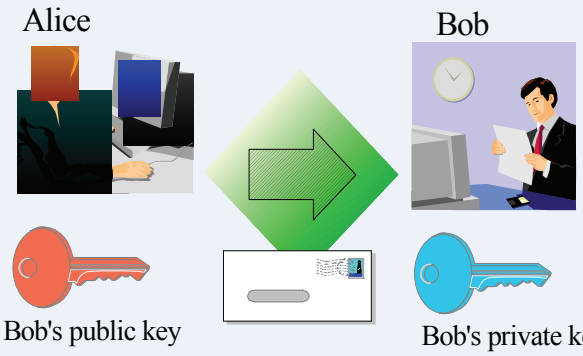
Public Key Cryptography (asymmetric)

- One "public key" and one "private key“
- "Private key" is kept secret (private)
- "Public key" is published

- Asymmetric cryptography is based on mathematical problems, that are much easier to create than to solve
  - RSA Rivest Shamir Adleman
  - DSA Digital Signature Algorithm
  - DHE Diffie Hellman Algorithm

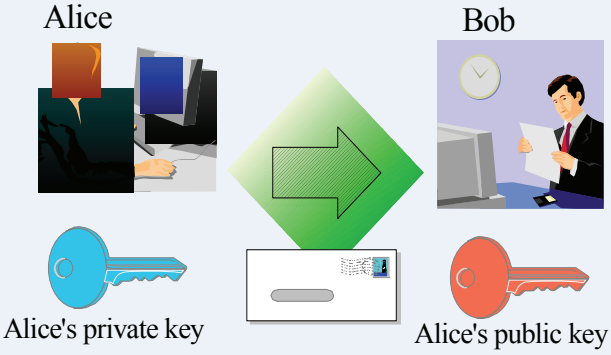
- Typical key length: 512, 1024 or 2048 bit

Public Key Cryptography - Encrypting



Alice encrypts the message using Bobs public key and sends it to Bob. Bob decrypts it using his private key.
Since only Bob knows his private key, only he can read the message.

Public Key Cryptography - Signing

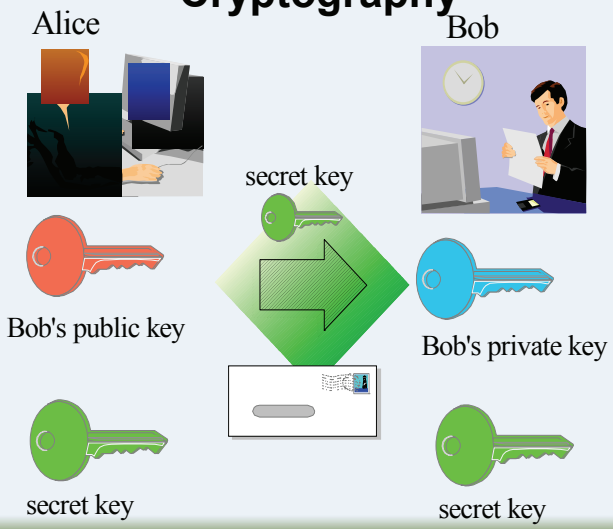


Alice encrypts the message using her private key and sends it to Bob. Bob decrypts it using Alice's public key.
The message is "signed" by Alice since it can only be decrypted using **her** public key.

Combined Symmetric and Asymmetric Cryptography

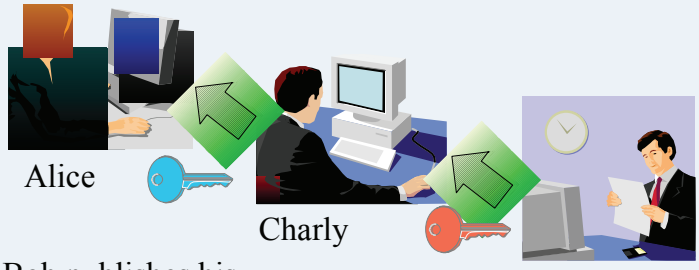
- Asymmetric cryptography is very CPU-time consuming
- Use asymmetric cryptography only for secret key exchange
- Data encryption uses symmetric cryptography
- Secret key is generated by random
- SSL also uses this mechanism

Combined Symmetric and Asymmetric Cryptography



Key Management

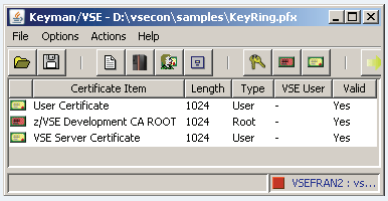
- Key exchange is not trivial:
  - Is the public key really from the right person?



Bob publishes his public key, but Charly intercepts this and instead sends his public key to Alice.

Key Management

- **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 http://www.ibm.com/servers/eserver/zseries/zvse/downloads/#vkeyman**



Certificates

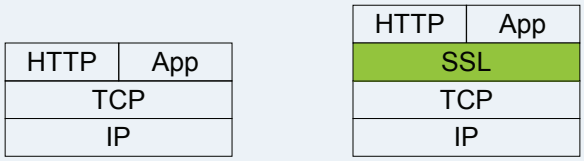
- A certificate contains the following items
  - The subject (name of the person)
  - The subject's public key
  - Period of validity
  - The issuer
  - Issuers signature
- The issuer "signs" the certificate by encrypting a hash of the certificate content with his private key
- Everyone can check the sign by decrypting it with the issuers public key

Certificate Authorities

- A certificate is issued by a certificate authority (CA)
- If a user trusts the certificate authority, he can trust the certificates issued by this CA
- CAs identify itself with a "self signed certificate":
  - The public key in the certificate is also the public key used to decrypt the signature
  - Subject and issuer are the same
- It is possible to build certificate hierarchies
- Certificate revocation lists are used to mark certificates that have been issued by error

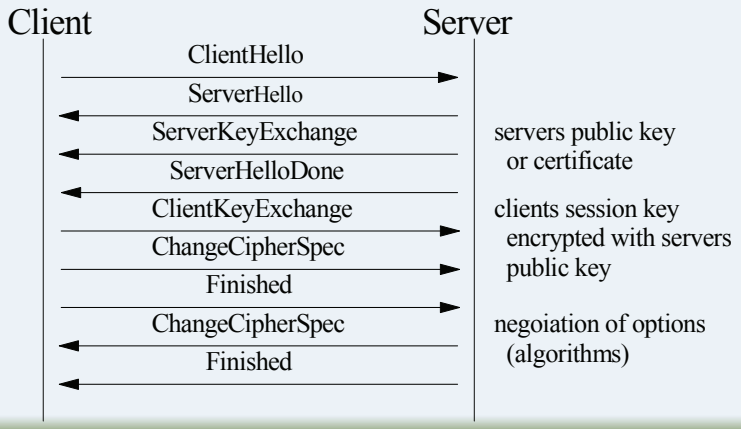
SSL (Secure Socket Layer)

- **SSL provides a communication channel with message integrity, authentication, and confidentiality**
- **SSL is a widely used protocol**
  - **Secure HTTP (HTTPS) is used very often in the Internet**
- **SSL uses a TCP connection to transfer encrypted messages**
  - **Uses asymmetric cryptography for session initiating**
  - **Uses symmetric cryptography for data encryption**
- **As the name implies, SSL is a layer on top of TCP**



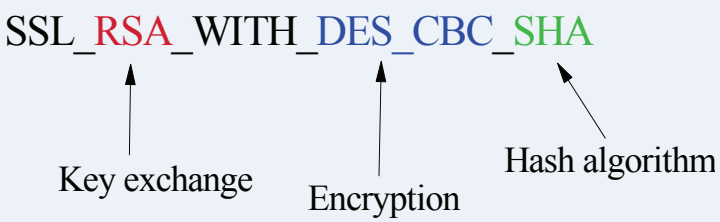
SSL Protocol

- **The SSL protocol defines a set of messages**



Cipher Suites

- Cipher suites defines the algorithms used:
  - For key exchange
  - For encryption
  - For hash algorithm



Session Caching

- "SSL Session" means
  - Secret key used for data encryption
  - Negotiated algorithms
- Establishing a SSL Session is a complex and time consuming mechanism
- Session caching allows to reuse previously negotiated SSL parameters
- No need of repeating the negotiations or authentications
  - The same symmetric key is used
- The connection becomes more unsecured
- A SSL Session time-out defines how long a session is kept alive

SSL for VSE

- SSL for VSE is part of the TCP/IP for VSE base
  - Enabled with the Application Pak
  - Integrated into TCP/IP for VSE
- Supports SSL 3.0 and TLS 1.0
  - Key exchange: RSA
  - Data Encryption: DES and Triple DES, AES
  - Hash algorithm: MD5, SHA
  - Supports X.509v3 PKI Certificates
- SSL daemon implementation for HTTPS, Telnet
- SSL API compatible with the OS/390 SSL API
- Uses Hardware Crypto acceleration if available

SSL Daemon (SSLD)

- Define a SSL daemon for each TCP port that you want to secure:

```
DEFINE TLSD,ID=MYSSLD,
            PORT=443,
            HTTPS port
            PASSPORT=443,
            CIPHER=0A096208,       Cipher suites
            CERTLIB=CRYPTO,        library name
            CERTSUB=KEYRING,       sub library name
            CERTMEM=MYKEY,         member name
            TYPE=1,                server application
            MINVERS=0300,          SSL 3.0
            DRIVER=SSLD            Driver phase name
```

Secure Socket Layer API

- Compatible to OS/390 SSL API
- Functions available for
  - Session initiating
  - Sending/receiving data
  - Ending a session
- SSL API is based on Socket API
- SSL API can be called from
  - LE-C programs
  - Assembler programs

Secure Socket Layer - Concepts

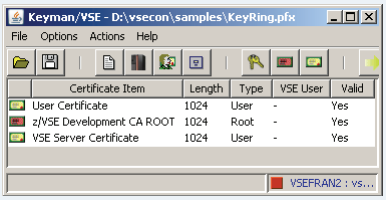
- When using SSL, you need to have a set of certificates and keys
  - A Public/Private key pair
  - Root Certificate
    - Certificate of a Certificate Authority (CA) that has issued the other certificates
  - Your own certificate
    - A certificate that was issued to you by a certificate authority
  - Partner Certificate(s)
    - Certificate(s) of your communication partners
- When you do HTTPS with your browser usually already contains these keys and certificates

Secure Socket Layer - Concepts

- For production purposes, certificates are usually issued by a well known and trusted Certificate Authorities (CA)
  - For example Thawte, VeriSign
  - Usually this cost money
- For in-house use (Intranet), you can have your own Company-wide Certificate Authority
  - Certificates are trusted inside your company, but not outside
- For test purposes you can use self-signed Certificates (you are your own Certificate Authority)
  - Nobody trusts these Certificates (except you)

Secure Socket Layer - Setup

- To setup all required keys and certificates, it is recommended to use the Tool Keyman/VSE
  - Download from VSE Homepage
    http://www.ibm.com/servers/eserver/zseries/zvse/downloads/#vkeyman
- Supports creation of keys and CA-signed or self-signed Certificates for use with SSL
- Online documentation contains 'How to' sections with step by step descriptions for creating keys and certificates



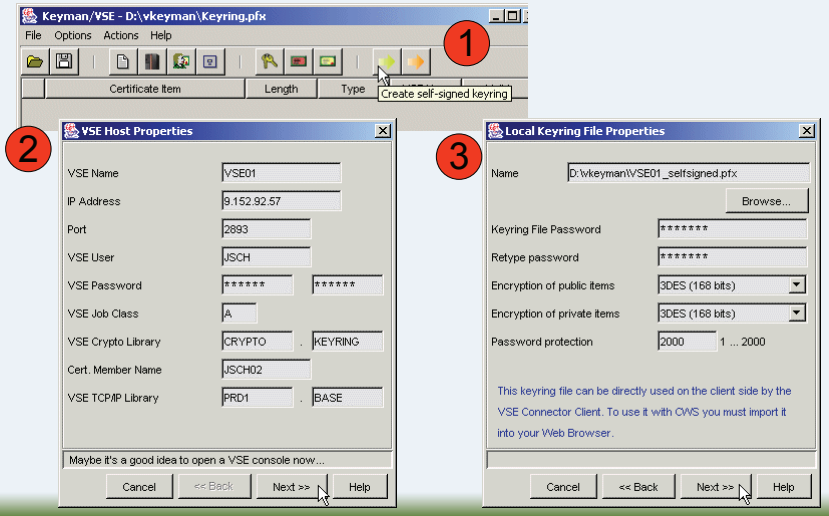
Setup a self signed certificate

- Steps for creating a self-signed certificate:

1. Create an RSA key pair
2. Create a self-signed root certificate
3. Create a VSE server certificate
4. Sign the request with your root certificate
5. Make your VSE host ready for uploading
6. Upload the key to VSE
7. Upload the root certificate to VSE
8. Upload the server certificate to VSE
9. Save your local keyring file

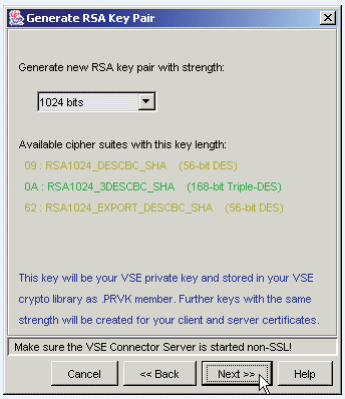
- Use Wizard Dialog “Create self-signed keyring”

Setup a self signed certificate - Wizard

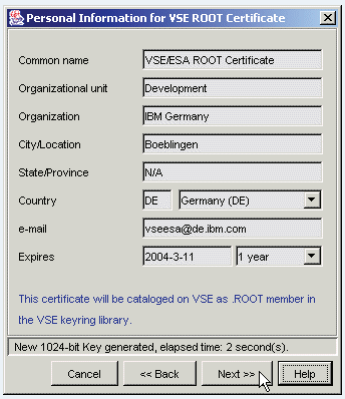


Setup a self signed certificate - Wizard

4

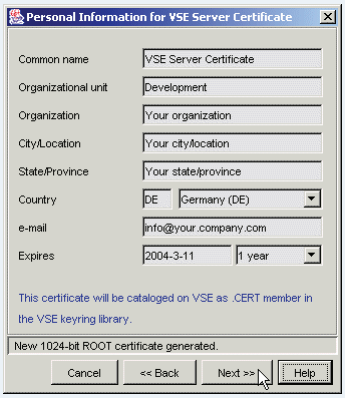


5

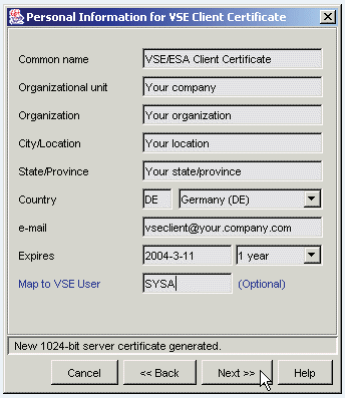


Setup a self signed certificate - Wizard

6

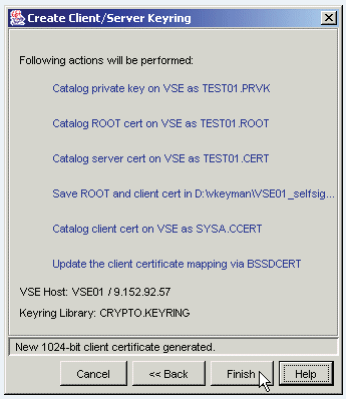


7

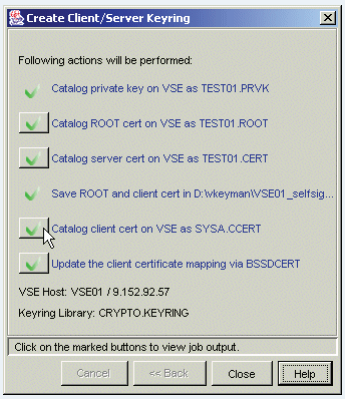


Setup a self signed certificate - Wizard

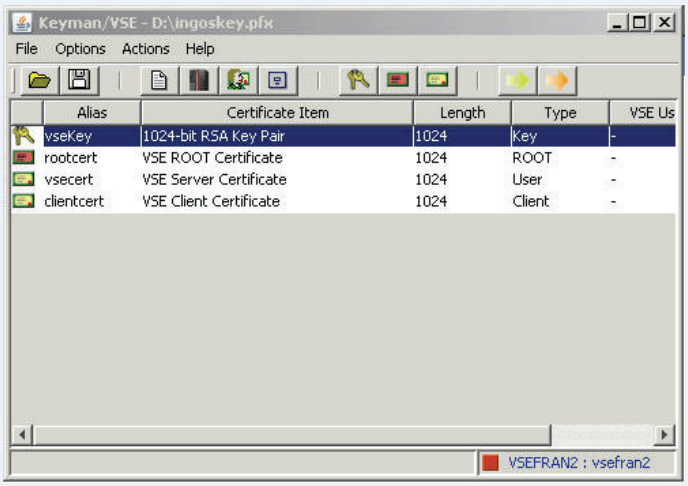
8



9



Setup a self signed certificate - Wizard



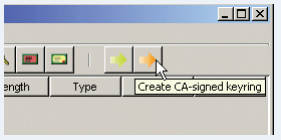
Setup a CA signed certificate

- **Steps for creating a CA signed certificate:**

1. **Create an RSA key pair**
2. **Create a certificate request**
3. **Copy request to clipboard**
4. **Go to the CA's web site (e.g. Thawte, VeriSign)**
5. **Request the server certificate on the CA's web site**
6. **Import signed server cert into Keyman/VSE**
7. **Get the CA's public root certificate**
8. **Make your VSE host ready for uploading**
9. **Upload the key to VSE**
10. **Upload the root certificate to VSE**
11. **Upload the server certificate to VSE**
12. **Save your local keyring file**

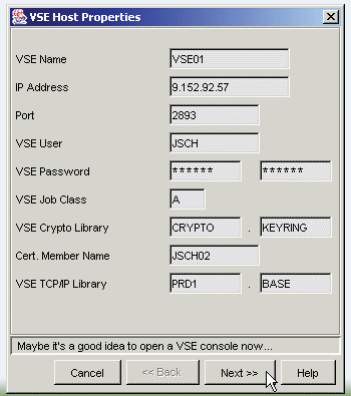
- **Use Wizard Dialog "Create CA signed keyring"**

Setup a CA signed certificate - Wizard

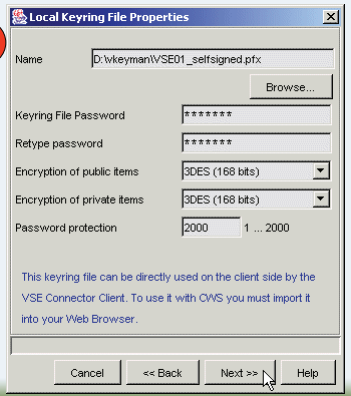


1

2

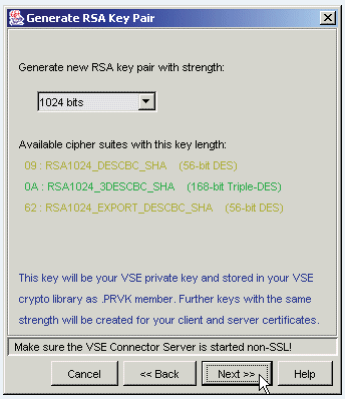


3

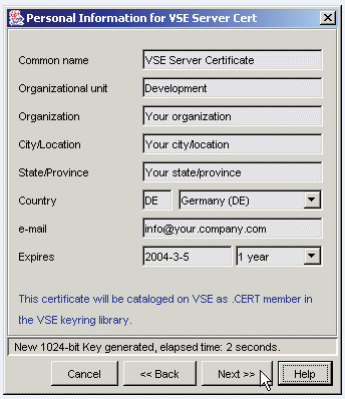


Setup a CA signed certificate - Wizard

4

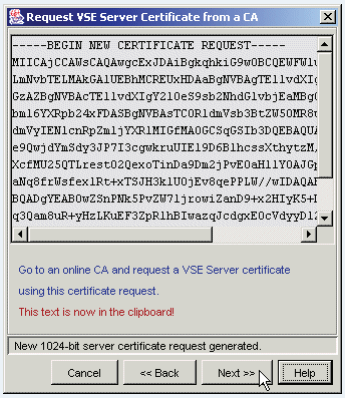


5



Setup a CA signed certificate - Wizard

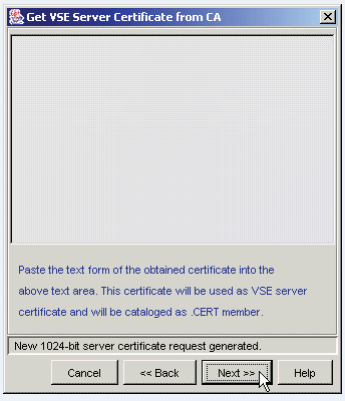
6



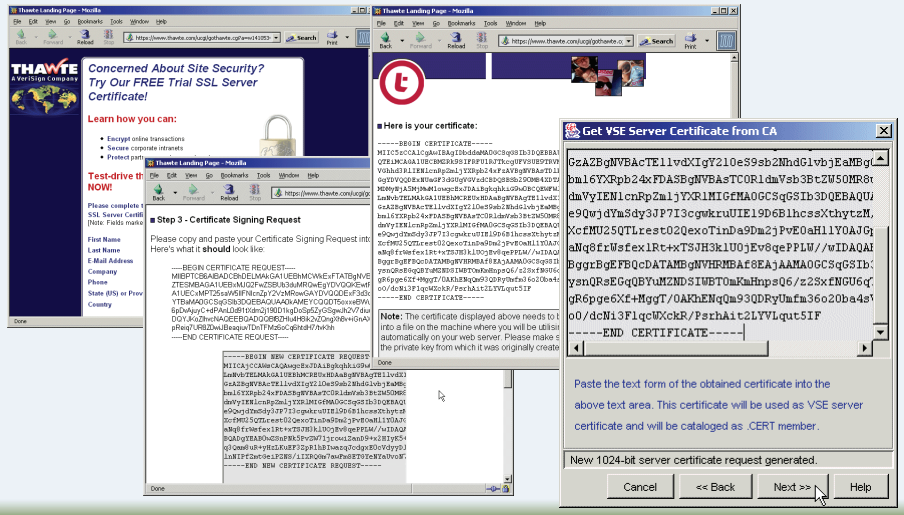
Copy and Paste the request into a CA's web site and let them sign the request.

You can paste the generated certificate into the text area on the following dialog box.

7



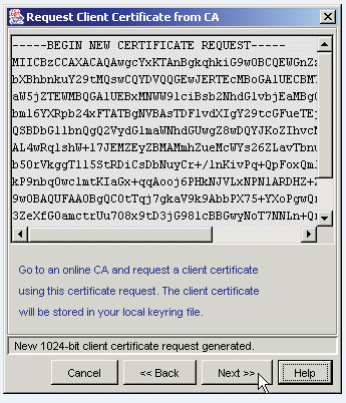
Setup a CA signed certificate - Wizard



Setup a CA signed certificate - Wizard

8

9

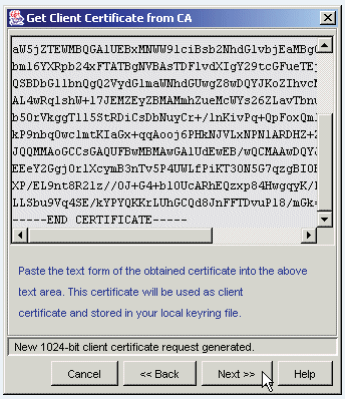


Copy and Paste the request into a CA's web site and let them sign the request.

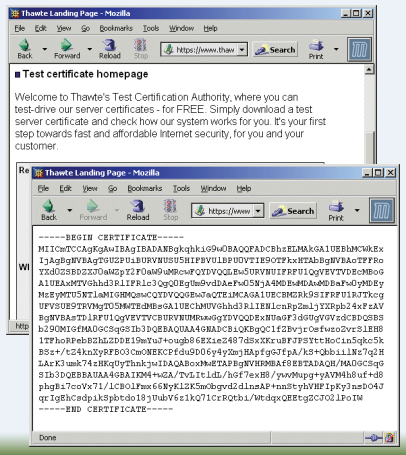
You can paste the generated certificate into the text area on the following dialog box.

Setup a CA signed certificate - Wizard

10



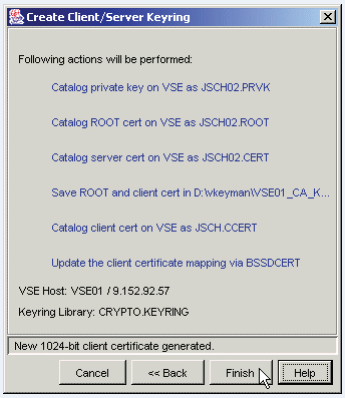
11



Setup a CA signed certificate - Wizard

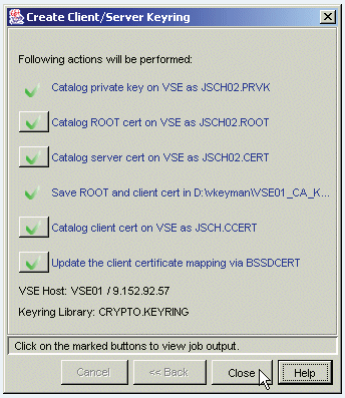
12

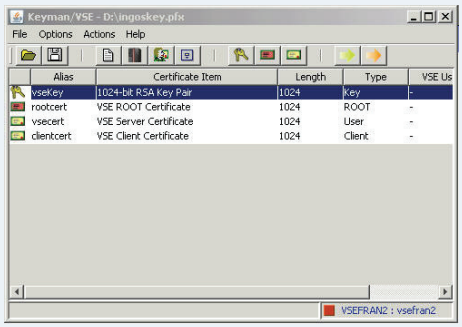
13



Setup a CA signed certificate - Wizard

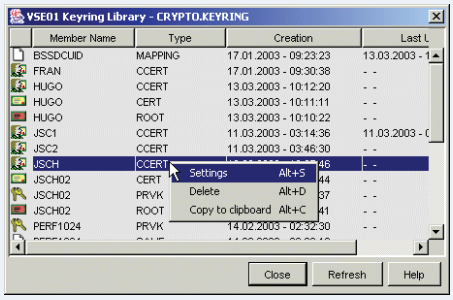
14





Where are keys and certificates stored on VSE ?

- **Keys and certificates are stored on a VSE Library**
  - **Usually in CRYPTO.KEYRING**
  - **This library should be secured using the VSE security mechanisms (access protection)**

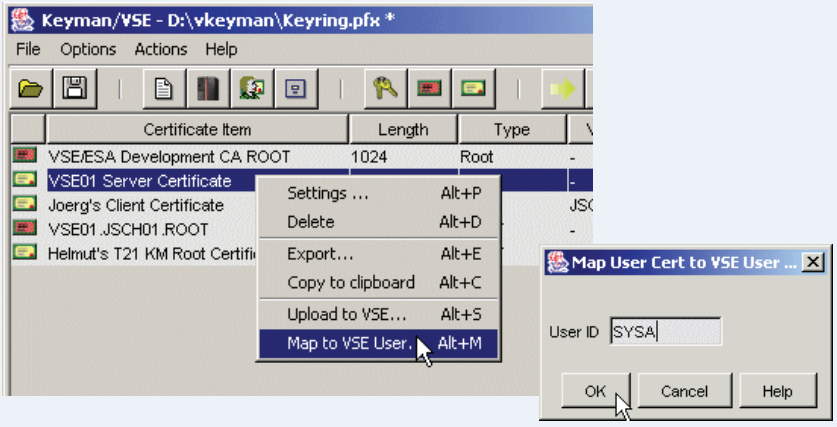


- **Member types:**
  - **.PRVK – Public/Private Key**
  - **.ROOT – Root Certificate**
  - **.CERT – Server Certificate**
  - **.CCERT – Client Certificate**
  - **BSSDCUID.MAPPING – Contains the User to Certificate mapping information**

SSL with client authentication

- **Server authentication means**
  - **The clients verifies the certificate received from the server**
  - **To make sure they are talking to the right server**
- **Client authentication means**
  - **The server verifies the certificates(s) received from the client(s)**
  - **To make sure only known clients can talk to the server**
  - **To do implicit logon by using the certificate (optional)**
    - **Map a user id to a certificate**

Map a VSE user id to a client certificate



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, DES encryption and SHA-1 secure hash functions**
  - **SecureFTP is integrated into TCP/IP for VSE with z/VSE V4.1 (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

Hardware Crypto Support on System z and VSE

by release

	z/VSE 4.1	z/VSE 3.1	VSE/ESA 2.7	VSE/ESA 2.6
PCICA	Yes	Yes	Yes	-
CEX2C	Yes	Yes	-	-
CPACF	Yes	Yes	-	-
CEX2A	Yes	Yes	-	-
PCIXCC	Yes	-	-	-

by server

	prior z800	z800	z900	z890	z990	z9
PCICA	-	Yes	Yes	Yes	Yes	-
PCIXCC	-	-	-	Yes	Yes	-
CEX2C	-	-	-	Yes	Yes	Yes
CPACF	-	-	-	Yes	Yes	Yes
CEX2A	-	-	-	-	-	Yes

CEX2C = Crypto Express2 in coprocessor mode
CEX2A = Crypto Express2 in accelerator mode
See: http://www.ibm.com/systems/z/security/cryptography.html

VSE Hardware Configuration

- 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 even TCP/IP applications
  - But use of crypto hardware can be disabled via TCP/IP SOCKOPT phase

HW-Crypto related console messages

- System with crypto hardware

```
FB 0095 1J023I FOUND A CRYPTO EXPRESS2 CARD AT DEVICE INDEX 0
FB 0095 1J023I FOUND A CRYPTO EXPRESS2 CARD AT DEVICE INDEX 1
FB 0095 1J014I FOUND A PCICA CARD AT DEVICE INDEX 6
FB 0095 1J014I FOUND A PCICA CARD AT DEVICE INDEX 7
FB 0095 1J005I HARDWARE CRYPTO ENVIRONMENT INITIALIZED SUCCESSFULLY.
FB 0095 1J006I USING CRYPTO DOMAIN 0
FB 0095 1J022I CPU CRYPTOGRAPHIC ASSIST FEATURE AVAILABLE.
```

- System without crypto hardware

```
FB 0093 1J020W THERE WAS NO PCICA OR CRYPTO EXPRESS2 CARD
FB 0093        FOUND. HARDWARE CRYPTO NOT AVAILABLE.
```

HW-Crypto status display

```
msg fb,data=status=cr
AR 0015 1I40I  READY
FB 0011 BST223I CURRENT STATUS OF THE SECURITY TRANSACTION SERVER:
FB 0011 ADJUNCT PROCESSOR CRYPTO SUBTASK STATUS:
FB 0011   AP CRYPTO SUBTASK STARTED .......... : YES
FB 0011   MAX REQUEST QUEUE SIZE ............. : 1
FB 0011   MAX PENDING QUEUE SIZE ............. : 1
FB 0011   TOTAL NO. OF AP REQUESTS ........... : 1234
FB 0011   NO. OF POSTED CALLERS .............. : 1234
FB 0011   AP CRYPTO POLLING TIME (1/300 SEC).. : 1
FB 0011   AP CRYPTO TRACE LEVEL ............... : 3
FB 0011   ASSIGNED APS : PCICC / PCICA ....... : 0 / 0
FB 0011                  CEX2C / CEX2A ....... : 1 / 2
FB 0011                  PCIXCC .............. : 0
FB 0011     AP  0 : CEX2C   - ONLINE
FB 0011     AP  4 : CEX2A   - ONLINE
FB 0011     AP  9 : CEX2A   - ONLINE
FB 0011   ASSIGNED AP QUEUE (CRYPTO DOMAIN)... : 6
FB 0011 CPU CRYPTOGRAPHIC ASSIST FEATURE:
FB 0011   CPACF AVAILABLE ..................... : YES
FB 0011   INSTALLED CPACF FUNCTIONS:
FB 0011     DES, TDES-128, TDES-192, SHA-1
FB 0011     AES-128
FB 0011     PRNG, SHA-256
FB 0011 END OF CPACF STATUS
```


Crypto HW exploitation in VSE

- Crypto cards are only used for RSA acceleration
  - RSA decrypt/encrypt for SSL session initiation
  - RSA encrypt for signing of certificates (CIALCREQ)
- CPACF
  - Acceleration of symmetric algorithms: DES, TDES, AES-128 (z9 only), SHA-1
  - Used at
    - SSL data transfer
    - CIAL functions in TCP/IP
- Usage is transparent for TCP/IP applications
  - If Crypto HW is available, it will be used. If not available, the SW implementation (as part of TCP/IP) will be used
  - You can disable the use of Crypto HW via a setting in $SOCKOPT Phase

Crypto HW exploitation in VSE

- **HW Crypto Functions that are not exploited in VSE**
  - **Special functions available in Coprocessor-Modus**
    - RSA Key-Generation
      - **RSA keys could be generated directly on VSE, no workstation tool would be required**
    - Secure Key functions
      - **PIN functions**
      - **Symmetric Key Import / Export (Key Transport)**
    - Special functions for banking-software
      - **ANSI X9.17 Standard: Key generate, export, import**
- **Requirements are welcome !**

Secure Key vs. Clear Key

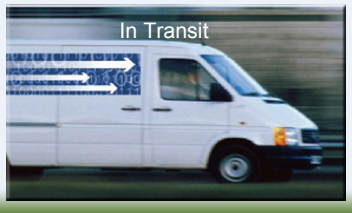
- **Different way of managing, storing and usage of keys**
  - **Keys reside unencrypted (clear) in the file system ("Clear Key")**
  - **Keys reside encrypted (TDES with fixed key) in the file system**
    - **➔ That's how VSE works today**
  - **Keys reside encrypted (using a "Secure Master Key") in the file system**
    - **The Master Key is stored in the hardware**
    - **Secure master key entry via TKE or Dialogs**
    - **Crypto operations are done in main storage, i.e. data keys are visible (unencrypted) in main storage for a very short time**
    - **Crypto operations are done on a coprocessor card, i.e. data keys will never reside unencrypted in the main storage**
      - **➔Required for banking applications, e.g. PIN Verification**
      - **➔Supported by z/OS ICSF**

CryptoVSE API

- **Native cryptographic API**
- **Provides cryptographic services:**
  - **Data encryption**
    - **DES**
    - **Triple DES**
    - **AES**
    - **RSA PKCS #1**
  - **Message Digest**
    - **MD5**
    - **SHA-1**
  - **Digital Signatures**
    - **RSA PKCS #1 with SHA1 or MD5**
  - **Message Authentication**
    - **HMAC**
- **Uses Hardware Crypto functions transparently when available**

Customer Data Protection Requirements

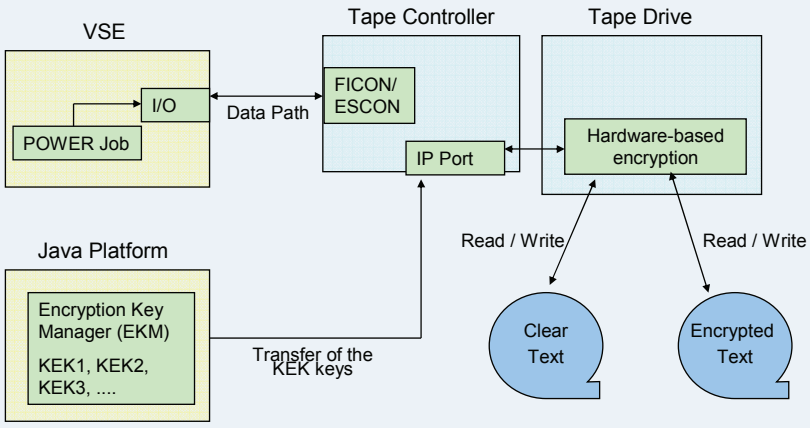
- Regulatory requirements driving need for greater data security, integrity, retention/auditability, and privacy
- Severe business impacts caused by loss or theft of data including financial liability, reputation damage, legal/compliance risk
- Increasing need to share data securely with business partners and maintain backups at remote locations
- Need to reduce complexity and improve processes around enterprise encryption management
- Need ability to cost effectively encrypt large quantities of tape data



IBM Tape Encryption – TS1120

- **The IBM System Storage TS1120 Tape Drive has been enhanced to provide drive based data encryption**
- **A new, separate IBM Encryption Key Manager component for the Java Platform (Encryption Key Manager) program is also being introduced:**
  - **supports the generation and communication of encryption keys for the tape drives across the enterprise.**
- ***New:* Support is now available for z/VSE V4 and V3:**
  - ***z/VSE V4.1:* DY46682 *(UD53141 and UD53142)***
  - ***z/VSE V3.1:* DY46685 *(UD53143,UD53144, UD53146) and* PK43473 *(UK24398)***
  - ***z/VM:* VM64062 *(UM32012)***
  - ***DITTO:* PK44172 *- With this Apar, DITTO/ESA for VSE supports tape encryption interactively and via standard VSE JCL in BATCH mode***

IBM Tape Encryption – TS1120



IBM Tape Encryption – TS1120

encryption mode
(03=write)

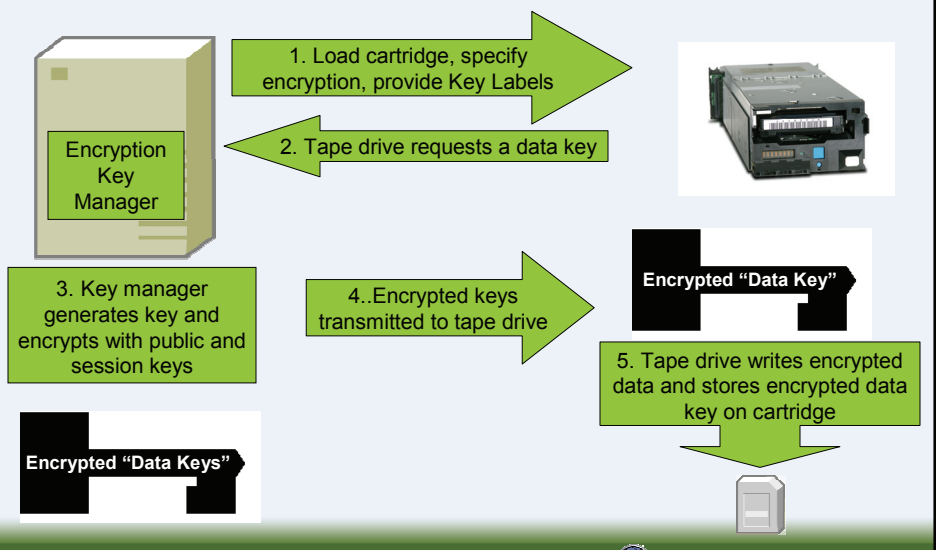
```
// JOB ENCRYPT
// ASSGN SYS005,480,03
// KEKL UNIT=480,KEKL1='MYKEKL1',KEM1=L,KEKL2='MYKEKL2',KEM2=L
// EXEC LIBR
  BACKUP LIB=PRD2 TAPE=SYS005
/*
/&
```

encoding mechanism
(L=Label, H=Hash)

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

- The Data-Key can be encrypted using 2 different public keys (KEK = Key Encrypting Keys), to be able to send the tape to 2 different receivers
- More info can be found in the *z/VSE 4.1 Administration* manual (VSE Homepage)

IBM Tape Encryption – TS1120



IBM Tape Encryption – TS1120

- Considerations and Restrictions:
  - A tape can either contain encrypted data or unencrypted data
  - If you encrypt the first file on the tape, all subsequent files will also be encrypted using the same key
    - Important for multi file tapes
  - If you send an encrypted tape to a business partner, the other side will also require a TS1120 to be able to read the tape

Other ways to encrypt your backups or tapes

- Can be done using VTAPE
  - Create a backup on a remote virtual tape
  - Store the tape image on an encrypted medium
    - Encrypted file system or directory (e.g. EcryptFS on Linux)
    - Use encryption tools (e.g. TrueCrypt)
    - Use Tivoli Storage Manager to store the backup data
- Encrypt data in applications
  - Use CryptoVSE API to encrypt the data
    - Uses Hardware Crypto Support if available

Related Documentation

- IBM System z cryptography for highly secure transactions
  - http://www.ibm.com/systems/z/security/cryptography.html
- VSE Security Homepage
  - http://www.ibm.com/servers/eserver/zseries/zvse/documentation/security.html
- z/VSE Planning
- z/VSE Administration
- VSE/ESA Software Newsletter No. 17, 18 and 20
- OS/390 Security Server External Security Interface (RACROUTE) Macro Reference (GC28-1922)
- OS/390 Security Server (RACF) Data Areas (SY27-2640)
- z/VSE V4R1.0 e-business Connectors, User's Guide
- CICS Enhancements Guide, GC34-5763
- VSE/ESA 2.7.3 Release Guide, Chapter 1, section "Hardware Crypto Support"

Questions ?