

# Security on IBM z/VSE

Helmut Hellner

Ingo Franzki

Antoinette Kaschner

Joerg Schmidbauer

Heiko Schnell

Klaus-Dieter Wacker







International Technical Support Organization

**Security on IBM z/VSE**

June 2018

**Note:** Before using this information and the product it supports, read the information in “Notices” on page xi.

**Fourth Edition (June 2018)**

This edition (SG24-7691-02) applies to IBM z/VSE V6R2.

**© Copyright International Business Machines Corporation 2009, 2011, 2018. All rights reserved.**

Note to U.S. Government Users Restricted Rights -- Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.



# Contents

<b>Notices</b> .....	xi
Trademarks .....	xii
<b>Preface</b> .....	xiii
Authors .....	xiii
Now you can become a published author, too! .....	xiv
Comments welcome .....	xv
Stay connected to IBM Redbooks .....	xv
<b>Summary of changes</b> .....	xvii
June 2018, Fourth Edition .....	xvii
November 2011, Third Edition .....	xvii
October 2009, Second Edition .....	xvii
<b>Chapter 1. z/VSE and security</b> .....	1
1.1 Introducing the z/VSE parts .....	2
1.1.1 Using z/VSE .....	2
1.1.2 How z/VSE stores data .....	4
1.2 z/VSE security features .....	5
1.2.1 Online security .....	5
1.2.2 Batch security .....	5
1.2.3 Basic Security Manager .....	6
1.2.4 Single sign-on and LDAP .....	7
1.2.5 System z cryptographic solution .....	7
1.2.6 CICS Web Support .....	8
1.2.7 Connector security .....	8
1.2.8 TCP/IP security .....	8
1.2.9 Secure FTP .....	8
1.2.10 Intrusion detection .....	9
1.2.11 Compliance to policy .....	9
<b>Chapter 2. z/VSE Basic Security Manager</b> .....	11
2.1 BSM concept .....	12
2.1.1 System Authorization Facility .....	12
2.1.2 Security files .....	13
2.1.3 Security server partition .....	15
2.1.4 BSM processing .....	15
2.1.5 Common startup for BSM and ESM .....	16
2.2 Installing and customizing BSM .....	16
2.3 BSM administration .....	17
2.3.1 Security system settings .....	17
2.3.2 Defining a User .....	23
2.3.3 Group definition .....	30
2.3.4 Resource profile definition .....	34
2.3.5 Batch resource administration .....	44
2.3.6 Generating BSM cross-reference reports .....	48
2.4 BSM auditing .....	50
2.4.1 Enabling auditing for resources defined in the BSM control file .....	51
2.4.2 Enabling auditing for resources defined in the DTSECTAB .....	52

2.4.3 DMF setup .....	52
2.4.4 BSM report writer (BSTRPWTR) .....	58
2.5 BSM backups .....	60
2.5.1 VSAM backups .....	61
2.5.2 BSM backup and migration with BSTSAVER .....	62
<b>Chapter 3. LDAP sign-on support</b> .....	<b>65</b>
3.1 LDAP and z/VSE .....	66
3.2 Risks of the current situation .....	67
3.3 LDAP terminology .....	68
3.3.1 Overview and terms .....	68
3.3.2 LDIF files .....	71
3.4 z/VM LDAP server .....	72
3.5 LDAP sign-on of z/VSE .....	72
3.5.1 LDAP user mapping file .....	74
3.5.2 Strict mode .....	75
3.5.3 LDAP password cache .....	75
3.6 Configure and activate LDAP sign-on support .....	75
3.6.1 LDAP configuration example skeleton .....	76
3.6.2 Sign on to z/VSE with active LDAP sign-on support .....	81
3.7 Administering the LDAP user mapping file .....	83
3.7.1 Using the Maintain LDAP user profiles dialog .....	85
3.8 LDAP sample setup .....	87
3.8.1 Modifying the LDAP configuration phase .....	87
3.8.2 Mapping an intranet user ID to a z/VSE user ID .....	87
3.8.3 Modifying the TCP/IP setup .....	88
3.8.4 Setting up for SSL .....	88
3.8.5 Observations .....	89
<b>Chapter 4. Cryptography on z/VSE</b> .....	<b>93</b>
4.1 Cryptography introduction .....	95
4.1.1 Modern cryptography .....	95
4.1.2 Encryption modes .....	96
4.1.3 Verifying the identity of communication partners .....	96
4.1.4 Ensuring data integrity .....	97
4.1.5 Secure Sockets Layer and Transport Layer Security .....	98
4.1.6 Use of certificates .....	99
4.1.7 Comparison of key sizes .....	100
4.1.8 Password-based encryption .....	101
4.1.9 Public key encryption .....	101
4.2 Configuring cryptographic hardware .....	102
4.2.1 Hardware overview .....	103
4.2.2 Planning your crypto configuration .....	104
4.2.3 Configuring LPAR activation profile .....	105
4.2.4 CPC cryptographic configuration .....	107
4.2.5 LPAR cryptographic configuration .....	109
4.2.6 Hardware crypto device driver in z/VSE .....	110
4.2.7 Disabling a crypto device .....	112
4.2.8 Cryptography for guests on z/VM .....	122
4.2.9 Cryptography when using an external security manager .....	124
4.2.10 Changing the status of hardware-based encryption .....	124
4.2.11 AP-queue Adapter Interruption Facility .....	126
4.3 Hardware-based tape encryption with z/VSE .....	126

4.3.1	Encrypting data . . . . .	127
4.3.2	Decrypting data . . . . .	129
4.3.3	z/VSE considerations . . . . .	129
4.3.4	Hardware and software requirements . . . . .	130
4.3.5	Writing and reading encrypted data in z/VSE . . . . .	130
4.3.6	Recognizing an encrypted tape . . . . .	132
4.3.7	More information about using hardware-based tape encryption . . . . .	133
4.4	Example of TS1120 installation. . . . .	134
4.4.1	Installing the prerequisite programs . . . . .	134
4.4.2	Setting up the TS1120 . . . . .	135
4.4.3	Setting up the EKM . . . . .	135
4.4.4	z/VSE considerations . . . . .	143
4.4.5	Observations . . . . .	144
4.5	Software-based encryption with Encryption Facility for z/VSE V1R1 . . . . .	145
4.5.1	Performance considerations . . . . .	148
4.5.2	Password-based encryption . . . . .	148
4.5.3	Public key encryption . . . . .	151
4.6	Software-based encryption with Encryption Facility for z/VSE V1R2 . . . . .	160
4.6.1	Prerequisites . . . . .	161
4.6.2	Differences in Encryption Facility between z/VSE V1R1 and V1R2 . . . . .	161
4.6.3	Downloading the prerequisite programs . . . . .	162
4.6.4	Usage hints . . . . .	163
4.6.5	Flexible support of record and stream data. . . . .	163
4.6.6	Considerations on compression . . . . .	164
4.6.7	Password-based encryption . . . . .	164
4.6.8	Public key encryption . . . . .	168
4.6.9	Advanced encryption options . . . . .	190
4.6.10	Observation. . . . .	193
4.7	z/VSE Navigator GUI for Encryption Facility . . . . .	193
<b>Chapter 5. Secure Sockets Layer with z/VSE . . . . .</b>		<b>197</b>
5.1	Generating the server key and certificates . . . . .	198
5.1.1	Defining the properties of the z/VSE system. . . . .	198
5.1.2	Creating the z/VSE key and certificates . . . . .	200
5.2	SSL setup for Java-based connector . . . . .	205
5.2.1	Setting up z/VSE Connector Server for SSL. . . . .	205
5.2.2	Setting up z/VSE Navigator for SSL . . . . .	208
5.2.3	Connecting to z/VSE by using SSL server authentication . . . . .	210
5.2.4	Considerations with client authentication . . . . .	212
5.2.5	Using encryption with AES-256. . . . .	212
5.3	SSL setup for web browsers . . . . .	214
5.3.1	Setting up SSL native mode with HTTPD. . . . .	215
5.3.2	Considerations on \$WEB user . . . . .	216
5.3.3	Connecting to HTTPD by using a web browser . . . . .	217
5.3.4	Configuring ciphers in Internet Explorer . . . . .	218
5.4	Debugging SSL/TLS connections . . . . .	218
5.4.1	Tracing on z/VSE . . . . .	218
5.4.2	Tracing in Java . . . . .	219
<b>Chapter 6. CICS Web Support security . . . . .</b>		<b>221</b>
6.1	Introduction . . . . .	222
6.2	Setting up CWS. . . . .	222
6.2.1	Defining the TCP/IP service . . . . .	223

6.2.2	Connecting to CWS .....	224
6.3	Setting up secure CWS .....	224
6.3.1	Configuring the TCP/IP service for SSL .....	225
6.3.2	Configuring the CICS system initialization parameters .....	226
6.3.3	Configuring OpenSSL .....	228
6.4	Client setup with Mozilla Firefox .....	228
6.4.1	Importing the z/VSE certificates during session establishment .....	229
6.4.2	Manually importing the z/VSE certificates into Firefox .....	230
6.4.3	Configuring cipher suites in Firefox .....	235
6.4.4	Starting a secure session with Firefox .....	236
6.4.5	Displaying SSL properties in Mozilla Firefox .....	237
6.5	Client setup with Microsoft Internet Explorer .....	237
6.5.1	Importing the z/VSE certificates during session establishment .....	238
6.5.2	Manually importing the z/VSE certificates into Internet Explorer .....	239
6.5.3	Configuring cipher suites in Internet Explorer .....	242
6.5.4	Starting a secure session with Internet Explorer .....	242
6.6	Setting up for client authentication .....	243
6.6.1	Using Internet Explorer .....	244
6.6.2	Client authentication with user ID mapping .....	245
6.7	Observations .....	248
6.7.1	Abend AKEA in DFHSEOSE .....	248
6.7.2	Abend code x'080C' in module DFHSEOSE .....	249
<b>Chapter 7.</b>	<b>Connector security .....</b>	<b>251</b>
7.1	Java-based connector security .....	252
7.1.1	Security features of the Java-based connector .....	253
7.2	z/VSE script connector security .....	255
7.2.1	Security features of the z/VSE script connector .....	256
7.2.2	Non-SSL setup with client on workstation .....	257
7.2.3	Non-SSL setup with a client on z/VSE .....	264
7.2.4	General SSL setup .....	265
7.2.5	SSL setup with client on z/VSE .....	267
7.2.6	Observations .....	271
7.2.7	Debugging hints .....	273
7.3	Web service security when using SOAP .....	273
7.3.1	Transport Layer Security and message layer security .....	275
7.3.2	Web service security features with z/VSE as the SOAP server .....	278
7.3.3	Web service security features with z/VSE as the SOAP client .....	279
7.4	z/VSE Database Connector (DBCLI) .....	280
7.4.1	z/VSE DBCLI security features .....	281
<b>Chapter 8.</b>	<b>TCP/IP security .....</b>	<b>283</b>
8.1	TCP/IP security concept .....	284
8.1.1	Control the security functions with the SECURITY command .....	285
8.2	Defining user IDs .....	287
8.2.1	Explicitly defining user IDs .....	287
8.3	Security exit points and security managers .....	288
8.3.1	Flow of a security request .....	289
8.3.2	Using Basic Security Manager with TCP/IP .....	289
<b>Chapter 9.</b>	<b>Secure Telnet .....</b>	<b>293</b>
9.1	Introduction .....	294
9.2	Setting up a Telnet daemon, TELNETD .....	294

9.3	z/VSE host setup for secure Telnet. . . . .	296
9.3.1	Setting up pass-through mode with a TLS D . . . . .	297
9.3.2	Setting up SSL native mode . . . . .	297
9.3.3	Setting up a Telnet listener daemon . . . . .	298
9.4	Client setup with Personal Communications. . . . .	298
9.4.1	Importing the z/VSE certificates into Personal Communications . . . . .	298
9.4.2	Starting a secure session . . . . .	304
9.4.3	Setting up for client authentication . . . . .	305
9.4.4	Taking a Personal Communications trace . . . . .	308
9.5	Client setup with Attachmate EXTRA! X-treme. . . . .	309
9.5.1	Importing certificates into the Windows certificate store . . . . .	310
9.5.2	Attachmate EXTRA! session setup . . . . .	312
9.5.3	Viewing the log . . . . .	314
9.5.4	Setting up for client authentication . . . . .	315
<b>Chapter 10. Secure File Transfer Protocol . . . . .</b>		<b>319</b>
10.1	Introduction . . . . .	320
10.2	z/VSE as FTP server. . . . .	320
10.2.1	Set up and start the z/VSE FTP server. . . . .	320
10.2.2	z/VM considerations . . . . .	321
10.2.3	Connect to z/VSE by using an FTP client . . . . .	322
10.2.4	Transferring the certificate to the client side . . . . .	323
10.3	z/VSE as FTP client . . . . .	324
10.3.1	Sample setup with FileZilla server . . . . .	324
10.3.2	Sample setup with vsftpd server on Linux. . . . .	335
10.4	Considerations for firewalls. . . . .	340
10.4.1	Passive versus active FTP mode . . . . .	340
10.4.2	Restricting the port range on the server side . . . . .	341
10.4.3	Restricting the port range on the client side . . . . .	342
10.4.4	Considerations on the DATAPORT parameter . . . . .	342
10.4.5	Firewall configuration . . . . .	343
10.5	Observations. . . . .	343
10.5.1	Cannot submit a VSE/POWER job with Keyman/VSE . . . . .	343
10.5.2	SSL handshaking fails . . . . .	344
<b>Chapter 11. WebSphere MQ with SSL . . . . .</b>		<b>345</b>
11.1	Introduction . . . . .	346
11.2	Installing WebSphere MQ . . . . .	346
11.2.1	MQ installation on z/VSE . . . . .	346
11.2.2	Maintaining security profiles . . . . .	346
11.2.3	MQ installation on Windows . . . . .	348
11.3	Configuring WebSphere MQ. . . . .	356
11.3.1	MQ configuration on z/VSE. . . . .	356
11.3.2	MQ configuration on Windows . . . . .	362
11.3.3	Testing the setup . . . . .	375
11.4	Configuring for SSL. . . . .	378
11.4.1	Creating the keys and certificates . . . . .	378
11.4.2	SSL configuration on z/VSE . . . . .	389
11.4.3	SSL configuration on Windows . . . . .	391
11.5	Implementing SSL client authentication . . . . .	393
11.5.1	Configuring for client authentication on z/VSE . . . . .	393
11.5.2	Configuring for client authentication on Windows . . . . .	393
11.6	Using SSL peer attributes . . . . .	393

11.6.1	Example 1: Specifying matching peer attributes . . . . .	395
11.6.2	Example 2: Specifying peer attributes that do not match . . . . .	395
11.7	Configuring a z/VSE queue manager remotely . . . . .	396
11.7.1	What you can do remotely . . . . .	396
11.7.2	Preparing the z/VSE side for PCF . . . . .	397
11.7.3	Defining more queues . . . . .	397
11.7.4	Defining the MQ Explorer reply model queue . . . . .	398
11.7.5	Defining a server-connection channel . . . . .	399
11.7.6	Defining a remote queue manager . . . . .	399
11.7.7	Exchanging test messages . . . . .	404
11.7.8	Defining SSL . . . . .	406
11.8	Observations . . . . .	409
11.8.1	Message sequence number error . . . . .	409
11.8.2	RC=2092 when sending a test message to Windows . . . . .	409
11.8.3	Open of file MQFADMN failed . . . . .	410
11.8.4	No space available for PUT request . . . . .	411
<b>Appendix A. Security APIs . . . . .</b>		<b>413</b>
A.1	Client-side Java APIs . . . . .	414
A.1.1	z/VSE Connector Client . . . . .	414
A.1.2	Security class library . . . . .	414
A.2	Host-side APIs . . . . .	414
A.2.1	Using APIs to write your own SSL/TLS applications . . . . .	414
A.2.2	CPU Assist Facility . . . . .	416
A.2.3	Summary of available SSL functions . . . . .	416
<b>Appendix B. Setting up and using Keyman/VSE . . . . .</b>		<b>417</b>
B.1	Keyman/VSE . . . . .	418
B.2	Installing the prerequisite programs . . . . .	418
B.3	Initial Keyman/VSE set-up . . . . .	419
B.4	Basic characteristics of RSA keys . . . . .	420
B.4.1	Symmetric and asymmetric keys . . . . .	420
B.4.2	Internal structure of RSA keys . . . . .	420
B.4.3	Types of SSL certificates . . . . .	421
B.4.4	Structure of a keyring . . . . .	421
B.5	Relationship to TCP/IP utilities . . . . .	422
B.5.1	CIALSRVR . . . . .	422
B.5.2	CIALROOT . . . . .	423
B.5.3	CIALCERT . . . . .	423
B.5.4	CIALSIGV . . . . .	424
B.5.5	CIALCREQ . . . . .	425
B.6	Keystores . . . . .	425
B.7	Using Keyman/VSE . . . . .	426
B.7.1	Support a CIALEXIT phase . . . . .	427
B.7.2	Activating a CIAL trace . . . . .	429
B.8	Selected Keyman/VSE functions . . . . .	429
B.8.1	Creating keys . . . . .	430
B.8.2	Creating certificates . . . . .	431
B.8.3	Using certificate mappings . . . . .	433
B.8.4	Displaying key material . . . . .	435
B.9	Observation . . . . .	437
B.9.1	SSL203E RSAD failed . . . . .	437
<b>Related publications . . . . .</b>		<b>439</b>

IBM Redbooks publications .....	439
IBM Knowledge Center .....	439
Online resources .....	439
How to get IBM Redbooks publications .....	441
Help from IBM .....	441
<b>Index</b> .....	<b>443</b>





# Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

*IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785 U.S.A.*

**The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:** INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of 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.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.


## COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

# Trademarks

IBM, the IBM logo, and [ibm.com](http://www.ibm.com) are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. These and other IBM trademarked terms are marked on their first occurrence in this information with the appropriate symbol (® or ™), indicating US registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at <http://www.ibm.com/legal/copytrade.shtml>

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

AIX®	Lotus®	Tivoli®
CICS®	MQSeries®	VTAM®
DB2®	OS/390®	WebSphere®
DS8000®	POWER®	Word Pro®
FlashCopy®	RACF®	z/OS®
IBM®	Redbooks®	z/VM®
IBM Blue®	Redbooks (logo)  ®	z/VSE®
IBM Z®	S/390®	z10™
IBM z13®	System Storage®	z13®
IBM z14™	System z®	z9®
Interconnect®	System z10®	zEnterprise®
Language Environment®	System z9®	

The following terms are trademarks of other companies:

Intel, Intel logo, Intel Inside logo, and Intel Centrino logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java, and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

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

Other company, product, or service names may be trademarks or service marks of others.

# Preface

One of a firm's most valuable resources is its data: client lists, accounting data, employee information, and so on. This critical data must be securely managed and controlled, and simultaneously made available to those users authorized to see it.

The IBM® z/VSE® system features extensive capabilities to simultaneously share the firm's data among multiple users and protect them. Threats to this data come from various sources. Insider threats and malicious hackers are not only difficult to detect and prevent, they might be using resources with the business being unaware.

This IBM Redbooks® publication was written to assist z/VSE support and security personnel in providing the enterprise with a safe, secure and manageable environment.

This book provides an overview of the security that is provided by z/VSE and the processes for the implementation and configuration of z/VSE security components, Basic Security Manager (BSM), IBM CICS® security, TCP/IP security, single sign-on using LDAP, and connector security.

## Authors

This book was produced by the z/VSE development team in Boeblingen, Germany and specialists from around the world:

**Helmut Hellner** is a z/VSE Developer in the IBM Development Laboratory, Boeblingen, Germany. He studied Computer Science at the University of Stuttgart, graduating in 1981. After several years of working with IBM OS/390® and VM, he joined the z/VSE Development and Service Team. His focus is on security in z/VSE.

**Ingo Franzki** is a z/VSE Developer in the IBM Development Laboratory, Boeblingen, Germany. He joined IBM in 1997. His main focus is on z/VSE connectors, security, and performance. He is also consultant for solutions that involve distributed environments with z/VSE. Ingo is often a speaker at IBM events, user group meetings, and workshops.

**Antoinette Kaschner** is a z/VSE Developer in the IBM Development Laboratory, Boeblingen, Germany. She holds a degree in Mathematics. Her main focus is on z/VSE I/O Development, especially tape and tape library support, Virtual Tape, IBM FlashCopy®, and Basic Access Methods (BAM).

**Joerg Schmidbauer** is a z/VSE Developer in the IBM Development Laboratory, Boeblingen, Germany. He has worked for IBM since 1989 in VSE Development. In addition to z/VSE connector-related work, he works on z/VSE hardware cryptographic support. He speaks at many client events.

**Heiko Schnell** is a z/VSE Developer in the IBM Development Laboratory, Boeblingen, Germany. He studied Computer Engineering at the Faculty of Information Technology at Esslingen University of Applied Sciences, graduating in 1996. Since 1997 he has worked in VSE Development; his areas of expertise include IBM Language Environment® and TCP/IP. Heiko also lectured in Software Engineering at the University of Cooperative Education in Stuttgart between 2001 and 2008.

**Klaus-Dieter Wacker** is a z/VSE Developer in the IBM Development Laboratory, Boeblingen, Germany. He has worked on various development positions in z/VSE and on Linux on IBM z.

Thanks to the following people for their contributions to this project:

Wolfgang Bosch  
Hans-Joerg Dietmann  
Bret Dixon  
Michael Holzheu  
Gerald (Jerry) Johnston  
Dagmar Kruse  
Silvano Prez  
Ingolf Salm  
Hans-Ulrich Schmidt  
Gerhard Schneidt  
Robert J. Sodan  
Edmund Wilhelm

Special thanks to the ITSO center in Poughkeepsie, New York:

Mike Ebbers, Project Leader  
Diane Sherman, Editor  
Alfred Schwab, Editor

## **Now you can become a published author, too!**

Here's an opportunity to spotlight your skills, grow your career, and become a published author - all at the same time! Join an ITSO residency project and help write a book in your area of expertise, while honing your experience using leading-edge technologies. Your efforts will help to increase product acceptance and customer satisfaction, as you expand your network of technical contacts and relationships. Residencies run from two to six weeks in length, and you can participate either in person or as a remote resident working from your home base.

Find out more about the residency program, browse the residency index, and apply online at:

[ibm.com/redbooks/residencies.html](http://ibm.com/redbooks/residencies.html)

## Comments welcome

Your comments are important to us!

We want our books to be as helpful as possible. Send us your comments about this book or other IBM Redbooks publications in one of the following ways:

- ▶ Use the online **Contact us** review Redbooks form found at:

[ibm.com/redbooks](http://ibm.com/redbooks)

- ▶ Send your comments in an email to:

[redbooks@us.ibm.com](mailto:redbooks@us.ibm.com)

- ▶ Mail your comments to:

IBM Corporation, International Technical Support Organization  
Dept. HYTD Mail Station P099  
2455 South Road  
Poughkeepsie, NY 12601-5400

## Stay connected to IBM Redbooks

- ▶ Find us on Facebook:

<http://www.facebook.com/IBMRedbooks>

- ▶ Follow us on Twitter:

<http://twitter.com/ibmredbooks>

- ▶ Look for us on LinkedIn:

<http://www.linkedin.com/groups?home=&gid=2130806>

- ▶ Explore new Redbooks publications, residencies, and workshops with the IBM Redbooks weekly newsletter:

<https://www.redbooks.ibm.com/Redbooks.nsf/subscribe?OpenForm>

- ▶ Stay current on recent Redbooks publications with RSS Feeds:

<http://www.redbooks.ibm.com/rss.html>



# Summary of changes

This section describes the technical changes that were made in this edition of the book and in previous editions. This edition can also include minor corrections and editorial changes that are not identified.

Summary of Changes  
for SG24-7691-03  
for *Security on IBM z/VSE*  
as created or updated on June 14, 2018.

## June 2018, Fourth Edition

This revision reflects the addition, deletion, or modification of new and changed information described here.

### **New and changed information**

- ▶ Explaining crypto setup on z14
- ▶ Added the BSM batch resource administration (DTSECTAB dialog) in z/VSE 6.2
- ▶ Added the BSM AUDITOR attribute
- ▶ Extended the LDAP sign-on support with the z/VSE 6.2 extensions
- ▶ CICS web support with OpenSSL in z/VSE 6.2
- ▶ Connector security with DBCLI in z/VSE 5.1 and with updates in z/VSE 6.1

## November 2011, Third Edition

This revision reflects the addition, deletion, or modification of new and changed information described here.

### **New and changed information**

This edition adds information about security enhancements in z/VSE V4R3 and setup information for SSL and Keyman/VSE.

## October 2009, Second Edition

This revision reflects the addition, deletion, or modification of new and changed information described below.

### **New information**

- ▶ Added information about PGP privacy in 4.6, “Software-based encryption with Encryption Facility for z/VSE V1R2”

### **Changed information**

- ▶ Chapter 4, “Cryptography on z/VSE” on page 93 is reworked.
- ▶ Typographical errors are fixed.







# z/VSE and security

In this chapter, we describe the security aspects of z/VSE facilities and introduce how z/VSE can share data among multiple users and protect them. We also describe online security and batch security on z/VSE and how the z/VSE Basic Security Manager (BSM) supports them.

z/VSE is designed to provide robust, cost-effective solutions for clients with a wide range of processor capacity requirements. In particular, clients with lower processor capacity requirements value the relatively small cost of operation and administration.

As with all IBM Z® mainframe operating systems, z/VSE can run as a single operating system or share the mainframe with others by using the LPAR capabilities of IBM System z®. z/VSE can also run as a guest on the IBM z/VM® system to use a customized environment that might contain emulated hardware features that are not necessarily installed on that particular model of the server.

The Transmission Control Protocol/Internet Protocol (TCP/IP)<sup>1</sup> in z/VSE allows interconnection to other platforms. With TCP/IP, z/VSE became accessible through the internet. Special security functions are provided to make those connections secure.

TCP/IP enables z/VSE to support single sign-on with Lightweight Directory Access Protocol (LDAP) and secured integration in a multiplatform environment that uses z/VSE connectors.

z/VSE supports the System z cryptographic solutions. They are used within the connection security and within the tape encryption.

This chapter includes the following topics:

- ▶ 1.1, “Introducing the z/VSE parts” on page 2
- ▶ 1.2, “z/VSE security features” on page 5

---

<sup>1</sup> TCP/IP information in this book is based on TCP/IP for VSE/ESA distributed by IBM, program number 5686-A04.

## 1.1 Introducing the z/VSE parts

z/VSE is designed to use selected features of IBM Z hardware. Many products and functions refer to the term *VSE* or continue to use *VSE/ESA* in their names. We use the term *z/VSE* to refer to the latest version of VSE.

### 1.1.1 Using z/VSE

Most of the z/VSE clients use this operating system for batch and online processing. *Batch processing* means that you start a job that consists of one or more tasks and runs in the background without further interaction necessary. When it finishes, you receive the result (which can be, for example, jobs for payroll, print accounting lists, and backups of databases for a company).

By contrast, *online processing* means that you engage in a dialog with the machine. Thousands of other interactive users might be working with this particular machine at the same time as you; for example, employees of the courier company that is processing delivery orders and updating track and tracing information.

In principle, all online applications do the same thing: they start and run transactions. On z/VSE, these transactions are processed by the IBM CICS Transaction Server for VSE<sup>2</sup>. CICS is a component of z/VSE that is heavily used by the z/VSE clients.

z/VSE includes the following major features:

- ▶ z/VSE Central Functions
- ▶ TCP/IP
- ▶ ACF/IBM VTAM®
- ▶ CICS

The z/VSE Central Functions include basic system control functions and the following components:

- ▶ VSE/IBM POWER® Priority Output Writers, Execution Processors, and Input Readers
- ▶ Interactive Computing and Control Facility (ICCF)
- ▶ Interactive Interface
- ▶ Connectors
- ▶ Data access methods

Next, we describe the following z/VSE Central Functions and parts:

- ▶ z/VSE Central Functions

This component is the base component of z/VSE. Central Functions provides basic system control for a z/VSE system through the supervisor. Basic system control includes storage management and input/output handling for the hardware that is attached to the machine. Central Functions also contains various programs and functions, such as the central security services and the job control language (JCL).

The process works as follows: A job or a job stream consists of JCL statements or commands that define a task to z/VSE and specify its requirements, such as loading a program (referred to as a *phase* in z/VSE), or assigning input/output devices to symbolic names. The z/VSE dialogs enable you to easily create the job streams for many required tasks, or to provide job stream skeletons for further completion.

---

<sup>2</sup> Normally, we use the term CICS for CICS Transaction Server. If necessary, we also use the term CICS TS to differentiate between CICS/VSE and CICS Transaction Server.

► VSE/POWER Priority Output Writers, Execution Processors, and Input Readers

This system is the spooling system of z/VSE, and it performs the following functions:

- Reads jobs from various input devices, including a Remote Job Entry (RJE) workstation, and stores them in the input queue.
- Starts jobs from the input queue in one of the partitions that it controls.
- Stores output from various jobs in one of the output queues (LIST, PUNCH, or XMIT) or on tape and, if required, controls the writing of it on a printer.
- On request, it transfers spooled output to a subsystem in another partition. The subsystem then can print, display, or punch this output.
- Maintains a transmit queue for jobs or output to be transmitted to another node.

To manage the batch queues and the VSE/POWER networking facilities, you can use VSE/POWER commands or dialogs. VSE/POWER controls the access to the queue files.

► z/VSE Interactive Computing and Control Facility (ICCF)

ICCF is the interactive tool for system administration and for program development. You enter source code and data, edit this information, and save it in an ICCF library. You can also create jobs and submit them for processing in a batch partition, or in an ICCF interactive partition. ICCF supports system control and functions, such as dialog processing.

► z/VSE Interactive Interface

This interface is an extension of the ICCF dialog functions for system administration. It also is an implementation of a CICS Transaction Server (CICS TS) application. The Interactive Interface's dialogs also include ICCF dialogs. It is the central place to define user IDs in z/VSE and protect CICS resources, such as CICS files.

► z/VSE connectors

To enhance interoperability of z/VSE with other systems, new functions that are based on client/server technology were implemented in recent releases of z/VSE. They are known as the z/VSE connectors.

Currently, z/VSE provides the following connectors:

- VSAM redirector connector
- z/VSE script connector
- IBM DB2®-based connector
- LDAP connector
- Java-based connector

The Java-based connector enables you to integrate your z/VSE system into an e-business world. You can have real-time access to z/VSE resources from remote platforms. The z/VSE connector consists of the following parts:

– z/VSE Connector Client

This client runs on a middle tier between the user (for example, a web browser) and z/VSE. It provides a z/VSE JavaBeans class library, online documentation, and programming reference. It also includes many samples, including Java source code for writing web applications, such as applets, servlets, and Enterprise JavaBeans (EJBs). The z/VSE Connector Client is part of the z/VSE connector component, and is available for download from the z/VSE home page.

- z/VSE Connector Server

The server runs on z/VSE and implements native access methods to z/VSE data and VSE/POWER, which allows you to submit jobs and access the z/VSE operator console. The z/VSE Connector Server accepts secure connections from the z/VSE Connector Client. This feature enhances the security between the middle tier and z/VSE.

z/VSE can be connected to other platforms through CICS Web Support (CWS). CWS allows a user to have direct access to CICS TS applications from a web browser without the use of a middle tier.

z/VSE can also act as a client, which means that you can access data, such as databases or flat files on other systems from your z/VSE programs.

## 1.1.2 How z/VSE stores data

z/VSE stores data in the following files, z/VSE libraries, and ICCF libraries:

- ▶ z/VSE files:

- Basic Access Method (BAM) files

When you define a z/VSE BAM file, you must know exactly where and on which disk your file should be allocated. You store this information in a label area to make it available when you work with this file later on.

- Virtual Storage Access Method (VSAM) files

First, you must define a VSAM space with a detailed location specification and store it in a so-called VSAM catalog. Then, you can define one or more VSAM files in this catalog. VSAM allocates storage for these files in the VSAM space and provides the access methods. It also ensures input/output processing and enables programmers to more easily use VSAM files rather than BAM files.

- ▶ z/VSE libraries

z/VSE libraries can be defined in a VSAM space or as a BAM file on disk, but they have their own access method to support the following tree structure:

- The library at the top level.
- Sublibraries at the next level.
- Members at the bottom level, which contains data, such as programs, procedures, and text.

- ▶ ICCF libraries

ICCF uses a different concept. The ICCF data and the ICCF user IDs are stored in one file, called the DTSFILE. The DTSFILE is allocated similar to a BAM file. It contains the ICCF libraries, which are identified by numbers, not by names.

The ICCF libraries consist of members. The contents of a member can be any text, a job skeleton to administer your system, source code of a program to be compiled, or another batch job to catalog a member in a z/VSE sublibrary.

For all these ways to store data, z/VSE provides functions to ensure that this data is secure.

## 1.2 z/VSE security features

The security features of today's version of z/VSE are a result of technical innovations and client requirements. Depending on a particular company's security policy, only a subset of the features might be necessary. The current version of z/VSE includes the following features:

- ▶ Online security
- ▶ Batch security
- ▶ Basic Security Manager (BSM)
- ▶ Single sign-on support with LDAP
- ▶ System z cryptography
- ▶ CICS Web Support (CWS)
- ▶ Connector security
- ▶ TCP/IP security
- ▶ Security for FTP
- ▶ OpenSSL
- ▶ Intrusion detection
- ▶ Policy compliance

These features are described next.

### 1.2.1 Online security

Historically, z/VSE was heavily used for online processing. Therefore, the central subsystem on z/VSE is a transaction management system. The most widely used transaction management system on z/VSE is Customer Information Control System (CICS). It is a general-purpose online transaction processing (OLTP) server for mission-critical applications. CICS manages the sharing of resources, the integrity of data, and prioritization of execution, with fast response. CICS authorizes users, allocates resources (real storage and cycles), and passes on database requests by the application to the appropriate database manager (such as DB2)<sup>3</sup>. CICS acts similar to and performs many of the same functions as, the z/VSE operating system. Also, as with the z/VSE operating system, CICS uses a security manager (such as BSM) to make its security decisions.

The online security might be sufficient if you use only online processing. When the computers become more connected over a widespread network, such as an intranet or the internet, protecting the new paths to enter the system and all accessible resources is a requirement. z/VSE achieves this requirement by using batch security.

### 1.2.2 Batch security

Today, batch jobs can be sent to z/VSE systems in many ways; for example, through NJE from a z/VM system, through TCP/IP FTP, or through z/VSE connectors. If a job is running on a z/VSE system, it can access all unprotected resources, including data files and system libraries. To control access to these resources, you must apply batch security.

Batch security performs the following functions:

- ▶ It allows the assignment of a user and authorizations to a batch job. To do so, you specify the user ID and password at the VSE/POWER JOB statement or the JCL ID statement. If you are an online user and submit a batch job, the new job inherits the user ID and authorization from your online session.

---

<sup>3</sup> In this publication, we do not describe all CICS security functions. For more information about such functions, see *CICS Transaction Server for VSE/ESA Security Guide*, SC33-1942.

- ▶ It provides access control to z/VSE files, z/VSE libraries, sublibraries, and members when they are accessed from batch jobs.

### 1.2.3 Basic Security Manager

The Basic Security Manager (BSM) provides basic security support that includes user verification, resource authorization, and logging capabilities. It can be customized after the initial installation of z/VSE.

BSM controls access to and protects resources in an online (CICS) environment and in a batch environment. For a software access control mechanism to work effectively, it must first identify the person who is trying to gain access to the system, and then verify that the user is really that person.

Even with BSM, you are responsible for protecting the system resources, such as CICS resources, z/VSE files, and z/VSE libraries that can be accessed out of CICS or from a batch job. You are also responsible for issuing the authorities by which those resources are made available to users. The BSM stores your security assignments as profiles in its repositories. BSM then refers to the information in the profiles to decide whether a user is permitted to access a system resource.

The ability to log information, such as attempted accesses to a resource, and to generate reports that contain that information can prove useful to a resource owner. It also is important to a smoothly functioning security system.

Because BSM can identify and verify a user ID and recognize which resources the user can access, BSM can record the events where user-resource interaction were attempted. This function records actual access activities or variances from the expected use of the system.

BSM includes several logging and reporting functions that allow a resource owner to identify users who attempt to access the resource. In addition, you or your auditor can use these functions to log all detected successful and unsuccessful attempts to access the BSM repositories and BSM-protected resources. Logging all access attempts allows you to detect possible security exposures or threats. The following logging and reporting functions are available:

- ▶ Logging

BSM writes audit records in a file for detected, unauthorized attempts to enter the system. Optionally, BSM can also write records for authorized attempts or detected, unauthorized attempts to:

- Access BSM-protected resources<sup>4</sup>
- Issue BSM commands
- Modify profiles on the BSM repositories

BSM uses the Data Management Facility (DMF) of CICS to write its audit records.

- ▶ Sending messages

BSM can send messages to the console for detected, unauthorized attempts to enter the system and for detected, unauthorized attempts to access BSM-protected resources.

For more information about BSM, see Chapter 2, “z/VSE Basic Security Manager” on page 11.

---

<sup>4</sup> For z/VSE files and libraries that are defined in the BSM repository DTSECTAB, you can also use the Access Control Logging and Reporting (ACLR) licensed program.

## 1.2.4 Single sign-on and LDAP

Today, people and businesses rely on networked computer systems to support distributed applications. These distributed applications might interact with computers on the same local area network, within a corporate intranet, within an extranet that links partners and suppliers, or anywhere on the internet.

To improve functionality and ease-of-use and to enable cost-effective administration of distributed applications, information about the services, resources, users, and other objects accessible from the applications must be organized in a clear and consistent manner. Much of this information can be shared among many applications, but it must also be protected to prevent unauthorized modification or the disclosure of private information.

Information that describes the various users, applications, files, printers, and other resources that are accessible from a network is often collected into a special database that is sometimes called a directory. As the number of networks and applications grows, the number of specialized directories of information also grows, which results in islands of information that are difficult to share and manage. If all of this information could be maintained and accessed in a consistent and controlled manner, it would provide a focal point for integrating a distributed environment into a consistent and seamless system.

The Lightweight Directory Access Protocol (LDAP) is an open industry standard that evolved to meet these needs. LDAP defines a standard method for accessing and updating information in a directory. LDAP gained wide acceptance as the directory access method of the internet and is also becoming strategic within corporate intranets.

It is being supported by a growing number of software vendors and incorporated into a growing number of applications. For example, application middle ware, such as the IBM WebSphere® Application Server or the IBM HTTP server, Linux APACHE Webserver, and support LDAP functionality as a base feature.

As one part of a client environment, z/VSE supports the single sign-on. For more information, see Chapter 3, “LDAP sign-on support” on page 65.

## 1.2.5 System z cryptographic solution

System z is well-equipped to address modern cryptographic security needs. Since early in the 1990s, the hardware included a cryptographic processor. All the System z operating systems (IBM z/OS® system, Linux on System z, z/VM, and z/VSE) provide software to implement the necessary security-related solutions. Today, the mainframe offers everything that you need for an effective cryptographic solution in your environment.

The z/VSE operating system supports cryptographic processing to support Secure Sockets Layer (SSL) for its TCP/IP-based processes. If cryptographic accelerator hardware is available, z/VSE can use it automatically, including CPACF, and Crypto Express cards up to the Crypto Express 6S. If no cryptographic hardware is available, z/VSE performs the necessary cryptographic functions in software, as far as possible.

Encryption Facility (EF) for z/VSE gives you another encryption option and complements z/VSE support for the encrypting tape drives IBM TS1120 and TS1130. TS1120 and TS1130 are the preferred solution for high-volume backup and archive applications. EF for z/VSE is designed to allow secure exchange of encrypted data with other locations within your company, and with Business Partners, suppliers, and clients.

## 1.2.6 CICS Web Support

CICS Web Support (CWS) is a set of services that enables direct connection from a web browser to CICS Transaction Server for VSE/ESA. CWS enables access from internet to CICS application programs and transactions. This access requires special attention to security aspects.

The following key security components are available:

- ▶ Secure Sockets Layer (SSL)
- ▶ Server authentication
- ▶ Client authentication

For more information about CWS, see Chapter 6, “CICS Web Support security” on page 221.

## 1.2.7 Connector security

The z/VSE e-business connectors support the Application Framework for e-business, and provide you with the resources to extend your *core applications* to *e-business applications*. By doing so, you can protect and use existing core-application investments.

Core applications, mainly CICS, COBOL, VSAM programs, run on the z/VSE host, are critical to the company’s operations. They are expected to remain in production for many years to come and often represent an enormous investment of past resources.

E-business applications are typically based on common standards, such as TCP/IP, HTML, XML, Secure Sockets Layer (SSL), and Secure Electronic Transaction (SET). They include the server and client code that is written in Java, access relational data locally or remotely, and interface with users through a standard web browser.

For more information about z/VSE connectors, see Chapter 7, “Connector security” on page 251.

## 1.2.8 TCP/IP security

TCP/IP for VSE/ESA (the native TCP/IP solution for z/VSE) is preinstalled in the z/VSE (or VSE/ESA) base. It is available as an optional-priced IBM program with IBM product number 5686-A04, and can be activated by an activation key. IBM licensed this program from Connectivity Systems Incorporated.

TCP/IP opened z/VSE communication paths to virtually every other computer platform. With this new openness, new security concerns might arise.

Although TCP/IP for VSE/ESA is not a security product, it does provide numerous functions that secure system resources.

For more information about TCP/IP and security, see Chapter 8, “TCP/IP security” on page 283.

## 1.2.9 Secure FTP

If you transfer files over the internet, you normally use the File Transfer Protocol (FTP). The only security that is provided is the optional use of a user ID and a password during the initialization of the FTP connection.



With the Secure FTP for VSE feature of TCP/IP for VSE/ESA, FTP on z/VSE supports user authentication, confidentiality, and data integrity by using digitally signed certificates, data encryption, and secure hash functions.

For more information about FTP and security, see Chapter 10, “Secure File Transfer Protocol” on page 319.

### **1.2.10 Intrusion detection**

One element of z/VSE intrusion detection capabilities is that if a sign-on is denied, the event is tracked. When a maximum of sign-on failures is reached, sign-on to the user ID is disabled, and the terminal session is ended.

Logging is supported on z/VSE. Logon attempts and access attempts to z/VSE resources can be detected and recorded. By using the recorded information, you can identify attempts to log on with a user ID by using invalid passwords or unauthorized access attempts to z/VSE resources.

### **1.2.11 Compliance to policy**

A z/VSE system is secured by using security features of the System z hardware, maintaining compliance to security policy within operating practices, and by using the functions of the installed security manager.

The Basic Security Manager provides basic security support and is part of z/VSE.

As an alternative to the use of the BSM, z/VSE allows you to use an external security manager (ESM). This feature often is a priced product from an independent software vendor (ISV) and must be installed separately.

An ESM is for users who require more functionality and flexibility in their security implementation than the BSM can provide. For example, the BSM does not support CICS field-level security and CICS command security. If you need these functions, an ESM might be a solution.





## z/VSE Basic Security Manager

The Basic Security Manager (BSM) was introduced with VSE/ESA V2R4 in response to the security requirements of CICS Transaction Server (CICS TS), which replaced the old CICS/VSE.

Currently, the BSM supports the following central areas:

- ▶ Online (signon) security
- ▶ Batch security
- ▶ CICS transaction security
- ▶ CICS resource security

The BSM of z/VSE V6R2 contains all functions of the previous BSM releases with improved security support. The following security support was added since initial introduction:

- ▶ Protecting WebSphere MQ for z/VSE V3 resources using IBM MQ resource classes
- ▶ SMF logging and reporting for DTSECTAB resources
- ▶ Protecting selected operands of JCL statements
- ▶ Easier administration of users and BSM groups using the enhanced dialogs
- ▶ Easier administration of DTSECTAB resources with dialog support
- ▶ Help on BSM profile administration by way of BSM Cross Reference reports

This chapter includes the following topics:

- ▶ 2.1, “BSM concept” on page 12
- ▶ 2.2, “Installing and customizing BSM” on page 16
- ▶ 2.3, “BSM administration” on page 17
- ▶ 2.4, “BSM auditing” on page 50
- ▶ 2.5, “BSM backups” on page 60

## 2.1 BSM concept

The BSM was built to provide basic security functions. If a client needs more functionality, they can use an external security manager (ESM) from an independent software vendor (ISV) instead of the BSM.

BSM and ESM must support the RACROUTE requests of the System Authorization Facility, which is described next.

### 2.1.1 System Authorization Facility

CICS TS was introduced with VSE/ESA V2R4. This CICS no longer contains internal security functions. Instead, it issues RACROUTE calls to a security manager.

RACROUTE is the macro interface to the centralized security component, the System Authorization Facility (SAF) router. When started, the SAF router first calls an optional installation exit routine and then, the BSM or an external security manager from an independent software vendor (ISV), if one is installed and active on the system.

Figure 2-1 shows the processing of a RACROUTE request, when issued from a resource manager. Examples of such resource managers include CICS TS, TCP/IP with security exit BSSTISX, DITTO, and the connectors of z/VSE. The SAF and the security manager are running in the partition of the resource manager.

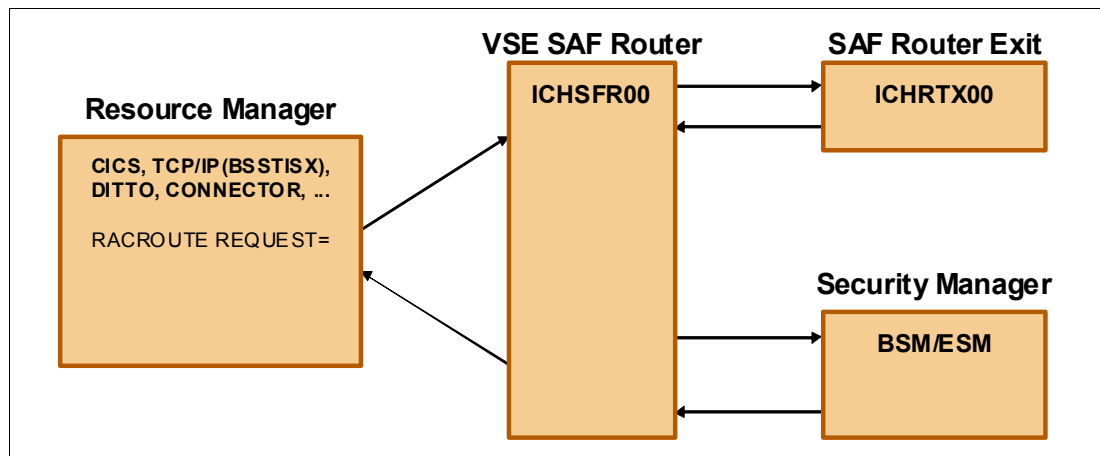


Figure 2-1 Processing a RACROUTE request

The SAF that is provided with z/VSE was ported from OS/390. Although z/VSE and OS/390 are based on the same IBM S/390® platform, differences exist. For more information, see *z/VSE Planning*, SC34-2681.

## 2.1.2 Security files

z/VSE includes several files that contain security information; for example, crypto key files, mapping files for certificates or LDAP user IDs, and files with user profiles and resource profiles.

Figure 2-2 shows the relation of the online and batch security to those files that are handled by BSM and ICCF.

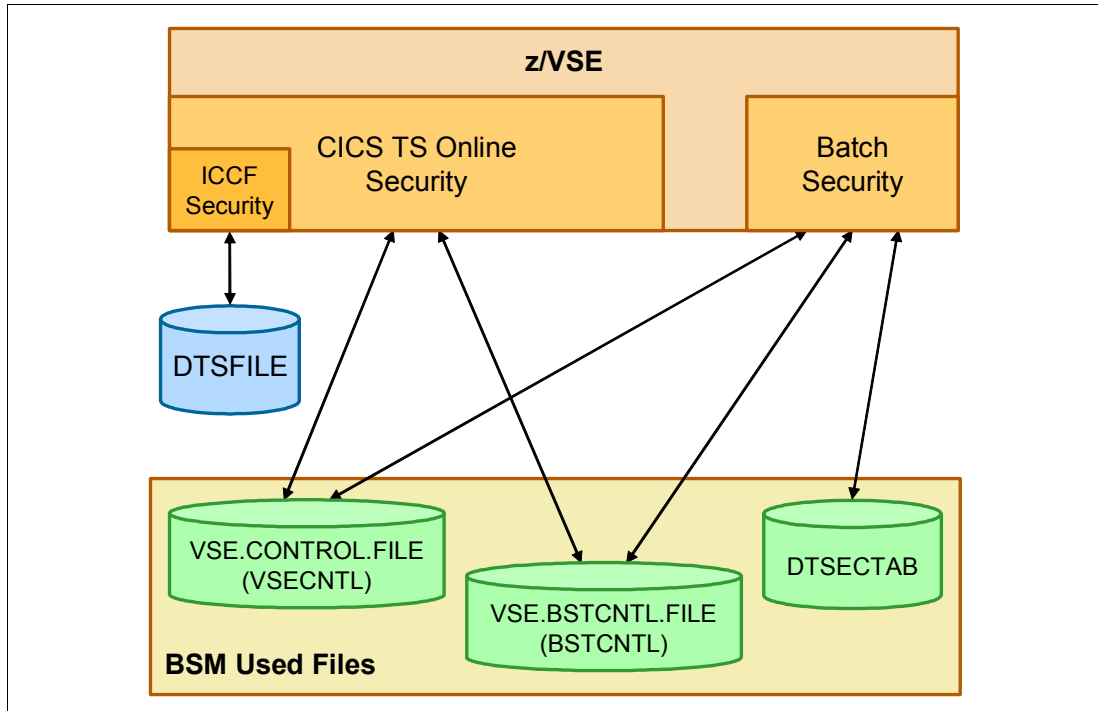


Figure 2-2 Online and batch security that is related to security files of BSM and ICCF

The BSM uses the following files to store its security information about user profiles and resource profiles:

- ▶ VSE control file VSE.CONTROL.FILE (VSECNTL)

This file existed when VSE/ESA V2R4 introduced the BSM. With the BSM, it became the central repository for the user IDs (user profiles).

- ▶ BSM control file VSE.BSTCNTL.FILE (BSTCNTL)

With z/VSE V3R1M1, the BSM was enhanced to support more types of resources, which are called *resource classes*. The profiles of these resource classes are stored in a new KSDS VSAM file, which is the BSM control file.

The resources that can be defined in the BSM control file are listed in Table 2-1 on page 14. Most of these resources are CICS-owned resources.

Table 2-1 Resources defined in BSM control file

Resource	Resource Class	Parameter in DFHSIT <sup>a</sup>
CICS transactions	TCICSTRN	XTRAN= YES
CICS application programs	MCICSPPT	XPPT=NO
CICS files	FCICSFCT	XFCT=NO
CICS journals	JCICSJCT	XJCT=NO
CICS temporary storage queues	SCICSTST	XTST=NO
CICS transient data queues	DCICISDCT	XDCT=NO
CICS started transactions and some EXEC CICS commands	ACICSPCT	XPCT=NO
Applications like VTAM applications	APPL	-
Miscellaneous resources	FACILITY	-
WebSphere MQ administration-type functions	MQADMIN	-
WebSphere MQ commands	MQCMDS	-
WebSphere MQ connections	MQCONN	-
WebSphere MQ namelists	MQNLIST	-
WebSphere MQ queues	MQQUEUE	-
WebSphere MQ mixed case topic	MXTOPIC	-
Surrogate user IDs	SURROGAT	XUSER=NO

a. The parameters show the default setting.

To protect a CICS-owned resource, complete the following steps:

- a. Define the resource through a profile of the related resource class in BSM.
- b. Specify SEC=YES in your DFHSIT to activate security in CICS.
- c. Set the appropriate parameter for the resource class in your DFHSIT to YES. For example, set XFCT=YES to protect the files that are defined in BSM resource class FCICSFCT.
- d. Define in your transaction definition whether resource security checking is to be used for the resources that are accessed by your transaction.

**Note:** Within your transaction definition, RESSEC=NO indicates that all resources are available to any user who can use this transaction. RESSEC=YES indicates that a security manager, such as BSM, is used for resource security checking.

If you activated the security checking for a resource class in CICS, *all* resources of this resource class *must* be defined to BSM. You can use one profile for each resource of this class or generic profiles for several resources. For more information about BSM profiles, see *z/VSE Administration*, SC34-2692.

For more information about CICS security, including BSM examples, see *CICS Transaction Server for VSE/ESA Security Guide*, SC33-1942. For information about WebSphere MQ security, see *WebSphere MQ for z/VSE System Management Guide*, GC34-6981.

IBM provides resource profiles in the BSM control file for CICS transactions<sup>1</sup> and other resources that are part of z/VSE. In addition to the resource profiles, the BSM control file contains the system-wide security settings.

► DTSECTAB table

As with the VSE control file, the DTSECTAB table and its checking logic were established long before the BSM was introduced. It is used for the security definitions of z/VSE files, libraries, sublibraries, and members. The name of the table is the same as the name of the macro that is used to define security entries in the table.

The ICCF DTSECTAB is a file that contains user ID information and resource information. The types of information are restricted to ICCF only. It is not used by BSM but ICCF uses user information from BSM.

### 2.1.3 Security server partition

The security server was established to control the access to the VSE control file and the BSM control file when RACROUTE requests are processed by the BSM. Today, it also contains functions to support the cryptography features of System z and the BSM logging with DMF.

The security server runs, by default, in the FB partition and is always active. It includes the job name SECSERV. Because of DMF, this server requires an environment that is similar to z/OS with specific control blocks. Therefore, the option OS390 is specified in this job.

The security server provides a set of commands to control its operation and display server status information. These commands can be entered from the system console as shown in the following example:

```
MSG xx,DATA=command
```

Where xx indicates the selected server partition (default FB). To obtain a list of possible commands, enter any of the following commands:

- HELP
- ?
- blank

For more information, see *z/VSE Operation*, SC33-8309 and *z/VSE Administration*, SC34-2692.

### 2.1.4 BSM processing

Resource managers, such as CICS, TCP/IP, or connectors, issue RACROUTE requests. BSM receives those requests from SAF. Depending on the request, the BSM searches in its security files for the required information and returns the result. The resource manager uses this result, for example, to decide whether a user can access a specific resource.

For compatibility reasons, the authorization checks of DTSECTAB resources with the old SECHECK macro interface are still available and used.

<sup>1</sup> The first versions of BSM stored CICS transaction profiles in the DTSECTXN table. This table was replaced by the BSM control file in z/VSE V3R1M1. For more information about DTSECTXN and migration, see *z/VSE Administration*, SC34-2692.

## 2.1.5 Common startup for BSM and ESM

During IPL, z/VSE gets the information about whether the BSM or an ESM should be started. If you want to use an ESM, you must specify the name of the ESM initialization routine in the IPL SYS command, as shown in the following example:

```
SYS ESM=CAKSESM
```

During BGINIT processing, the common security initialization routine BSSINIT starts the ESM initialization routine. If no ESM initialization routine is specified or the specified routine does not exist, BSM initialization is started.

The BSM initialization routine starts the security server partition and waits until the security server partition is ready for work. If an ESM requires a security server partition, BSSINIT starts the security server partition and waits until the ESM is ready on the security server partition.

The security server is started in partition FB by default. If you want to use a different partition, use the following IPL command:

```
SYS SERVPART=
```

If you use batch security (SYS SEC=YES) with an ESM, the DTSECTAB processing is active until the ESM is ready for work.

## 2.2 Installing and customizing BSM

BSM is part of z/VSE. It is ready for customization after initial installation.

The BSM is always activated during start, independent of setting SEC=YES or SEC=NO in the IPL SYS command. This configuration is used to provide online security; that is, sign-on security (signing on through the Interactive Interface or the CICS Transaction Server), and CICS transaction security.

**Note:** For this support, the DFHSIT must include SEC=YES, which is the default. If you reset it to SEC=NO, no CICS security can be used, including sign-on security.

During initial installation, you are asked whether you want to run your system with *security on*. If you respond with YES (as suggested), the IPL SYS command is set with SEC=(YES,NOTAPE), which, in addition to online security, provides batch security and access control for resources defined in DTSECTAB. This DTSECTAB is the default DTSECTAB, which contains entries for the resources that are provided by IBM. If you want to customize the DTSECTAB later, use the member DTSECTRC in ICCF library 59.

Other security settings can be changed by using BSTADMIN commands, which is described next.



## 2.3 BSM administration

To administer security, you must review the entire system. Focusing on one part of the system while ignoring other parts does not help.

Another consideration is the cost of security. Security is not free. At the very least, it requires system resources. You must find the right balance between the value of what you protect and the degree of protection with its costs. Remember this fact when you start to administer all the components of your security system.

The BSM administration includes the following components:

- ▶ Security system settings
- ▶ User definition
- ▶ Group definition
- ▶ Resource profile definition

These components are described next.

For more information about the logging and reporting function, see 2.4, “BSM auditing” on page 50.

### 2.3.1 Security system settings

The security system settings are seldom changed. Changes might be required for the following reasons:

- ▶ As a result of a security audit
- ▶ When the security policies are changed
- ▶ When resources of an inactive resource class must be protected
- ▶ When the size of the security data space must be adjusted

We suggest that you control the status of your security system before you start changing your security settings.

#### Getting the status of the security system

With the console command SIR, you can see which security manager was selected and the status of this security manager.

Example 2-1 shows that the BSM was started, and that online security and batch security are active.

*Example 2-1 Security status from the SIR command*

---

```
sir
...
AR 0015 IPL-PROC = $IPLESA          JCL-PROC = $$JCL
AR 0015 SUPVR   = $$A$SUPI          TURBO-DISPATCHER (66) ACTIVE
AR 0015                                     HARDWARE COMPRESSION ENABLED
AR 0015 SEC. MGR. = BASIC          SECURITY = ONLINE and BATCH
AR 0015 VIRTCPU = 0000:06:19.623     CP = 0000:00:35.465
AR 0015 CPU-ADDR. = 0000(IPL)    ACTIVE
...
```

---

The BSM controls various settings of security options. With the BSM administration program BSTADMIN, you can change the security settings and the resource profiles that are defined in the BSM control file BSTCNL. The BSTADMIN command **STATUS** lists the current security settings.

Example 2-2 shows how to start BSTADMIN, how to issue the **STATUS** command, and the information you receive from the **STATUS** command.

*Example 2-2 BSTADMIN STATUS command output*

---

```

r rdr,pausebg
r rdr,pausebg
AR 0015 1C39I COMMAND PASSED TO VSE/POWER
F1 0001 1R88I OK : 1 ENTRY PROCESSED BY R RDR,PAUSEBG
BG 0001 1Q47I BG PAUSEBG 02675 FROM (SYSA) , TIME=10:10:14 , TKN=00000019
BG 0000 // JOB PAUSEBG
          DATE 03/07/2018, CLOCK 10/10/14
BG-0000 // PAUSE
BG 0000 // ID (PARAMETERS SUPPRESSED)
BG-0000
0 exec bstadmin
BG 0000 1S54I PHASE BSTADMIN IS TO BE FETCHED FROM IJSYSRS.SYSLIB
BG-0000 BST901A ENTER COMMAND OR END
0 st
BG 0000 CLASS      ACTIVE      CMDAUDIT
BG 0000 -----      -----      -----
BG 0000 USER        YES          NO
BG 0000 GROUP       YES          YES
BG 0000 DATASET     YES          NO
BG 0000 VSELIB      YES          NO
BG 0000 VESLIB      YES          NO
BG 0000 VSEMEM      YES          NO
BG 0000 TCICSTRN   YES          YES
BG 0000 ACICSPCT   YES          YES
BG 0000 DCICSDCT   YES          YES
BG 0000 FCICSFCT   YES          YES
BG 0000 JCICSJCT   YES          YES
BG 0000 MCICSPPT   YES          YES
BG 0000 SCICSTST   YES          YES
BG 0000 APPL        YES          YES
BG 0000 FACILITY    YES          YES
BG 0000 MQADMIN     YES          YES
BG 0000 MQCMDS      YES          YES
BG 0000 MQCONN      YES          YES
BG 0000 MQQLIST     YES          YES
BG 0000 MQQUEUE     YES          YES
BG 0000 SURROGAT   YES          YES
BG 0000 MXTOPIC     YES          YES
BG 0000
BG 0000 PASSWORD PROCESSING OPTIONS:
BG 0000 12 GENERATIONS OF PREVIOUS PASSWORDS BEING MAINTAINED.
BG 0000 AFTER 5 CONSECUTIVE UNSUCCESSFUL PASSWORD ATTEMPTS,
BG 0000 A USERID WILL BE REVOKED.
BG 0000 PASSWORD EXPIRATION WARNING LEVEL IS 6 DAYS.
BG 0000 A PASSWORD CAN HAVE 8 TO 8 CHARACTERS.
BG 0000

```

```

BG 0000 AUDIT OPTIONS:
BG 0000 ADMINISTRATOR ACCESSES TO RESOURCES ARE LOGGED
BG 0000
BG 0000 GENERAL OPTIONS:
BG 0000 A USER ID IS REQUIRED TO USE BSTADMIN WITHOUT BATCH SECURITY
BG 0000 BSTADMIN IS USING USER ID SYSA FOR AUTHORIZATION
BG 0000
BG 0000 LOGGING STATUS:
BG 0000 SMF LOGGER FOR DTSECTAB RESOURCES IS INSTALLED AND ACTIVE
BG 0000 DMF STATUS IS: ACTIVE
BG 0000
BG 0000 DATA SPACE STATUS:
BG 0000 CURRENT DATA SPACE SIZE IS 960K.
BG 0000 USAGE OF DATA SPACE STORAGE IS 15%.
BG 0000 DATA PART SIZE IS 150K.
BG 0000 SIZE OF PREVIOUS DATA SPACE WAS 960K.
BG 0000 USAGE OF PREVIOUS DATA SPACE WAS 15%.
BG 0000 DATA PART SIZE WAS 150K.
BG 0000
BG 0000 BST904I RETURN CODE OF STATUS IS 00
BG-0000 BST901A ENTER COMMAND OR END

```

---

After the job PAUSEBG was released, an ID statement with user ID follows. For security reasons, the user ID and password are suppressed. Only administrator user IDs can be used for BSTADMIN.

The **exec bstadmin** command starts the BSM administration service. The short form, **st**, of the BSTADMIN command **STATUS** was used to obtain this status list.

The output that is shown in Example 2-2 on page 18 is described next.

The columns CLASS and ACTIVE show all the resource classes that are supported by BSM and their status. If a resource class is active, security checks for resources of this class are performed. In column CMDAUDIT, you see for which resource class the changes to resource profiles should be logged. For user IDs and DTSECTAB resources, command audit is not available. For more information about command audit, see 2.4, “BSM auditing” on page 50.

The PASSWORD PROCESSING OPTIONS shows the following information:

- ▶ A password history of 12 passwords is used.
- ▶ If a user attempts to sign on five times with the wrong password, the user ID is revoked.
- ▶ Interactive Interface starts informing that your password expires six days before it expires.
- ▶ A new password must have a minimum length of eight characters.

The AUDIT OPTIONS indicate that all resources that are accessed by administrators are logged. An administrator can access all resources by default and normally does not create a log entry. With this option, the accesses of administrators are also logged.

The GENERAL OPTIONS show that in an environment without batch security active, you must enter a **USERID** command before you can enter any other **BSTADMIN** command. In this example, batch security is active. The current user ID is SYSA. This user ID was specified in the ID statement before BSTADMIN was called. Later on, you find this user ID in the command report of the BSM report writer, when the log records of this BSTADMIN session are listed.

The LOGGING STATUS shows whether the logger program that creates SMF records for DTSECTAB resources is installed and active. In addition, it shows the status of the DMF program, which collects the SMF records.

The BSM uses data spaces to reduce the I/O to the BSM control file. The DATA SPACE STATUS provides statistics of the data space usage.

After you control the status of your security settings, you can start changing them, as described in the following sections.

## Changing the status of resource classes

Use the BSTADMIN command **PERFORM (pf)** with keyword CLASS to change the status of resource classes that are defined in the BSM control file (BSTCNTL).

The following options are available:

ACTIVE	Activates authorization checking for the resources of this class.
INACTIVE	Stops all authorization checking for the resources of this class.
CMDAUDIT	Activates logging of changes to resource profiles of this class. Activate command auditing if you use BSM auditing.
NOCMDAUDIT	Stops logging of changes to resource profiles of this class.

Example 2-3 shows how to activate security checking for CICS temporary storage queues with logging of profile modifications.

*Example 2-3 Change class settings*

---

```
0 perform class(scicstst) active cmdaudit
```

---

**Note:** If you do not want to enter so much information, you can use the short form of the commands and the abbreviations of the keywords and options only if they are unique.

Example 2-4 shows the shortest form of the same command that is shown in Example 2-3.

*Example 2-4 Short form of a command to change the class settings*

---

```
0 pf c(scicstst) a c
```

---

The **PERFORM** commands with keyword CLASS are always logged if DMF is active. For more information about logging and reporting, see 2.4, “BSM auditing” on page 50.

## Change the password processing options

The password processing options specify actions for the monitoring and checking of passwords. The keyword PASSWORD can be preceded by the following keywords:

HISTORY	Specifies that for each user ID, the BSM should save the previous passwords and compare the passwords in the list with an intended new password. If one of these previous passwords or the current password matches, the BSM rejects the intended new password. Currently, a history of 12 previous passwords is saved.
NOHISTORY	Specifies that the new password information should be compared with the current password only.
LENGTH	Specifies the minimum password length when a user changes password. A password includes a minimum length value of 3 - 8 characters.

REVOKE	Specifies the number (1 - 254) of consecutive incorrect password attempts that the BSM allows before it revokes the user ID on the next incorrect attempt. The default value is 5.
NOREVOKE	Specifies that the BSM should ignore the number of consecutive invalid password attempts and should not revoke the user ID.
WARNING	Specifies the number of days (1 - 255) before a password expires that the sign-on program should issue a warning message to a user. The default is 7 days.  The sign-on program of the z/VSE Interactive Interface supports only an internal maximum value of 7 days.
NOWARNING	Specifies that the sign-on program should not issue a warning message for password expiration.

Example 2-5 shows how the password processing options can be changed with the **PERFORM** command.

*Example 2-5 Change password processing options*

---

```
0 pf password history length(8) revoke(5) warning(7)
```

---

**Note:** Ensure that you removed the // EXEC IESIRCVT statement in the procedure USERBG.PROC. To do so, use skeleton SKUSERBG (found in ICCF library 59). If this statement is contained in USERBG.PROC, the IESIRCVT settings overwrite the settings that are specified by the **PERFORM PASSWORD** command.

## Starting and stopping logging for administrator accesses

A system administrator (user type 1) can access all resources that are independent of the definition in the resource profiles. For resource profiles that are stored in the BSM control file, you can specify that the *administrator accesses* to those resources should be logged.

Use the option ADMINACC of the keyword AUDIT to start the logging of administrator accesses. Example 2-6 shows how you can stop logging of administrator resource accesses by using the option NOADMINACC.

*Example 2-6 Stop logging for administrator resource accesses*

---

```
0 pf audit noadminacc
```

---

**Note:** If you use logging for DTSECTAB resources, administrator accesses to DTSECTAB resources are always logged. This behavior cannot be changed with the **PERFORM** command.

## Setting option for the USERID command

It is not required to set option CMDUSERID if you use BSM online security and batch security. If batch security is active, you often specified a user ID before you call BSTADMIN to enter any command. The logging and reporting function of BSM show this user ID in the log records of the BSM commands.

If batch security is not active, no user ID is assigned to a batch job where BSTADMIN runs. Use the **PERFORM** command and keyword SETOPT with option CMDUSERID if you want command auditing.

After option CMDUSERID is set, the next time BSTADMIN is started, it requires that the first BSTADMIN command is the **USERID** command. Other BSTADMIN commands are rejected. After the **USERID** command with USER and PASSWORD is issued, you can enter other BSTADMIN commands. The specified user ID is used for the logging of those BSTADMIN commands.

## Requesting a new data space size and refresh the data space contents

The security settings and most of the resource profiles are stored in the BSM control file. This information is heavily used during normal work. For performance reasons, the security settings are stored in control blocks and the profile information is stored in a data space.

The output of the **STATUS** command that is in Example 2-2 on page 18 shows the current size of the data space and how much of this data space is used for data and index. It also shows the data space size and data space usage as it was before the latest IPL was done.

If you plan to add many profiles or you want to reduce the amount of unused space, request a new size by using the **PERFORM** command with keyword DATASPACE and option SIZE. You can specify the new size in kilobytes (k) or in megabytes (m).

Example 2-7 shows a size request of 2 megabytes. The data space size is changed with the next IPL. If the size request is not processed (that is, no IPL was done), another message in the STATUS command output shows the requested data space size.

### *Example 2-7 Requesting a new data space size*

---

```
0 pf dataspace size(2m)
```

---

Example 2-8 shows the data space part of the STATUS command output with the information that a new data space size of 2048k was requested.

### *Example 2-8 Status output when a new data space size was requested*

---

```
BG 0000 DATA SPACE STATUS:
BG 0000 CURRENT DATA SPACE SIZE IS 960K.
BG 0000 USAGE OF DATA SPACE STORAGE IS 17%.
BG 0000 DATA PART SIZE IS 161K.
BG 0000 REQUESTED DATA SPACE SIZE FOR NEXT START IS 2048K.
BG 0000 SIZE OF PREVIOUS DATA SPACE WAS 960K.
BG 0000 USAGE OF PREVIOUS DATA SPACE WAS 17%.
BG 0000 DATA PART SIZE WAS 164K.
```

---

At the BSM initialization during the next IPL, two data spaces of the requested size are allocated. One is initialized with the profile information from the BSM control file. This data space is the active space.

Later on, when you add, delete, or change profiles, only the BSM control file is updated. To activate those modifications, use the **PERFORM** command with keyword DATASPACE and option REFRESH. This function copies the profile information to the inactive data space. If it was successfully copied, the data spaces are switched. The inactive data space becomes active and the old active space becomes inactive.

If the copy process failed because of data space problems, the system is still up and running that uses the old data space, but the changes are not active. However, you can perform a controlled shutdown and IPL your system to solve the data space problem. If the BSM was successfully initialized, the changes are active.

**Note:** CICS TS issues data space refresh requests internally during startup, as a result of CEMT PERFORM SECURITY REBUILD, or through Interactive Interface dialog fast path 283 BSM Security Rebuild.

## 2.3.2 Defining a User

For the user definition, we use the example that is described in this section.

You want to define a new programmer that is named Hugo Smith to z/VSE and BSM. He has the user ID HUGO with access to ICCF. To sign on as programmer to CICS and the Interactive Interface, Hugo must have access to all CICS transactions that are required for the programmer. The programmers are authorized for the transaction by way of groups. Therefore, he should be connected to the groups that are used by programmers.

Your company uses LDAP user IDs for sign-on to different systems by using the same LDAP user ID and password, as described in Chapter 3, “LDAP sign-on support” on page 65. The new programmer features a company-wide LDAP user ID. He also wants to use the LDAP user ID for z/VSE.

To define a new user ID to BSM, the following process is used:

1. Add the new user ID to the VSE control file.
2. Map this new VSE user ID to Hugo Smith's LDAP user ID.
3. Connect the new user ID to groups.
4. Activate your changes to the BSM control file.

With z/VSE V4R3, the dialog sequence to add a new z/VSE user ID includes the steps for LDAP and group definitions.

These steps are described next.

### Defining a new user ID

You often define a new user by using the Interactive Interface dialog. You sign on as an administrator and select the MAINTAIN USER PROFILES panel by using fast path 211.

Next, you add user Hugo as programmer, which is a type-2 user. Therefore, you can use the user ID PROG as a template, and specify option 1 (for ADD) in front of this user, as shown in Figure 2-3 on page 24.

```

IESADMUPL2                MAINTAIN USER PROFILES
VSE CONTROL FILE
START.... _____
OPTIONS:  1 = ADD                2 = CHANGE                5 = DELETE
          PASSWORD              REVOKE      USER INITIAL NAME
          VALID UNTIL          DATE        TYPE  NAME      TYPE
OPT  USERID                VALID UNTIL          DATE        TYPE  NAME      TYPE
-    $SRV                   01/01/97 *                2    IESERSUP  2
-    CICSUSER               01/01/97 *                3    DFLESEL   2
-    CNSL                   01/01/97 *                1    DUMMY     1
-    DBDCCICS                1    DUMMY     1
-    FORSEC                  1    IESEADM  2
-    OPER                   04/24/18                  2    IESEOPER  2
-    POST                    1    IESA$FST  1
-    PRODCICS                1    DUMMY     1
1    PROG                   04/24/18                  2    IESEPROG  2
-    SYSA                    1    IESEADM  2
-    VCSRV                   1    DUMMY     1

PF1=HELP                3=END
PF7=BACKWARD  8=FORWARD  9=PRINT

```

Figure 2-3 Adding a new user ID

After pressing Enter, the ADD OR CHANGE USER PROFILE panel opens. In this panel, specify for the new programmer the user ID as HUGO (hugo) and the initial password, as shown in Figure 2-4.

```

IESADMUPBA                ADD OR CHANGE USER PROFILE
Base   II      CICS      ResClass ICCF
To ADD, specify information for all of the entries.

USERID..... hugo_____ 4 - 8 characters (4 characters for ICCF users)
INITIAL PASSWORD... prog1new 3 - 8 characters
DAYS..... 090           0-365 Number of days before password expires
REVOKE DATE..... _____ Date when Userid will be revoked (mm/dd/yy)

USER TYPE..... 2        1=Administrator, 2=Programmer, 3=General
AUDITOR..... 2         1=yes, 2=no
INITIAL NAME..... IESEPROG Initial function performed at signon
NAME TYPE..... 2       1=Application, 2=Selection Panel
SYNONYM MODEL.....     Userid to be used as model for synonyms
PROGRAMMER NAME.... Hugo Smith      Supplementary user name

PF1=HELP                3=END                5=UPDATE
                        8=FORWARD

```

Figure 2-4 Add or change a user profile

The new password is expired and normally must be changed when user Hugo first signs on. Because this user ID is mapped to his LDAP user ID, (as shown in Figure 2-6 on page 26), the z/VSE password expiration can be ignored.



You can specify the name of this user in the field PROGRAMMER NAME. This information identifies the person that is assigned to that user ID, which is helpful if you must administer many user IDs. You also see this user name in the BSM security reports.

In the example, you might want to use the same definition for the Interactive Interface, CICS, and resource classes for the new user ID as it was defined in your model user ID PROG. Therefore, you do not step through all the panels by pressing F8. Instead, press F5 for an update, and directly receive the ICCF panel.

The user Hugo is a programmer who can access to the ICCF libraries. Figure 2-5 shows the ICCF panel. Library 9 is the primary ICCF library for the programmer.

```
ADM$XSN1          MAINTAIN USER PROFILES: SPECIFY LIBRARY
Enter the required data and press ENTER.

User Id = HUGO

LIBRARY..... 9          The primary ICCF library for this
                        user.

DEFAULTS..... 1          Do you accept the remaining defaults?
                        Enter 2 = no, 1 = yes.
                        (Do not change defaults without care-
                        ful consideration.)

PF2=REDISPLAY  3=END
```

Figure 2-5 Specify ICCF library

Press Enter to finish adding a new user to z/VSE.

With z/VSE V4R3, the dialog sequence to define a new user ID does not end here. Instead, you see the panel to maintain LDAP user profiles if your z/VSE system uses the LDAP sign-on support.

Specify option 1 for ADD (as shown in Figure 2-6) to start the mapping of user ID HUGO to an LDAP user ID.

```

IESADMLUPM          MAINTAIN LDAP USER PROFILES

START....
VSE USERID.... HUGO
OPTIONS:   1 = ADD

OPT  LDAP USERID                                USER
 1                                     TYPE

PF1=HELP                      3=END
                              9=PRINT          10=EXPORT
MAKE NECESSARY CHANGES TO VSE USER-ID 'HUGO  '.
```

Figure 2-6 Adding a new z/VSE user ID to your LDAP user profile

After pressing Enter, the ADD OR CHANGE USER PROFILE panel opens. Specify the LDAP user ID of user Hugo Smith and a short description. Have LDAP generate the z/VSE passwords for the z/VSE user ID HUGO, as shown in Figure 2-7.

```

IESADMLUPA          ADD OR CHANGE LDAP USER PROFILE

LDAP USERID.. hugos@de.ibm.com
DESCRIPTION.. Hugo Smith VSE programmer

VSE USERID..... HUGO      Assigned VSE user-ID. 1-8 characters
VSE PASSWORD.....         Specifies VSE password. 3-8 characters or blank
GENERATE PASSWORD.. 1      1 - Forces generation of random VSE password
                          2 - Use current password
PASSWORD PATTERN... aaaaaaaa Specifies a pattern for password generation
                          Required if password is generated
                          d - decimal digit (0-9)
                          c - character (A-Z)
                          a - decimal digit (0-9) or character (A-Z)
                          x - special character (@, # or $)
                          other - place is filled with specified character
                          blank - place is not filled with a character.

PF1=HELP                      3=END                      5=PROCESS
```

Figure 2-7 Specify the LDAP user information to assign the new z/VSE user ID

Press F5 to process your LDAP change.

Figure 2-8 shows the information that the change was successful.

```

IESADMLUPM          MAINTAIN LDAP USER PROFILES

START....
VSE USERID.... HUGO
OPTIONS:   1 = ADD

OPT  LDAP USERID          USER
-                                     TYPE

PF1=HELP          3=END          10=EXPORT
                  9=PRINT

USER PROFILE WAS ADDED SUCCESSFULLY.
  
```

Figure 2-8 Get status of LDAP change

Use F3 to close the LDAP dialog.

To sign on to CICS, the new user needs authorization to several transactions. The authorization is provided by way of groups. With z/VSE V4R3, the dialog sequence to define a new user ID includes the panels for the group definitions. It starts with the list of all groups to connect the new user ID. Groups to which the model user ID PROG is connected are marked with the character M in column USERID CONNECTED, as shown in Figure 2-9.

```

IESADMBSLG          MAINTAIN SECURITY PROFILES
BSM RESOURCE CLASS:  GROUP
START....
OPTIONS:   1 = ADD          2 = CHANGE          5 = DELETE          6 = USER LIST
           8 = CONNECT      9 = REMOVE          USERID
           OPT  GROUP NAME  DESCRIPTION          CONNECTED?
                                     HUGO
-          GROUP01         TRANSEC CLASS MIGRAT  M
-          GROUP02         TRANSEC CLASS MIGRAT
-          GROUP03         TRANSEC CLASS MIGRAT
-          GROUP04         TRANSEC CLASS MIGRAT
-          GROUP05         TRANSEC CLASS MIGRAT
-          GROUP06         TRANSEC CLASS MIGRAT
-          GROUP07         TRANSEC CLASS MIGRAT
-          GROUP08         TRANSEC CLASS MIGRAT
-          GROUP09         TRANSEC CLASS MIGRAT
-          GROUP10         TRANSEC CLASS MIGRAT
-          GROUP11         TRANSEC CLASS MIGRAT
-          GROUP12         TRANSEC CLASS MIGRAT

PF1=HELP          3=END          6=CONNECT MODEL
                  8=FORWARD      9=PRINT          10=REMOVE ALL
CHANGE THE SECURITY PROFILE OF USERID ACCORDING TO THE MODEL PROG .
  
```

Figure 2-9 List of groups with model information

Press F6 to connect user ID HUGO to the same groups to which the model user ID is connected.

The character M is replaced by an asterisk (\*), as shown in Figure 2-10. It indicates that the user ID HUGO is connected to this group.

```

IESADMBSLG                MAINTAIN SECURITY PROFILES
BSM RESOURCE CLASS:      GROUP
START....
OPTIONS:  1 = ADD          2 = CHANGE          5 = DELETE          6 = USER LIST
           8 = CONNECT     9 = REMOVE          USERID
           OPT  GROUP NAME  DESCRIPTION          CONNECTED?
           HUGO
           --  GROUP01     TRANSEC CLASS MIGRAT  *
           --  GROUP02     TRANSEC CLASS MIGRAT
           --  GROUP03     TRANSEC CLASS MIGRAT
           --  GROUP04     TRANSEC CLASS MIGRAT
           --  GROUP05     TRANSEC CLASS MIGRAT
           --  GROUP06     TRANSEC CLASS MIGRAT
           --  GROUP07     TRANSEC CLASS MIGRAT
           --  GROUP08     TRANSEC CLASS MIGRAT
           --  GROUP09     TRANSEC CLASS MIGRAT
           --  GROUP10     TRANSEC CLASS MIGRAT
           --  GROUP11     TRANSEC CLASS MIGRAT
           --  GROUP12     TRANSEC CLASS MIGRAT

PF1=HELP                3=END          6=CONNECT MODEL
                        8=FORWARD     9=PRINT      10=REMOVE ALL
CHANGE THE SECURITY PROFILE OF USERID ACCORDING TO THE MODEL PROG .

```

Figure 2-10 List of groups with connect information

Press F3 to close the MAINTAIN SECURITY PROFILES dialog and return to the MAINTAIN USER PROFILES panel, as shown in Figure 2-11.

```

IESADMUPL2                MAINTAIN USER PROFILES
VSE CONTROL FILE
START....
OPTIONS:  1 = ADD          2 = CHANGE          5 = DELETE
           PASSWORD      REVOKE      USER INITIAL  NAME
           OPT  USERID   VALID UNTIL  DATE  TYPE  NAME  TYPE
           --  HUGO      03/06/18 *          2  IESEPROG  2
           --  OPER      04/24/18          2  IESEOPER  2
           --  POST                                     1  IESA$FST  1
           --  PRODCICS                                     1  DUMMY    1
           --  PROG      04/24/18          2  IESEPROG  2
           --  SYSA                                     1  IESEADM   2
           --  VCSRVR                                     1  DUMMY    1

PF1=HELP                3=END
PF7=BACKWARD  8=FORWARD  9=PRINT

```

Figure 2-11 User profile information is updated

Now, the user ID HUGO is defined in the VSE control file, mapped to an LDAP user ID, and connected to groups.

Therefore, these group updates are stored in the BSM control file only. They are not yet activated. Enter fast path 283 in the entry panel z/VSE FUNCTION SELECTION to rebuild the security information in the data space and activate the changes.

**Note:** If LDAP is not installed, the panels MAINTAIN LDAP USER PROFILES are not shown. If the new user ID is a type one (administrator) user ID, the panels to connect the user ID to groups are not shown because administrators are authorized by default to access all resources.

The user ID HUGO is now ready to sign on to the Interactive Interface by using the transactions, which are provided by IBM.

In this description, you saw how you can connect a new user ID to groups by using the Interactive Interface dialog. In the next section, we show how you can create the same connections with a batch job.

### Connecting the new user ID to groups by using a batch job

Batch jobs can also be used to administer users. The BSM supports this ability with the BSTADMIN commands.

Example 2-9 shows a batch job for an environment where batch security is active. The JCL ID statement identifies an administrator user ID and provides the required authorization to use BSTADMIN. By using BSTADMIN CONNECT (C0) commands, the new user ID is connected to GROUP01, GROUP60, GROUP61, GROUP62, GROUP63, and GROUP64. This configuration is equivalent to the connections that were done by the dialog that uses the IBM-provided user ID PROG as model. The BSTADMIN command **PERFORM** (PF) activates the changes.

*Example 2-9 Batch job to connect a new user ID to default groups*

---

```
* $$ JOB JNM=BSTADMIN,CLASS=0,DISP=D
// JOB BSTADMIN
* Specify administrator user ID
// ID USER=SYSA,PWD=...
// EXEC BSTADMIN,OS3902
* Connect HUGO to GROUP01 and GROUP60 to GROUP64
CO GROUP01 HUGO
CO GROUP60 HUGO
CO GROUP61 HUGO
CO GROUP62 HUGO
CO GROUP63 HUGO
CO GROUP64 HUGO
* Activate the changes of the BSM control file
PF DATASPACE REFRESH
/*
/&
* $$ EOJ
```

---

If you are not using batch security, but you depend on command auditing, ensure that you set security option CMDUSERID, as described in “Setting option for the USERID command” on page 21.

Example 2-10 on page 30 shows that the BSTADMIN command **USERID (ID)** is used instead of the JCL ID statement. The USERID command must be the first command after BSTADMIN was started. The other commands are the same as shown in Example 2-9 on page 29.

---

<sup>2</sup> Option OS390 ensures that BSTADMIN runs in an environment with control information similar to z/OS. Option OS390 is used for the command logging with DMF.

*Example 2-10 Batch job to connect a user ID in an environment without batch security*

---

```
* $$ JOB JNM=BSTADMIN,CLASS=0,DISP=D
// JOB BSTADMIN
// EXEC BSTADMIN,OS390
ID USER(SYSA) PASSWORD(...)
* Specifies administrator user ID
* Connect HUGO to GROUP01 and GROUP60 to GROUP64
CO GROUP01 HUGO
CO GROUP60 HUGO
CO GROUP61 HUGO
CO GROUP62 HUGO
CO GROUP63 HUGO
CO GROUP64 HUGO
* Activate the changes of the BSM control file
PF DATASPACE REFRESH
/*
/&
* $$ EOJ
```

---

**Note:** If no other information is provided, all subsequent examples of batch jobs in this IBM Redbooks publication are written for an environment where batch security is active.

After connecting a new user ID to groups, you can now create a new group, as described next.

### 2.3.3 Group definition

If you want to authorize a user for a resource, you can add its user ID to the access list of the resource profile or connect it to a group that is on the access list of the resource profile.

Groups are the preferred way to provide authorizations if the protected resources are used by many users in the same way. For example, CICS transactions (provided by IBM) are such resources. The initial BSM profiles for these transactions include groups (provided by IBM) on their access list. These groups are GROUP01 and GROUP61.

The BSM as distributed with z/VSE V4R2 contains 64 groups, GROUP01 to GROUP64. These 64 groups correspond to the 64 CICS transaction security keys. Maintenance to BSM definitions might expect that those group definitions exist. Therefore, do not delete these groups.

You are responsible for defining your own groups in BSM and using these groups on access lists of resource profiles. For example, you have resources that belong to the department with the number 1234. Only the department members should have access to those resources. In this case, a department group, such as D1234, is the first choice.

To avoid unwanted behavior on access list processing because of duplicate user ID and group name, z/VSE checks for unique profile names when creating these resources. Therefore, the BSM does not allow new groups with the same name as existing user IDs, and vice versa.

This section describes how to create the group D1234 and connect user ID HUGO to this group.

Start with the entry panel z/VSE FUNCTION SELECTION and enter fast path 282 to get the panel MAINTAIN SECURITY PROFILES for class GROUP. Specify option 1 (for ADD) in front of GROUP01, as shown in Figure 2-12. Press Enter.

```

IESADMBSLG                                MAINTAIN SECURITY PROFILES
BSM RESOURCE CLASS:  GROUP
START....
OPTIONS:  1 = ADD          2 = CHANGE      5 = DELETE      6 = USER LIST
           8 = CONNECT     9 = REMOVE      USERID
           OPT  GROUP NAME  DESCRIPTION  CONNECTED?
           ---
           1  GROUP01      TRANSEC CLASS MIGRAT
           -  GROUP02      TRANSEC CLASS MIGRAT
           -  GROUP03      TRANSEC CLASS MIGRAT
           -  GROUP04      TRANSEC CLASS MIGRAT
           -  GROUP05      TRANSEC CLASS MIGRAT
           -  GROUP06      TRANSEC CLASS MIGRAT
           -  GROUP07      TRANSEC CLASS MIGRAT
           -  GROUP08      TRANSEC CLASS MIGRAT
           -  GROUP09      TRANSEC CLASS MIGRAT
           -  GROUP10      TRANSEC CLASS MIGRAT
           -  GROUP11      TRANSEC CLASS MIGRAT
           -  GROUP12      TRANSEC CLASS MIGRAT

PF1=HELP          3=END
                  8=FORWARD  9=PRINT  10=REMOVE ALL
    
```

Figure 2-12 List of groups

You see the panel to add a group with prefilled information for GROUP01. Overwrite the group name with D1234 and the description with Department 1234, as shown in Figure 2-13.

```

IESADMBSAG                                MAINTAIN SECURITY PROFILES
BSM CLASS: GROUP

Add Group:

GROUP NAME..... D1234          1 - 8 characters
DESCRIPTION..... Department 1234  Optional remark

PF1=HELP          3=END          5=UPDATE
    
```

Figure 2-13 Adding new group D1234

Press F5 to save this change in the BSM control file and return to the group selection panel.

Figure 2-14 shows the new group D1234 in the group selection panel.

```

IESADMBSLG                               MAINTAIN SECURITY PROFILES
BSM RESOURCE CLASS:   GROUP
START...
OPTIONS:   1 = ADD           2 = CHANGE           5 = DELETE           6 = USER LIST
           8 = CONNECT       9 = REMOVE           USERID
           OPT   GROUP NAME  DESCRIPTION           CONNECTED?
           6    D1234       Department 1234
           _    GROUP01     TRANSEC CLASS MIGRAT
           _    GROUP02     TRANSEC CLASS MIGRAT
           _    GROUP03     TRANSEC CLASS MIGRAT
           _    GROUP04     TRANSEC CLASS MIGRAT
           _    GROUP05     TRANSEC CLASS MIGRAT
           _    GROUP06     TRANSEC CLASS MIGRAT
           _    GROUP07     TRANSEC CLASS MIGRAT
           _    GROUP08     TRANSEC CLASS MIGRAT
           _    GROUP09     TRANSEC CLASS MIGRAT
           _    GROUP10     TRANSEC CLASS MIGRAT
           _    GROUP11     TRANSEC CLASS MIGRAT

PF1=HELP           3=END
PF7=BACKWARD      8=FORWARD      9=PRINT      10=REMOVE ALL

```

Figure 2-14 List of groups with the new group

To connect a user to this group, specify option **6** (for USER LIST) in front of D1234 and press Enter.

Figure 2-15 shows that no user is in this group. You connect the first user to this group by specifying option **1** (for ADD). Press Enter to see the next panel.

```

IESADMBSLU                               MAINTAIN USER LIST
BSM CLASS:   GROUP   GROUP: D1234
START...
OPTIONS:   1 = ADD           5 = DELETE

           OPT   USERID

           1

PF1=HELP           3=END
PF7=BACKWARD      8=FORWARD

```

Figure 2-15 Maintain an empty user list for group D1234

Enter the user ID, as shown in Figure 2-16 on page 33. Press F5 to save this change in the BSM control file.



```

IESADMBSAU                                MAINTAIN SECURITY PROFILES
BSM    CLASS: GROUP

Connect Userid to group:

GROUP NAME..... D1234                    Group name
USERID..... HUGO                          1 - 8 characters

PF1=HELP                                3=END                                5=UPDATE

```

Figure 2-16 Connect the first user ID to the new group

The next panel is again MAINTAIN USER LIST in which you can connect more user IDs to this group. When the group updates are completed, enter fast path **283** in the entry panel z/VSE FUNCTION SELECTION to rebuild the security information in the data space that activates the changes.

You now defined the new group D1234 and connected the related user IDs to this group. In the next step, you authorize this group to use certain protected resources, as described in “Specifying access rights for users or groups” on page 37.

In this section, you saw how to define a new group and connect the first user ID to this group by using Interactive Interface dialogs. In the next section, we show how to accomplish the same by using a batch job.

### Creating a group and connecting a user ID by using a batch job

Example 2-11 shows a batch job that is used to create the group D1234 and connect user ID HUGO to this group.

Example 2-11 Batch job to create a group and connect a user ID to this group

---

```

* $$ JOB JNM=BSTADMIN,CLASS=0,DISP=D
// JOB BSTADMIN
* Specify administrator user ID
// ID USER=SYSA,PWD=...
// EXEC BSTADMIN,OS390
* Add group D1234 to BSM
AG D1234 DATA('Department 1234')
* Connect HUGO to D1234
CO D1234 HUGO
* Activate the changes of the BSM control file
PF DATASPACE REFRESH
/*
/&
* $$ EOJ

```

---

Next, we show how to define a resource profile.

## 2.3.4 Resource profile definition

The following repositories are available for resource profiles:

- ▶ The DTSECTAB repository contains the entries for z/VSE files, libraries, sublibraries, and members. This repository is a table of macro calls. Skeleton DTSECTRC in ICCF library 59 contains the DTSECTAB definitions (provided by IBM). You can add your own entry and then assemble, link, and catalog it as a phase. A best practice is to IPL your system after you build a new DTSECTAB.

For more information about the DTSECTAB definitions, see *z/VSE Administration*, SC34-2692.

- ▶ The BSM control file repository keeps the profiles for all the new resource classes that are supported by BSM. Therefore, we describe those profiles next.

A profile of the BSM control file contains the profile name, a description, and several attributes in an area of fixed length. This area is followed by an area of variable length that contains the access list.

Before you can build an access list, you must create the profile.

### Creating resource profiles<sup>3</sup>

The widely used protected resources are the transactions of CICS TS. IBM provides several transaction profiles that can easily be used as examples to define new transaction profiles. For more information about CICS security (including BSM examples), see *CICS Transaction Server for VSE/ESA Security Guide*, SC33-1942.

To provide a more sophisticated example, this section demonstrates how to create profiles for class FACILITY and how to provide the access rights for users. We use the following scenario:

You have an application named ANAPPL. This application behaves as a resource manager and issues RACROUTE requests of resource class FACILITY to control access to its functions and other application resources. The resource class FACILITY is a general purpose class that can be used by many applications.

To avoid conflicts in the resource names, the first qualifier identifies the application. In this example, we use ANAPPL as application identifier. All the profile names that this application expects start with ANAPPL.

In this section, we show how to protect all resources of this application with one generic profile. Later in this section, we show how to protect two special services of this application with discrete profiles.

---

<sup>3</sup> To create resource profiles for resource classes MQADMIN, MQCMD5, MQCONN, MQNLIST, MQQUEUE, MXTOPIC, and SURROGAT, use the batch support as described in “Defining resource profiles and authorizing users” on page 43.

Starting from the entry panel z/VSE FUNCTION SELECTION, enter fast path 281. The panel BSM RESOURCE PROFILE MAINTENANCE opens, as shown in Figure 2-17.

```

IESADMSL.IESEBSCL          BSM RESOURCE PROFILE MAINTENANCE          APPLID: DBDCCICS

Enter the number of your selection and press the ENTER key:

    1 Maintain Transaction Profiles
    2 Maintain PCT Profiles
    3 Maintain DCT Profiles
    4 Maintain FCT Profiles
    5 Maintain JCT Profiles
    6 Maintain PPT Profiles
    7 Maintain TST Profiles
    8 Maintain APPL Profiles
    9 Maintain FACILITY Profiles

PF1=HELP                    3=END                    4=RETURN                    6=ESCAPE (U)
                             9=Escape (m)

==> 9                                Path: 281

```

Figure 2-17 BSM resource profile maintenance

The panel shows a list of the resource classes whose profiles are stored in the BSM control file. Specify **9** and press Enter to open the panel MAINTAIN SECURITY PROFILES, for profiles of resource class FACILITY.

Figure 2-18 shows existing profiles of resource class FACILITY. To create a profile, specify **1** (for ADD) in the option column and press Enter.

```

IESADMBSLE                MAINTAIN SECURITY PROFILES
BSM RESOURCE CLASS:      FACILITY                                ACTIVE
START...                (CASE SENSITIVE)
OPTIONS:  1 = ADD        2 = CHANGE        5 = DELETE        6 = ACCESS LIST

OPT    PROFILE NAME                DESCRIPTION                UNIVERSAL AUDIT
>                                     ACCESS VALUE

  1    DFHRCF.BRSLPU                >                                     12
  -    DFHRCF.BRSL01                >                                     12
  -    DFHRCF.BRSL02                >                                     12
  -    DFHRCF.BRSL03                >                                     12
  -    DFHRCF.BRSL04                >                                     12
  -    DFHRCF.BRSL05                >                                     12
  -    DFHRCF.BRSL06                >                                     12
  -    DFHRCF.BRSL07                >                                     12
  -    DFHRCF.BRSL08                >                                     12
  -    DFHRCF.BRSL09                >                                     12
  -    DFHRCF.BRSL10                >                                     12
  -    DFHRCF.BRSL11                >                                     12

PF1=HELP                    3=END
PF7=BACKWARD                8=FORWARD                9=PRINT                    11=NAME RIGHT

```

Figure 2-18 Adding new resource profile

Figure 2-19 shows the fields that are available when you add the new profile for class FACILITY.

```

IESADMBSAE                MAINTAIN SECURITY PROFILES
BSM RESOURCE CLASS:      FACILITY

Add Profile:

PREFIX..... _____      CICS region

RESOURCE NAME..... ANAPPL      Maximum length is 39 characters.
                                (1=yes, 2=no)

GENERIC..... 1

UNIVERSAL ACCESS... _      (_=None, 2=Read, 3=Update, 4=Alter)

AUDIT-LEVEL 1 ..... 1      (_=None, 1=Failure, 2=Success, 3=All)
ACCESS-LEVEL 1 ..... 2      (2=Read, 3=Update, 4=Alter, _=default)

AUDIT-LEVEL 2 .....      (_=None, 1=Failure, 2=Success, 3=All)
ACCESS-LEVEL 2 .....      (2=Read, 3=Update, 4=Alter, _=default)
DESCRIPTION..... An application      Optional remark
PF1=HELP                3=END                5=UPDATE

RESOURCE NAME FIELD IS CASE SENSITIVE. ENTER DATA AS REQUIRED.

```

Figure 2-19 Adding profile ANAPPL

Before you define more specific profiles, we suggest that you define one profile that protects all resources of this application and allows only administrators to use them.

The following fields are shown in Figure 2-19:

- |                  |  |
|------------------|--|
| PREFIX           | In this panel, the field is locked for input because resource class FACILITY is not a CICS class that supports CICS prefixing.                                       |
| RESOURCE NAME    | This field originally contained the name of the profile from the selection list. You change this name to ANAPPL.   |
| GENERIC          | This profile should be used for all authorization requests of this application if no other specific profiles are defined to BSM. Therefore, specify 1 (for GENERIC). |
| UNIVERSAL ACCESS | This application should not be available to every user. The default value of None ensures it.  |
| AUDIT-LEVEL 1    | We want to log unauthorized access attempts. The default level Failure is wanted.  |
| ACCESS-LEVEL 1   | All access levels should be logged. The level Read (default) includes update and alter requests. Keep the default.   |
| AUDIT-LEVEL 2    | Audit-level 1 completely specifies what we need; therefore, we do not specify audit-level 2.   |
| ACCESS-LEVEL 2   | This level is not used because audit-level 2 is not specified.   |
| DESCRIPTION      | This field can be used to specify up to 20 characters of installation-defined data. In the example, enter An application.  |

With F5 (for UPDATE), you save this new resource profile in the BSM control file and return to the profile selection list.

Figure 2-20 shows this new profile in the profile selection list. An asterisk (\*) appears in front of the profile name to indicate that this profile is generic.

```

IESADMBSLE                MAINTAIN SECURITY PROFILES
BSM RESOURCE CLASS:      FACILITY                ACTIVE
START...                  (CASE SENSITIVE)
OPTIONS:  1 = ADD        2 = CHANGE        5 = DELETE        6 = ACCESS LIST

   OPT   PROFILE NAME                DESCRIPTION                UNIVERSAL AUDIT
   >
   -     *ANAPPL                    An application                12
   -     DFHRCF.BRSLPU                12
   -     DFHRCF.BRSL01                12
   -     DFHRCF.BRSL02                12
   -     DFHRCF.BRSL03                12
   -     DFHRCF.BRSL04                12
   -     DFHRCF.BRSL05                12
   -     DFHRCF.BRSL06                12
   -     DFHRCF.BRSL07                12
   -     DFHRCF.BRSL08                12
   -     DFHRCF.BRSL09                12
   -     DFHRCF.BRSL10                12

PF1=HELP                3=END
PF7=BACKWARD          8=FORWARD          9=PRINT                11=NAME RIGHT

```

Figure 2-20 Profile selection list with the new generic profile ANAPPL

Because you do not want to use an access list for this profile, press F3 to return to the entry panel z/VSE FUNCTION SELECTION and enter fast path 283 to rebuild the security information in the data space, which activates this new profile.

### Specifying access rights for users or groups

In this section, we extend the following example:

Assume that this application includes a broadcast service that everyone can use to get information. It also includes a database service that a defined group of users can use for read requests and only a few users can use for update requests. The application ANAPPL uses the resource names ANAPPL.BROADCAST.SERVICE and ANAPPL.DATABASE.SERVICE to check the authorization.

To define the profiles, start with entry panel z/VSE FUNCTION SELECTION and enter fast path 2819. You see the profile selection list of class FACILITY. Enter option 1 to add a new profile, as shown in Figure 2-21 on page 38, press Enter.

```

IESADMBSLE                MAINTAIN SECURITY PROFILES
BSM RESOURCE CLASS:      FACILITY                ACTIVE
START...                (CASE SENSITIVE)
OPTIONS:   1 = ADD        2 = CHANGE        5 = DELETE        6 = ACCESS LIST

      OPT      PROFILE NAME                DESCRIPTION                UNIVERSAL AUDIT
      >
      1      *ANAPPL                An application                12
      -      DFHRCF.BRSLPU                12
      -      DFHRCF.BRSL01                12
      -      DFHRCF.BRSL02                12
      -      DFHRCF.BRSL03                12
      -      DFHRCF.BRSL04                12
      -      DFHRCF.BRSL05                12
      -      DFHRCF.BRSL06                12
      -      DFHRCF.BRSL07                12
      -      DFHRCF.BRSL08                12
      -      DFHRCF.BRSL09                12
      -      DFHRCF.BRSL10                12

PF1=HELP                3=END
PF7=BACKWARD      8=FORWARD      9=PRINT                11=NAME RIGHT

```

Figure 2-21 Add a new profile in profile selection list

The panel MAINTAIN SECURITY PROFILES opens for class FACILITY, as shown in Figure 2-22.

First, add the profile for the broadcast service, as shown in Figure 2-22.

```

IESADMBSAE                MAINTAIN SECURITY PROFILES
BSM RESOURCE CLASS:      FACILITY

Add Profile:

PREFIX.....                CICS region

RESOURCE NAME..... ANAPPL.BROADCAST.SERVICE
Maximum length is 39 characters.
GENERIC..... 2                (1=yes, 2=no)
UNIVERSAL ACCESS... 2                (_=None, 2=Read, 3=Update, 4=Alter)
AUDIT-LEVEL 1 ..... 1                (_=None, 1=Failure, 2=Success, 3=All)
ACCESS-LEVEL 1 .... 2                (2=Read, 3=Update, 4=Alter, _=default)
AUDIT-LEVEL 2 .....                (_=None, 1=Failure, 2=Success, 3=All)
ACCESS-LEVEL 2 ....                (2=Read, 3=Update, 4=Alter, _=default)
DESCRIPTION..... An application        Optional remark
PF1=HELP                3=END                5=UPDATE

RESOURCE NAME FIELD IS CASE SENSITIVE. ENTER DATA AS REQUIRED.

```

Figure 2-22 Adding profile ANAPPL.BROADCAST.SERVICE

Specify the complete resource name and indicate that it is not a generic profile. To allow read access to all users, you specify universal access READ.

Press F5 (for UPDATE) to save this new profile and return to the profile selection list.

Figure 2-23 shows the new profile ANAPPL.BROADCAST.SERVICE and its universal access value 2 for READ.

```

IESADMBSLE                MAINTAIN SECURITY PROFILES
BSM RESOURCE CLASS:      FACILITY                ACTIVE
START...                  (CASE SENSITIVE)
OPTIONS:  1 = ADD        2 = CHANGE        5 = DELETE        6 = ACCESS LIST

      OPT    PROFILE NAME                DESCRIPTION                UNIVERSAL AUDIT
      >
      1      ANAPPL.BROADCAST.SERVICE    An application            2          12
      -      *ANAPPL                      An application            -          12
      -      DFHRCF.BRSLPU                -                          -          12
      -      DFHRCF.BRSL01                -                          -          12
      -      DFHRCF.BRSL02                -                          -          12
      -      DFHRCF.BRSL03                -                          -          12
      -      DFHRCF.BRSL04                -                          -          12
      -      DFHRCF.BRSL05                -                          -          12
      -      DFHRCF.BRSL06                -                          -          12
      -      DFHRCF.BRSL07                -                          -          12
      -      DFHRCF.BRSL08                -                          -          12
      -      DFHRCF.BRSL09                -                          -          12

PF1=HELP                3=END
PF7=BACKWARD          8=FORWARD          9=PRINT                11=NAME RIGHT
  
```

Figure 2-23 Showing new profile selection list and add a new profile

Set option 1 (for ADD) and press Enter to define another new profile.

The panel MAINTAIN SECURITY PROFILES opens. Change it as shown in Figure 2-24.

```

IESADMBSAE                MAINTAIN SECURITY PROFILES
BSM RESOURCE CLASS:      FACILITY

Add Profile:

PREFIX.....             _____             CICS region

RESOURCE NAME..... ANAPPL.DATABASE.SERVICE
Maximum length is 39 characters.

GENERIC..... 2          (1=yes, 2=no)

UNIVERSAL ACCESS... _   (_=None, 2=Read, 3=Update, 4=Alter)

AUDIT-LEVEL 1 ..... 1   (_=None, 1=Failure, 2=Success, 3=All)
ACCESS-LEVEL 1 .... 2   (2=Read, 3=Update, 4=Alter, _=default)

AUDIT-LEVEL 2 ..... 2   (_=None, 1=Failure, 2=Success, 3=All)
ACCESS-LEVEL 2 .... 3   (2=Read, 3=Update, 4=Alter, _=default)
DESCRIPTION..... An application             Optional remark
PF1=HELP                3=END                5=UPDATE

RESOURCE NAME FIELD IS CASE SENSITIVE. ENTER DATA AS REQUIRED.
  
```

Figure 2-24 Adding profile ANAPPL.DATABASE.SERVICE

The profile ANAPPL.DATABASE.SERVICE is not generic. The default setting None (for UNIVERSAL ACCESS) is used because only selected users should have access to it. AUDIT-LEVEL 1 and ACCESS-LEVEL 1 show the initial value to log all access violations at level 2 (Read). We also want successful update requests to be logged. Therefore, AUDIT-LEVEL 2 and ACCESS-LEVEL 2 are set.

Press F5 (for UPDATE) to save this new profile and go back to the profile selection list, as shown in Figure 2-25.

```

IESADMBSLE                MAINTAIN SECURITY PROFILES
BSM RESOURCE CLASS:      FACILITY                ACTIVE
START...                  (CASE SENSITIVE)
OPTIONS:  1 = ADD        2 = CHANGE        5 = DELETE        6 = ACCESS LIST

      OPT      PROFILE NAME                DESCRIPTION                UNIVERSAL AUDIT
      -      ANAPPL.BROADCAST.SERVICE    An application                2      12
      6      ANAPPL.DATABASE.SERVICE    An application                23     12
      -      *ANAPPL                      An application
      -      DFHRCF.BRSLPU                12
      -      DFHRCF.BRSL01                12
      -      DFHRCF.BRSL02                12
      -      DFHRCF.BRSL03                12
      -      DFHRCF.BRSL04                12
      -      DFHRCF.BRSL05                12
      -      DFHRCF.BRSL06                12
      -      DFHRCF.BRSL07                12
      -      DFHRCF.BRSL08                12

PF1=HELP                3=END
PF7=BACKWARD          8=FORWARD          9=PRINT                11=NAME RIGHT
  
```

Figure 2-25 Show new profile selection list and specify option 6

You want only selected users to have access to the database service. To modify the access list of the profile ANAPPL.DATABASE.SERVICE, specify option 6 next to the profile name and press Enter.

Figure 2-26 shows the empty access list of this new profile. Specify option 1 to add a new entry to the access list and press Enter.

```

IESADMBSLA                MAINTAIN ACCESS LIST
BSM CLASS: FACILITY      PROFILE: ANAPPL.DATABASE.SERVICE
START...                  NUMBER OF ENTRIES ON LIST: 00000
OPTIONS:  1 = ADD        2 = CHANGE        5 = DELETE

      OPT      NAME      ACC

      1

PF1=HELP                3=END
PF7=BACKWARD          8=FORWARD
  
```

Figure 2-26 Empty access list



In the next panel, you specify that the group D1234 can use the database service to read data (see Figure 2-27).

```

IESADMBSAA                MAINTAIN ACCESS LIST
BSM   CLASS: FACILITY     PROFILE: ANAPPL.DATABASE.SERVICE

Add Userid or Groupid:

NAME..... D1234          Userid or Groupid
ACCESS..... 2             (_=None,
                           2=Read, 3=Update, 4=Alter)

PF1=HELP                    3=END                    5=UPDATE

```

Figure 2-27 Adding group D1234 to the access list

Press F5 for update to save the profile together with this entry and return to the access list.

The access list that is shown in Figure 2-28 shows the group D1234 with the access right read. Notice the option 2 (for CHANGE). By using this option, you can directly change the access rights of a user or group in this panel.

```

IESADMBSLA                MAINTAIN ACCESS LIST
BSM   CLASS: FACILITY     PROFILE: ANAPPL.DATABASE.SERVICE
START....                NUMBER OF ENTRIES ON LIST: 00001
OPTIONS:  1 = ADD         2 = CHANGE         5 = DELETE

   OPT   NAME   ACC
   ----   ---   ---
   1     D1234   2

PF1=HELP                    3=END
PF7=BACKWARD                8=FORWARD

```

Figure 2-28 Access list with one entry

Based on this example, we show next how a single user can be added to the access list.

Specify option 1 (for ADD) and press Enter.

With the definitions that are shown in Figure 2-29, user Hugo can use the database service to update data.

```
IESADMBSAA                MAINTAIN ACCESS LIST
BSM      CLASS: FACILITY   PROFILE: ANAPPL.DATABASE.SERVICE

Add Userid or Groupid:

NAME..... HUGO           Userid or Groupid
ACCESS..... 3             (_=None,
                          2=Read, 3=Update, 4=Alter)

PF1=HELP                    3=END                    5=UPDATE
```

Figure 2-29 Adding user ID HUGO to the access list

User Hugo is also connected to group D1234. Therefore, user Hugo has READ authorization from the group entry and UPDATE from its user ID entry in the access list. Which authorization is now used by the BSM?

The BSM features the following rules:

- ▶ If the user ID is defined on the access list, this authorization is used, independent of whether the access right is sufficient. No group on the access list is checked.  
Normally, you use this rule to give a single user a higher access authority than the rest of their group. However, you might use it in reverse by specifying NONE for a user ID to remove that user from accessing this resource, where the rest of its group is authorized.
- ▶ If the user ID is not defined on the access list, the groups on the access list that have the requested access authorization are scanned for this user ID. If the user is connected to such a group, that user is authorized to use this resource. Otherwise, the access attempt fails.

Return to the panel by pressing F5 (for UPDATE), save this entry, and return to the access list, as shown in Figure 2-30.

```

IESADMBSLA                                MAINTAIN ACCESS LIST
BSM CLASS: FACILITY                        PROFILE: ANAPPL.DATABASE.SERVICE
START...                                  NUMBER OF ENTRIES ON LIST: 00002
OPTIONS:  1 = ADD                          2 = CHANGE          5 = DELETE

      OPT   NAME   ACC
      --   --    --
      -   D1234   2
      -   HUGO    3

PF1=HELP                                3=END
PF7=BACKWARD  8=FORWARD

```

Figure 2-30 Access after adding user ID HUGO

You defined the access list as required. Return by pressing F3 to the entry panel z/VSE FUNCTION SELECTION and enter fast path 283 to rebuild the security information in the data space, which activates this profile with the access list.

In this section, we showed how to define a new resource profile and how to authorize users for these resources by using Interactive Interface panels. In the next section, we show how you can accomplish the same task by using a batch job.

## Defining resource profiles and authorizing users

Example 2-12 shows a batch job to create the group D1234 and connect user ID HUGO to this group.

Example 2-12 Batch job to define resource profiles and authorize users for them

```

* $$ JOB JNM=BSTADMIN,CLASS=0,DISP=D
// JOB BSTADMIN
* Specify administrator user ID
// ID USER=SYSA,PWD=...
// EXEC BSTADMIN,OS390
* Add generic resource profile ANAPPL to BSM
AD FACILITY ANAPPL GEN DATA('An application')
* Add resource profile ANAPPL.BROADCAST.SERVICE to BSM
AD FACILITY ANAPPL.BROADCAST.SERVICE UACC(READ) DATA('An application')
* Add resource profile ANAPPL.DATABASE.SERVICE to BSM
AD FACILITY ANAPPL.DATABASE.SERVICE AUDIT(SUCC(UPD),FAIL(FAIL)) -
DATA('An application')
* Permit access to resource for group D1234 and user ID HUGO
PE FACILITY ANAPPL.DATABASE.SERVICE ID(D1234) ACCESS(READ)
PE FACILITY ANAPPL.DATABASE.SERVICE ID(HUGO) ACCESS(UPDATE)
* Activate the changes of the BSM control file
PF DATASPACE REFRESH
/*

```

```
/&
* $$ E0J
```

---

After defining user profiles, group profiles, and resource profiles, you are ready to use the BSM audit function.

### Protecting selected operands of JCL statements

Job Control Commands (JCCs) and Job Control Statements (JCSs) whose effect extends beyond the end of the current VSE job are subject to security checks.

For example, a job stream that uses PERM on an ASSGN (implicit or explicit), on a LIBDEF or LIBDROP, or that uses the OPTION PARSTD/STDLABEL, changes the system that remain in effect to the end of the current IPL.

For now, IBM provides resource profiles in class FACILITY to control access to these operands. You can use the dialog, as shown in Figure 2-31, to maintain these profiles and their access lists.

IESADMBSLE				MAINTAIN SECURITY PROFILES			
BSM RESOURCE CLASS:		FACILITY		ACTIVE			
START . . .		IBMVSE		(CASE SENSITIVE)			
OPTIONS:		1 = ADD	2 = CHANGE	5 = DELETE	6 = ACCESS LIST		
OPT	PROFILE NAME	DESCRIPTION		UNIVERSAL AUDIT	ACCESS VALUE		
		>					
-	IBMVSE.JCL.ASSGN.PERM				12		
-	IBMVSE.JCL.LIBDEF.PERM				12		
-	IBMVSE.JCL.LIBDROP.PERM				12		
-	IBMVSE.JCL.OPTION.PARSTD				12		
-	IBMVSE.JCL.OPTION.STDLABEL				12		

PF1=HELP		3=END	
PF7=BACKWARD	8=FORWARD	9=PRINT	11=NAME RIGHT

Figure 2-31 Maintain resource profiles for JCL operands

To activate the authorization checks for these resources, specify the JCL parameter in the IPL SYS SEC command, as shown is the following example:

```
SYS SEC=(YES,JCL)
```

For more information, see *z/VSE Planning*, SC34-2681 and *z/VSE Administration*, SC34-2692.

## 2.3.5 Batch resource administration

In z/VSE 6.2, the dialog-supported resource administration is extended to include the following resource classes:

- ▶ DATASET: z/VSE files
- ▶ VSELIB: z/VSE libraries
- ▶ VESLIB: z/VSE sublibraries
- ▶ VSEMEM: z/VSE librarian members

These resource classes are stored in the DTSECTAB repository, not in the BSM control file repository. The DTSECTAB repository keeps its central role to control the access to z/VSE batch resources, even when the batch resource classes are maintained with the dialog.

When modifying the batch resources, the following process is used:

1. Read the DTSECTAB.PHASE repository into a temporary VSAM/KSDS cluster IESBSTW. To allow the dialog to read the version that is in SVA, the CICS DFHSIT should be configured with "SVA=YES".
2. For the duration of the dialog, the added and modified entries are kept in the temporary IESBSTW cluster.
3. When the changes are saved, the corresponding DTSECTAB macro definitions are created by the dialog and a batch job is submitted to assemble, link, and catalog a new DTSECTAB.PHASE. For that process, the CICS DFHSIT must be configured with "SPOOL=YES".

Only after cataloging a new DTSECTAB are the changes recognized by the BSM.

### Creating resource profiles

Starting from the entry panel z/VSE FUNCTION SELECTION, enter fast path 286. The panel RESOURCE CLASSES opens, as shown in Figure 2-32.

IESADMBSLC		BSM RESOURCE CLASSES	
START....			
OPTIONS:		6=PROFILE LIST	
OPT	RESOURCE CLASS NAME	STATUS	
-	ACICSPCT	ACTIVE	
-	APPL	ACTIVE	
-	DATASET	ACTIVE	
-	DCICSDCT	ACTIVE	
-	FACILITY	ACTIVE	
-	FCICSFCT	ACTIVE	
-	JCICSJCT	ACTIVE	
-	MCICSPPT	ACTIVE	
-	MQADMIN	ACTIVE	
-	MQCMDS	ACTIVE	
-	MQCONN	ACTIVE	
-	MQNLIST	ACTIVE	
PF1=HELP      2=REFRESH      3=END			
8=FORWARD      9=PRINT			

Figure 2-32 Unified BSM Resource Profile Maintenance

Select one of the VSE batch resource classes by specifying **6** in the OPT field and press Enter.

Figure 2-33 shows existing entries of resource class VSEMEM. To create an entry, specify 1 (for ADD) in the option column and press Enter.

```

IESADMBSLD                MAINTAIN SECURITY PROFILES
BSM RESOURCE CLASS: VSEMEM (START is Case Sensitive)  STATUS: ACTIVE
START.... PRD1.BASE.
OPTIONS:  1 = ADD        2 = CHANGE        5 = DELETE        6 = ACCESS LIST

  OPT  PROFILE NAME                                UNIVERSAL
                                           ACCESS
  -    *PRD1.BASE.                                2
  -    *PRD1.BASED.                                2
  -    *PRD1.MACLIB.                                2
  -    *PRD1.MACLIBD.                              2
  1    *PRD2.AFP.                                  2
  -    *PRD2.COMM.                                  2
  -    *PRD2.COMM2.                                 2
  -    *PRD2.CONFIG.DTSEC
      PRD2.CONFIG.SRV$SYS                          3
  -    *PRD2.CONFIG.                              3
  -    *PRD2.DBASE.                                2
  -    *PRD2.DB2STP.                               2

PF1=HELP                    3=END
PF7=BACKWARD  8=FORWARD

```

Figure 2-33 ADD new entry for VSE librarian member

Figure 2-34 shows the fields that are available when you add the new entry for class VSEMEM.

```

IESADMBSAD                MAINTAIN SECURITY PROFILES
BSM RESOURCE CLASS:  VSEMEM

Add Profile:

VOLID..... _____  Where the file or library resides.

RESOURCE NAME.....     Maximum length is 25 characters.
..... PRD2.AFP.

GENERIC.....           (1=yes, 2=no)

UNIVERSAL ACCESS... _  (_=None, 1=Connect,
                       2=Read, 3=Update, 4=Alter)

PF1=HELP                    3=END                    5=UPDATE

```

Figure 2-34 Adding new VSEMEM profile

The following fields are shown in Figure 2-34 on page 46:

- VOLID** In this panel, the field is locked for input because VSEMEM is not a batch class that supports VOLID specification.
- RESOURCE NAME** This field originally contains the name of the entry from the selection list. Change this name to the member name required. Specify in LIB.SUBLIB.MEMBER format.
- GENERIC** When an entry should match multiple discrete names that start with the given name, specify 1 (for GENERIC).
- UNIVERSAL ACCESS** Specify the access right valid for all users (not having private access right specified).

With PF5 (for UPDATE), you save your new entry in the temporary VSAM/KSDS file IESBSTW. When exiting the dialog, you are requested to catalog a new DTSECTAB, as shown in Example 2-35.

```
IESADMBSGE                MAINTAIN SECURITY PROFILES
BSM BATCH CLASSES

The BSM batch classes were modified. To store the modifications
permanent they must be cataloged into the system library.

CATALOG BATCH CLASSES.... 1          (1=yes, 2=no)

Authentication information for catalog job:

USERID..... WACK          Your userid will be used

PASSWORD.....            Specify your password

PF1=HELP                  3=END
```

Figure 2-35 Cataloging a new DTSECTAB

When you request to catalog the batch classes, a job is submitted to batch to assemble, link, and catalog the new DTSECTAB. It is a good practice to view the listing of the job (jobname: DTSECTRC) for errors.

### Customizing the JCL for cataloging DTSECTAB

The job that is submitted in the IESADMBSGE panel is created out of a job skeleton. The job skeleton is named IESECTAB.HTML and stored into library IJSYSRS.SYSLIB. The default job skeleton provides the following functions:

- ▶ Authenticate the job with the specific user ID and password.
- ▶ Specify the DTSECTAB entries for users “DUMMY” and “FORSEC”.
- ▶ Catalog a DTSECTAB source version in library PRD2.CONFIG.
- ▶ Perform an assemble, link, and catalog step for the generated DTSECTAB source part.

If the default functionality must be modified, the skeleton IESECTAB.HTML can be modified. The help panel of IESADMBSGE includes more information about how to manage the job authentication. Save the original part before IESECTAB.HTML is modified.

## 2.3.6 Generating BSM cross-reference reports

In a complex and large user environment with many user IDs, groups, and resource profiles, you might need more information to ensure that all the different definitions provide a consistent security setup.

Therefore, the BSM provides cross-reference reports with information about the following components:

- ▶ User IDs
- ▶ Groups
- ▶ Access control classes (ACCs) that are used with DTSECTAB
- ▶ Resources that can be accessed by way of the UACC definition in the profile
- ▶ Undefined user IDs that are found in groups and access lists of resource profiles

By using these reports, you can manage the profile definitions that are stored in the following repositories:

- ▶ VSE control file (IESCNTL)
- ▶ Table DTSECTAB
- ▶ BSM control file (BSTCNTL)

BSM cross-reference reports are especially useful for checking that after deleting a user ID, you deleted a user ID from all profile access lists and groups after you deleted the user ID.

You can generate your BSM cross-reference reports by using one of the following methods:

- ▶ BSTXREF service
- ▶ BSM Cross Reference Report panel (fast path 286)

In both cases, you use jobs that run BSTXREF and create the reports in the VSE/POWER list queue.

For example, you want to get more information about user ID HUGO and access control class 6. Start a job with the BSTXREF calls, as shown in Example 2-13.

*Example 2-13 Creating cross-reference report for a user ID and an access control class*

---

```
* $$ JOB JNM=BSTXREF,CLASS=0,DISP=D
// JOB BSTXREF
*      This job creates BSM cross reference reports
*      Enter ID statement or press enter to continue
// PAUSE
// EXEC BSTXREF,PARM='USERID=HUGO'
/*
// EXEC BSTXREF,PARM='ACC=6'
/*
/&
* $$ E0J
```

---

The report that is shown in Example 2-14 on page 49 shows that the user ID definition for HUGO exists in the VSE control file. The user ID is connected to eight groups and is defined on the access lists of three resource profiles. HUGO has read authority by way of access control class 6 to DTSECTAB resources.



*Example 2-14 Cross-reference report of user ID HUGO*

---

BSM Cross Reference Report  
of User ID HUGO

Occurrences of user HUGO

User entry exists  
Connected to group D1234  
Connected to group GROUP01  
Connected to group GROUP60  
Connected to group GROUP61  
Connected to group GROUP62  
Connected to group GROUP63  
Connected to group GROUP64  
Read authority in access list of profile APPL DBDCCICS  
Read authority in access list of profile FACILITY DITTO.SPOOL.DISPLAY  
Update authority in access list of profile FACILITY MSGTEST  
Read authority for access control class 6

(G) - Profile name generic.

\* - Truncation indication, if shown at the end of large profile names.

---

Example 2-15 shows the report for access control class 6. The user ID HUGO has read authority to the resources protected by DTSECTAB profiles, which have ACC=(6) specified. In this example, eight profiles are shown.

*Example 2-15 Cross-reference report of access control class 6*

---

BSM Cross Reference Report  
of DTSECTAB Access Control Class (ACC) 6

Occurrences of access control class 6

HUGO has read authority for this class  
It is specified in profile DATASET VOL001.TEST01.FILE  
It is specified in profile DATASET VOL002.TEST02.FI (G)  
It is specified in profile VSELIB \*.VSE.PRD7.LIBRARY.PRD7  
It is specified in profile VESLIB PRD7.TESTSL01  
It is specified in profile VESLIB PRD7.TESTSL02  
It is specified in profile VSEMEM PRD7.TESTSL01.A1234567  
It is specified in profile VSEMEM PRD7.TESTSL01.A (G)  
It is specified in profile VSEMEM PRD7.TESTSL02.C (G)

(G) - Profile name generic.

\* - Truncation indication, if shown at the end of large profile names.

---

For more information about the other BSM cross-reference reports, see *z/VSE Administration*, SC34-2692.

## 2.4 BSM auditing

One of the many reasons to run a security manager on your system is to have a tool that can help you determine whether any attempts were made to bypass system integrity. BSM audit data is a record of an installation's security-relevant events. This data is used to verify the effectiveness of an installation's security policy, determine whether the installation's security objectives are being met, and identify unexpected security-relevant events.

As all other security managers, the BSM can customize auditing requirements for your installation, which allows you to meet your corporate security policies. The BSM uses attributes to allow or disallow BSM command execution. The administrator attribute (type 1 user) allows a user to use the security administration dialogs in the Interactive Interface or to issue BSM commands in a batch job. The option CMDUSERID also enforces this requirement for an environment in which batch security is not active.

Similar to the attributes for BSM command execution, the BSM uses several attributes to control the audit functions.

From z/VSE V5.2 on, the BSM optionally separates the auditor function from the administrator function, which implements the check and balance concept for VSE security. The BSM introduces a new user ID of type AUDITOR and allows you to assign the administration of the system-wide audit options to the AUDITOR only. The administrator keeps the responsibility to process logging information, but no longer has any auditor rights. The Interactive User Interface (IUI) of z/VSE is extended to define the AUDITOR ID.

For example, the user ID SYSN is defined as system administrator (user type=1) but has no AUDITOR capabilities. Having established the logging of administrator accesses (ADMINACC) in our environment, the user SYSN tries to stop logging for administrator resource accesses. Although that SYSN is a system administrator, the PF function fails because of insufficient authorization. A sample BSTADMIN run to attempt this process is shown in Example 2-16.

*Example 2-16 Non-auditor user SYSN trying to change AUDIT settings.*

---

```
BG 0000 // JOB PAUSEBG
          DATE 03/28/2018, CLOCK 14/28/58
BG-0000 // PAUSE
BG 0000 // ID (PARAMETERS SUPPRESSED)
BG-0000
0 exec bstadmin
BG 0000 1S54I PHASE BSTADMIN IS TO BE FETCHED FROM IJSYSRS.SYSLIB
BG-0000 BST901A ENTER COMMAND OR END
0 pf audit noadminacc
BG 0000 BST932I USER NOT AUTHORIZED TO SPECIFY AUDIT, KEYWORD IGNORED
BG 0000 BST904I RETURN CODE OF PERFORM IS 08
BG-0000 BST901A ENTER COMMAND OR END
```

---

## 2.4.1 Enabling auditing for resources defined in the BSM control file

You can enable (audit) or disable (noaudit) functions dynamically to meet the requirements of your installation. When you enable the feature to collect the audit records, SMF<sup>4</sup> records are generated. For more information about available options, see 2.3.1, “Security system settings” on page 17.

**Note:** Certain events are always logged by BSM, other events are never logged by BSM, and some events are optionally logged by BSM.

The BSM always logs information about certain events that are essential to an effective data-security mechanism. The BSM always logs:

- ▶ Every use of the BSTADMIN command PERFORM with keywords AUDIT, SETOPT, and PASSWORD
- ▶ Whenever a RACROUTE REQUEST=VERIFY (sign-on) request fails

The BSM never logs certain events because knowing about these events is not essential to effective data security. BSM never logs any use of the following BSTADMIN commands:

- ▶ LIST
- ▶ LISTG
- ▶ LISTU
- ▶ PERFORM with keyword DATASPACE
- ▶ STATUS

The BSM can optionally log other events. Optional logging is under the control of the administrator who defined the resource profile or the resource manager that issues the RACROUTE request.

The next step in establishing the auditing environment is to decide whether you must log resource access attempts from an administrator user ID. Administrators have access to all resources. Many subsystems, such as CICS or TCP/IP, are running with the authorization of an administrator. Logging of the administrator access creates many SMF records. However, for effective data security logging, those events also are necessary.

Use the BSTADMIN command **PERFORM** with keyword **AUDIT** and option **ADMINACC** to activate logging of administrator access attempts.

After the auditing for resource access is established, you can activate BSM resource classes for command auditing by using the **BSTADMIN** command keyword **CLASS** and option **CMDAUDIT**. This option is applicable for the following classes:

- ▶ GROUP
- ▶ TCICSTRN
- ▶ ACICSPCT
- ▶ DCICSDCT
- ▶ FCICSFCT
- ▶ JCICSJCT
- ▶ MCICSPPT
- ▶ SCICSTST
- ▶ APPL
- ▶ FACILITY

---

<sup>4</sup> SMF records have their origin in the System Management Facility of z/OS. The audit records of BSM are based on the SMF 80 record.

For class GROUP, the following commands are audited:

- ▶ ADDROUP
- ▶ CHNGROUP
- ▶ DELGROUP
- ▶ CONNECT
- ▶ REMOVE

The following audited commands are available for the other resource classes:

- ▶ ADD
- ▶ CHANGE
- ▶ DELETE
- ▶ PERMIT

Use the BSTADMIN command **STATUS** to determine your current AUDIT environment, as shown in Example 2-2 on page 18.

## 2.4.2 Enabling auditing for resources defined in the DTSECTAB

Previously, the licensed program Access Control Logging and Reporting (ACLR) was the preferred program for a comprehensive auditing of DTSECTAB resources. With z/VSE V4R3, you can use DMF instead of ACLR for this auditing.

To prepare your system to create SMF records and send them to DMF, you must catalog only the \$SVALOG.PHASE in IJSYSRS.SYSLIB. You can use the skeleton SKSECLOG from ICCF library 59. The phase \$SVALOG ensures that the logger program is loaded in the system when the batch security is active.

**Note:** As in ACLR, access violations and administrator access to DTSECTAB resources are always logged.

If the DMF is not active, you do not receive any audit record from the logger program.

## 2.4.3 DMF setup

As with z/OS, the BSM uses System Management Facility (SMF) type 80 records to log security events. On z/VSE, we use the Data Management Facility (DMF) to collect and extract SMF records.

DMF is a part of CICS Transaction Server for VSE/ESA. It runs in a separate partition. CICS is not required to be active to run DMF.

DMF uses VSAM data sets to store the data. The z/VSE initialization defines two DMF data sets, CICS.DBCCICS.DFHDMFA and CICS.DBCCICS.DFHDMFB, as shown in Example 2-17.

*Example 2-17 Defining DMF data sets*

---

```
// JOB DEFINE DMF DATA SETS
// EXEC IDCAMS,SIZE=AUTO
DEFINE CLUSTER (NAME(CICS.DBCCICS.DFHDMFA) -
NONINDEXED -
RECORDS(2000,500) -
REUSE -
RECORDSIZE (125 32767) -
SPANNED -
```

```

VOLUMES (DOSRES,SYSWK1) -
NOCOMPRESSED -
SHAREOPTIONS (2) -
TO (99366 ) -
DATA (NAME (CICS.DBCCICS.DFHDMFA.@D@) -
CONTROLINTERVALSIZE (4096)) -
CATALOG (VSESP.USER.CATALOG)
DEFINE CLUSTER (NAME(CICS.DBCCICS.DFHDMFB) -
NONINDEXED -
RECORDS(2000,500) -
REUSE -
RECORDSIZE (125 32767) -
SPANNED -
VOLUMES (DOSRES SYSWK1) -
NOCOMPRESSED -
SHAREOPTIONS (2) -
TO (99366 ) -
DATA (NAME (CICS.DBCCICS.DFHDMFB.@D@) -
CONTROLINTERVALSIZE (4096)) -
CATALOG (VSESP.USER.CATALOG)

```

/\*

You can define up to 36 DMF data sets. However, in this IBM Redbooks publication, we use only these two DMF data sets to demonstrate setting up the DMF setup and the processing of SMF records.

In the next step, we show how to define the DMF initialization table. For the DMF setup, you can use the skeleton DFHDMFSP form ICCF library 59, as shown in Example 2-18.

*Example 2-18 Skeleton to define DMF initialization table*

```

* $$ JOB JNM=DFHDMFSP,CLASS=A,DISP=D,NTFY=YES
* $$ LST CLASS=Q,DISP=H
// JOB DFHDMFSP ASSEMBLE
// LIBDEF *,CATALOG=PRD2.CONFIG
// LIBDEF SOURCE,SEARCH=(PRD1.BASE,PRD1.MACLIB)
// OPTION CATAL,LIST
// EXEC ASMA90,SIZE=(ASMA90,64K),PARM='EXIT(LIBEXIT(EDECKXIT)),SIZE(MAXC
-200K,ABOVE)'
*****
*
* 5686-066 (C) COPYRIGHT IBM CORP. 1996
*
*****
TITLE 'DFHDMFSP -- SUPPLIED BY VSE/ESA'
PUNCH ' CATALOG DFHDMFSP.OBJ REP=YES'
DFHDMFM TABLE,
CATALOG=VSESP.USER.CATALOG, USE VSESPUC
FILELIST=(CICS.DBCCICS.DFHDMFA,CICS.DBCCICS.DFHDMFB),
INTERVAL=3000, 30 MINUTES 0 SECONDS
LISTDSN=YES, SHOW DATASETS WHEN DMF STARTS
SID=VSE, SYSTEM IDENTIFIER
SIZE=4, USE A 4M DATA SPACE
STATUS=ACTIVE, DMF IS ACTIVE AT START
SUFFIX=SP, THIS TABLE IS CALLED DFHDMFSP
TRACE=NO, NO TRACE ACTIVITY

```

```

                TRTABSZ=1024,      TRACE TABLE SIZE IS 1M      *
                TYPE=0:255,        RECORD ALL DMF DATA RECORD TYPES *
                USAGE=50           REDUCE SPACE WHEN 50% FULL
        END
/*
// IF $MRC GT 4 THEN
// GOTO NOLINK
// EXEC LNKEDT,PARM='MSHP'
/. NOLINK
/*
/&
* $$ E0J

```

---

For more information, see *CICS Transaction Server for VSE/ESA Operations and Utilities Guide*, SC33-1654.

After you build the DMF initialization table, put the DMF start job into the VSE/POWER reader queue. By default, this job runs in the dynamic partition Z. The job requires at least 5 MB of storage and 4 MB of DSPACE. You can use skeleton SKDMFST in ICCF library 59, as shown in Example 2-19.

*Example 2-19 DMF start job*

---

```

* $$ JOB JNM=DMFSTART,CLASS=Z,DISP=D,NTFY=YES
* $$ LST CLASS=Q,DISP=H
// JOB DMFSTART DMF STATISTIC SERVER
* -----*
*   THIS JOB STARTS THE DMF SERVER PARTITION FOR CICS/TS      *
* -----*
* FOR COMMUNICATION USE 'MSG XX' (PARTITION ID) AND
* 'XX SETDMF ...' COMMANDS.
* TERMINATING:
* 'XX SETDMF SHUTDOWN'
* -----*
// EXEC DFHDFOU
INDD ( DFHDMFA, OPTIONS (CLEAR) )
/*
// EXEC DFHDFOU
INDD ( DFHDMFB, OPTIONS (CLEAR) )
/*
// EXEC DFHDFSIP,SIZE=DFHDFSIP,OS390
SUFFIX=SP
/*
/&
* $$ E0J

```

---

The program DFHDFOU clears a DMF data set. EXEC DFHDFSIP starts the DMF server. As with CICS TS, this server requires a specific environment with a control block structure, which is similar to z/OS. Therefore, the option OS390 is specified in this job.

After the job DMFSTART is placed in the VSE/POWER reader queue, you must define when this job should be released to start DMF.

The BSM is activated during system startup before other partitions are started. Therefore, DMF should be activated as early as possible. Because DMF runs in a dynamic partition, the earliest execution time is after VSE/POWER activates the dynamic classes.

Example 2-20 shows how you can establish the DMF to start by using USERBG.PROC.

*Example 2-20 Start DMF from USERBG.PROC*

---

```

* START MODE FOR BG-PARTITION IS NORMAL
* *****
*                                     *
*           YOUR SYSTEM IS           *
// EXEC PROC=SPLEVEL
*                                     *
* *****
// EXEC PROC=LIBDEF
/*
// LIBDEF DUMP,CATALOG=SYSDUMP.BG,PERM
// EXEC ARXLINK                INITIALIZE REXX/VSE
// SETPARM XNCPU=' '
// EXEC PROC=$COMVAR,XNCPU      GET CPU NUMBER
// SETPARM XENVNR=' '
// SETPARM SSLCAUT=' '
// EXEC PROC=CPUVAR&XNCPU,XENVNR,SSLCAUT
PRTY Y,S,R,P,C,BG,FA,F9,F8,F6,F5,F4,F2,F7,FB=Z,F3,F1
// PWR PRELEASE RDR,VTAMSTRT    START VTAM IF REQUIRED
// EXEC IESWAIT,PARM='03'
/*
// PWR PRELEASE RDR,DMFSTART     START DMF IN Z
// EXEC REXX=IESWAITR,PARM='DMFSTART'
/*
// IF SSLCAUT NE YES THEN
// GOTO NOCAUT
// EXEC BSSDCERT,PARM='ACT'
/*
/. NOCAUT
// PWR PRELEASE RDR,CICSICCF     START CICS
// PWR PRELEASE RDR,CEEWARC     START LE AR I/F

```

---

We give the DMF partition Z a priority that is equal to the security server partition FB to prevent the security server or the resource managers in other partitions from having to wait for DMF.

After the job DMFSTART is released, we let partition BG wait for DMF before we release CICS. This configuration should allow CICS to find DMF active when it starts monitoring.

After the next IPL, DMF starts as defined in USERBG.PROC.

Figure 2-36 on page 56 shows how DMF receives the SMF records from the BSM. Depending on whether the resource is defined in the BSM control file or in the DTSECTAB, the paths for the SMF record are different.

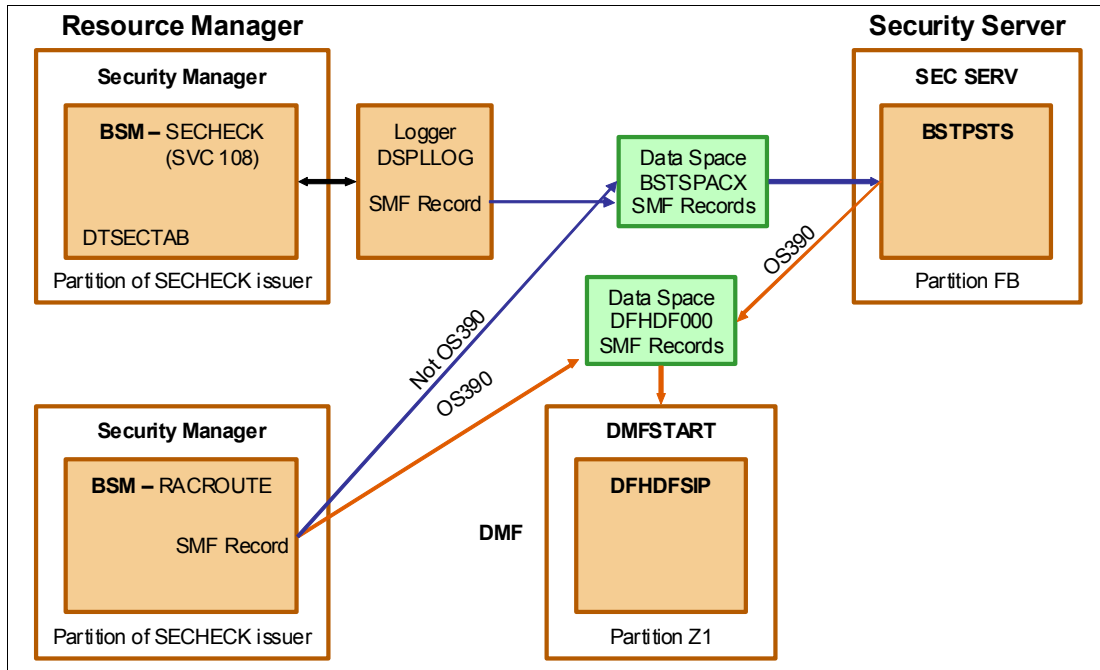


Figure 2-36 SMF record transfer from BSM to DMF

For resources that are defined in the DTSECTAB, the resource manager issues a SECHECK macro for the BSM. The BSM uses the LOG information of the resource profile to decide whether the LOGGER system task should build an SMF record. This task sends the SMF record by way of security server to DMF because it does not run in OS390 mode.

For resources that are defined in the BSM control file, the resource manager issues a RACROUTE call for the BSM<sup>5</sup>. The BSM uses the log information from the RACROUTE request with the AUDIT information of the profile to decide whether an SMF record should be built.

If an SMF record was built, the BSM checks the caller's environment. If the caller is in the OS390 environment, BSM sends the SMF record directly to the DMF data space.

For resource managers that are not in the OS390 environment, the BSM sends the SMF record to the data space of the security server. The security server runs in the OS390 environment and transfers the SMF record to the DMF data space.

<sup>5</sup> The RACROUTE requests can be issued for resource classes DATASET, VSELIB, VSESLIB, and VSEMEM. The BSM might show more SMF records because these records are DTSECTAB resources.



After the DMF data space is filled to a certain level with SMF records, DMF writes the SMF records to the currently open data set, as shown in Figure 2-37.

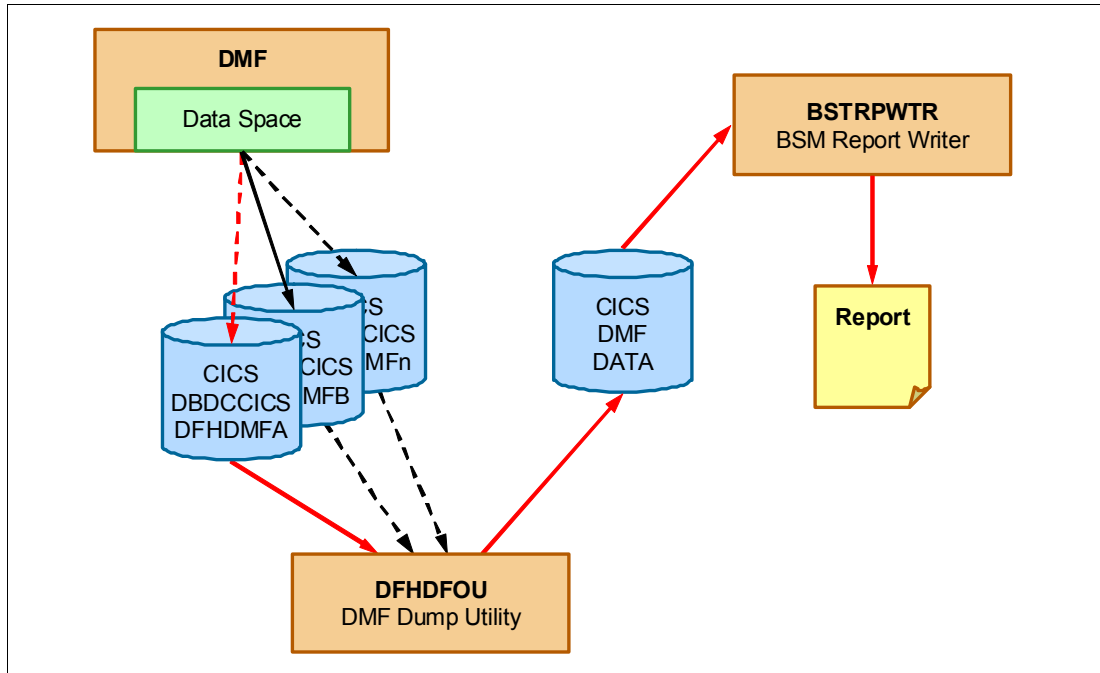


Figure 2-37 SMF record transfer from DMF to BSM report writer

When the open DMF data set is full, DMF closes it and opens another. It ensures that the new data set is empty before attempting to write to it. You must archive and clear the full data set manually by using the DMF dump utility DFHDFOU.

If you do not want to wait until the current data set is full, you can manually switch the data set.

First, you must direct DMF to write any remaining data in the data space to the currently open data set. Use the z/VSE operator command **msg** to send DMF in partition Z1 the command **setdmf flush**, as shown in the following example:

```
msg z1,data=setdmf flush
```

In the next step, you must direct DMF to close the currently open data set, and open the next available data set in the file list. Use the following command:

```
msg z1,data=setdmf switch
```

The data set that was closed is now ready to be processed with the DMF dump utility DFHDFOU.

## 2.4.4 BSM report writer (BSTRPWTR)

To create a BSM report, use DFHDFOU to select the SMF 80 records and store them in a separate data set; for example, CICS.DMF.DATA. Example 2-21 shows a job that combines the extraction of the SMF 80 records with the creation of the BSM security report.

*Example 2-21 Extract SMF80 records from DMF data set and create the report*

---

```

* $$ JOB JNM=BSTRPWTR,CLASS=0,DISP=L
// JOB BSTRPWTR
*   OFFLOAD DMF DATA SET A AND START REPORT
*   Enter ID statement or press enter to continue
// PAUSE
// DLBL VSESPUC,'VSESP.USER.CATALOG',,VSAM
// DLBL INFILE,'CICS.DBDCCICS.DFHDMFA',,VSAM,CAT=VSESPUC
// DLBL OUTFILE,'CICS.DMF.DATA',,VSAM,CAT=VSESPUC,          *
      DISP=(NEW,KEEP),RECORDS=2000,RECSIZE=5750
// LIBDEF *,SEARCH=PRD1.BASE
* Option ALL dumps and clears DMF data set
// EXEC DFHDFOU,SIZE=DFHDFOU
      INDD (INFILE, OPTIONS (ALL))
      OUTDD (OUTFILE, TYPE(80))
/*
// DLBL VSESPUC,'VSESP.USER.CATALOG',,VSAM
// DLBL INPUT,'CICS.DMF.DATA',,VSAM,DISP=(OLD,KEEP),CAT=VSESPUC
// EXEC BSTRPWTR,SIZE=BSTRPWTR
/*
/&
* $$ E0J

```

---

The option ALL of DFHDFOU indicates that the input data set is to be read and copied. After this process, the input data set is to be reset and initialized.

With TYPE(80), you specify that only SMF 80 records should be copied to the output file. In addition to the SMF record type, you can specify date and time as selection criteria with DFHDFOU. For more information, see *CICS Transaction Server for VSE/ESA Operations and Utilities Guide*, SC33-1654.

After SMF 80 records are copied to the data set CICS.DMF.DATA, the BSM report writer program BSTRPWTR uses this data set as input to generate the reports that are described next.

Example 2-22 shows part of a detailed report. Columns Date and Time contain the date and time when the SMF record was sent to DMF.

*Example 2-22 Detailed report*

---

08.075 12:29:29		BSM Report - Listing of Process Records
		E
		v Q
		e u
	*Job/User	n a
Date	Time	t l
08.075	12:25:23	DBDCCICS
		2 0 Job=(CICSICCF) - Resource access: Successful access
		Auth=(Administrator),Reason=(Administrator)
		Resource=CSNE,Intent=Read,Resource class=TCICSTRN
08.075	12:25:24	DBDCCICS
		2 0 Job=(CICSICCF) - Resource access: Successful access
		Auth=(Administrator),Reason=(Administrator)
		Resource=CWBG,Intent=Read,Resource class=TCICSTRN
08.075	12:25:49	DBDCCICS
		2 0 Job=(CICSICCF) - Resource access: Successful access
		Auth=(Administrator),Reason=(Administrator)

```

Resource=CATA,Intent=Read,Resource class=TCICSTRN
08.075 12:25:51 DBDCCICS 2 0 Job=(CICSICCF) - Resource access: Successful access
Auth=(Administrator),Reason=(Administrator)
Resource=CSPQ,Intent=Read,Resource class=TCICSTRN
08.075 12:25:55 SYSA 1 0 Job=(CICSICCF) - User verification: Successful initiation / logon
Auth=(None),Reason=(None)
08.075 12:26:27 SYSA 2 0 Job=(CICSICCF) - Resource access: Successful access
Auth=(Administrator),Reason=(Administrator)
Resource=IESI,Intent=Read,Resource class=TCICSTRN
08.075 12:28:07 SYSA 21 0 Job=(CICSICCF) - ADD command: No violation detected
Auth=(Administrator),Reason=(Class)
ADD FACILITY IBMVSE.JCL.HHTEST UACC(None) AUDIT(FAILURES(Update))
08.075 12:28:11 *DITESYSA 2 1 Job=(DITESYSA) - Resource access: Insufficient authority
Auth=(Normal),Reason=(Audit options)
Resource=DITTO.OTHER.ALL,Intent=Read,Allowed=None,Resource class=FACILITY
08.075 12:28:23 SYSA 19 0 Job=(CICSICCF) - PERMIT command: No violation detected
Auth=(Administrator),Reason=(Class)
PERMIT FACILITY IBMVSE.JCL.HHTEST ID(GROUP02) ACCESS(Update)

```

The column Job/User Name shows the user ID and (if available) the user name. If no user ID was related to the event, the job name is shown instead of a user ID. To indicate that the ID is a job name, the name starts with an asterisk (\*).

Columns Event and Qual contain the numerical value of the event and the related event qualifier. For example, event 2 is a resource access. The event qualifier 0 indicates *successful access*. The description of the numerical values follows in the same line after the job name.

Below this description, you find more information about this event, such as administrator authorization, reason for logging, the resource name and class, intended access, allowed access, or the issued BSTADMIN command.

After the detailed report with the list of processed records, the summary reports follow.

Example 2-23 shows the user summary, a resource summary, and a command summary of the short example.

**Example 2-23 Summary reports**

```

08.075 12:29:29 BSM Report - Listing of User Summary
----- R e s o u r c e S t a t i s t i c s -----
User/   Name      ---- Job/Logon ----      ----- I n t e n t s -----
*Job      Success Violation      Success Violation      Alter      Update      Read      Total
DBDCCICS          0          0          4          0          0          0          4          4
SYSA              1          0          1          0          0          0          1          1
*DITESYSA         0          0          0          1          0          0          1          1

```

```

08.075 12:29:29 BSM Report - Listing of Resource Summary
----- I n t e n t s -----
Resource Name      Success Violation      Alter      Update      Read      Total
Class = FACILITY
DITTO.OTHER.ALL          0          1          0          0          1          1
Class = TCICSTRN
CATA                    1          0          0          0          1          1
CSNE                    1          0          0          0          1          1
CSPQ                    1          0          0          0          1          1
CWBG                    1          0          0          0          1          1
DITT                    1          0          0          0          1          1
IESI                    1          0          0          0          1          1

```

```

08.075 12:29:29 BSM Report - Listing of Command Summary
Qualifier Occurrences

```

```

Event = 19 - PERMIT command
           0 - No violation detected           1
Event = 21 - ADD command
           0 - No violation detected           1
Accumulated totals -                           2

```

---

The user summary report lists the number of successful and failed logon (sign-on) attempts for each user. Also, the resource statistics show the number of successful and failing access attempts to resources and the sum for each access request type. If no user ID was available, the job name is listed with data in the resource statistic.

The resource summary report shows all involved resources sorted by resource class. For each resource, the number of successful and failing access attempts are listed and also the sum per access request type.

The command summary report shows the issued BSTADMIN commands, which are sorted by the event number.

The general summary report, as shown in Example 2-24, provides the number of processed SMF records. It also lists the total number of job or logon attempts and resource access attempts.

*Example 2-24 General summary report*

---

```

08.075 12:29:29                               BSM Report - General Summary

Process records:                               9

                                         --- Job / Logon Statistics ---
Total Job/Logon/Logoff                        1
Total Job/Logon successes                     1
Total Job/Logon violations                    0
Total Job/Logon attempts by undefined users  0
Total Job/Logon successful terminations      0

                                         --- Resource Statistics ---
Total resource accesses (all events)          6
Total resource access successes               5
Total resource access violations              1

```

---

## 2.5 BSM backups

The BSM keeps its security information in the following files:

- ▶ DTSECTAB.PHASE
- ▶ VSE.CONTROL.FILE
- ▶ VSE.BSTCNTL.FILE

For more information about backing up or restoring these files and the z/VSE system and its components, see *z/VSE Operation*, SC33-8309.

The DTSECTAB and the VSE control file existed for a long time in VSE. The BSM control file was introduced with z/VSE V3R1M1. This file is relatively new and has another service to back up and restore it. Therefore, we use this file as an example for backup and restore.

## 2.5.1 VSAM backups

The BSM control file is a VSAM file. You can use the VSAM functions to create copies of this file according to your backup policies. If you want a logical backup, use the REPRO function. If you prefer total replacements, use the BACKUP/RESTORE function of VSAM.

Here, we use the BACKUP/RESTORE approach as example.

To create the backup job that is shown in Example 2-25, you can use fast path 3713 in the administrator dialog.

*Example 2-25 VSAM backup job for the BSM control file*

---

```
* $$ JOB JNM=VSMBKUP1,DISP=D,PRI=3, C
* $$ NTFY=YES, C
* $$ CLASS=0
* $$ LST DISP=H
// JOB VSMBKUP1
// LIBDEF PHASE,SEARCH=IJSYSRS.SYSLIB
* THIS JOB BACKS UP VSAM DATASETS
// DLBL IJSYSUC,'VESP.USER.CATALOG',,VSAM
*
* THIS FUNCTION USES A TAPE FOR OUTPUT
* MOUNT LABELED TAPE WITH VOLUME ID=DDDDDD ON DEVICE 181
* THEN CONTINUE. IF NOT POSSIBLE CANCEL THIS JOB.
// PAUSE
// TLBL VSMBKUP,'BSTCNTL.BACKUP1',,'DDDDDD'
// ASSGN SYS005,181
// EXEC IDCAMS,SIZE=AUTO
      BACKUP ( -
              VSE.BSTCNTL.FILE -
            ) -
      UNLD -
      NOCOMPACT -
      BUFFERS(3) -
      STDLABEL(VSMBKUP)
/*
/*
/&
* $$ EOJ
```

---

Before you submit this job, ensure that the BSM control file is not open in any partition.

Normally, the CICS partition and the security server partition FB use this file. To close it in CICS, issue the transaction CEMT SET FI(BSTCNTL) CLO. For the security server, use the console command **msg fb,data=closebst**. Then, submit your job to build a backup on tape.

After your backup job successfully completes, reopen the BSM control file in FB by using the **msg fb,data=openbst** command; for CICS, use the transaction CEMT SET FI(BSTCNTL) OPE.

Example 2-26 shows the restore job of the VSAM backup. You can use fast path 3714 in the administrator dialog to create a similar VSAM restore job.

*Example 2-26 VSAM restore job for the BSM control file*

---

```

* $$ JOB JNM=VSMREST1,DISP=D,PRI=3, C
* $$ NTFY=YES, C
* $$ CLASS=0
* $$ LST DISP=H
// JOB VSMREST1
// LIBDEF PHASE,SEARCH=IJSYSRS.SYSLIB
* THIS JOB RESTORES VSAM DATASETS
*
* THIS FUNCTION USES A TAPE FOR INPUT
* MOUNT LABELED TAPE WITH VOLUME ID=DDDDDD ON DEVICE 181
* THEN CONTINUE. IF NOT POSSIBLE CANCEL THIS JOB.
// PAUSE
// TLBL VSMREST,'BSTCNTL.BACKUP1',,'DDDDDD'
// ASSGN SYS004,181
// EXEC IDCAMS,SIZE=AUTO
    RESTORE OBJECTS ( -
        * -
        ) -
        CATALOG(VSESP.USER.CATALOG) -
        STDLABEL(VSMREST) -
        VOLUMES(DOSRES) -
        UNLD -
        BUFFERS(3)
/*
/*
/&
* $$ EOJ

```

---

## 2.5.2 BSM backup and migration with BSTSAVER

As opposed to a VSAM backup, the batch program BSTSAVER builds BSTADMIN commands from the contents of the BSM control file and stores these BSTADMIN commands in a library member.

You can use this member as a backup of your BSM control file in readable text format, and use it to migrate the contents of your BSM control file to your current z/VSE release. For example, use it when migrating your BSM security to z/VSE V4R2 from:

- ▶ z/VSE V3R1M1 or a later refresh of z/VSE V3R1
- ▶ z/VSE V4R1 or a later refresh of z/VSE V4R1

Example 2-27 shows the BSTSAVER call in a batch job. You can also call BSTSAVER from the console by using PAUSEBG. The parameter specifies the library member where BSTSAVER should store the generated BSTADMIN commands. In the example, we use:

```
'PRD2.SAVE.BSTCNTL.SAVE10'
```

*Example 2-27 BSTSAVER start in batch job*

---

```

* $$ JOB JNM=BSTSAVER,CLASS=0,DISP=D
// JOB BSTSAVER
*     Save BSTCNTL as commands in PRD2.SAVE.BSTCNTL.SAVE10

```

---

```
*   Enter ID statement or press enter to continue
// PAUSE
// EXEC BSTSAVER,PARM='PRD2.SAVE.BSTCNTL.SAVE10'
/*
/&
* $$ E0J
```

---

Later, we can use this member to rebuild the definitions in the BSM control file with a BSTADMIN call in a batch job, as shown in Example 2-28.

*Example 2-28 Rebuild BSM control file contents with BSTADMIN*

---

```
* $$ JOB JNM=BSTADMIN,CLASS=0,DISP=D
// JOB BSTADMIN
*       Execute BSTADMIN commands specified in library member
*   Enter ID statement or press enter to continue
// PAUSE
// EXEC BSTADMIN,OS390
* $$ SLI MEM=BSTCNTL.SAVE10,S=PRD2.SAVE
/*
/&
* $$ E0J
```

---

This rebuild does not delete definitions in the BSM control file. If a profile already exists, the related ADD, ADDGROUP, and CONNECT commands fail. The PERMIT command replaces an existing permission for a user or group on the access list. The security settings from 'PRD2.SAVE.BSTCNTL.SAVE10' replace existing security settings.

If the rebuild was successful, you can activate the profile updates with the BSTADMIN command **PERFORM DATASPACE REFRESH**.







## LDAP sign-on support

In this chapter, we introduce Lightweight Directory Access Protocol (LDAP) sign-on support on z/VSE.

This chapter includes the following topics:

- ▶ 3.1, “LDAP and z/VSE” on page 66
- ▶ 3.2, “Risks of the current situation” on page 67
- ▶ 3.3, “LDAP terminology” on page 68
- ▶ 3.4, “z/VM LDAP server” on page 72
- ▶ 3.5, “LDAP sign-on of z/VSE” on page 72
- ▶ 3.6, “Configure and activate LDAP sign-on support” on page 75
- ▶ 3.7, “Administering the LDAP user mapping file” on page 83
- ▶ 3.8, “LDAP sample setup” on page 87

## 3.1 LDAP and z/VSE

z/VSE V4R2 introduces a new z/VSE LDAP client that can be used to perform authentication against a remote LDAP server. z/VSE V4R2 does not provide an LDAP server.

The LDAP sign-on support enables users to sign on to z/VSE by using long, *company-wide* (corporate) user IDs and passwords. The user ID and password are authenticated by using an LDAP server that is reachable through the TCP/IP network. The LDAP server is installed on the company network.

This use of company-wide user IDs connects z/VSE with the centralized management of user IDs. As a result, password rules and password renewal can be enforced through the LDAP server.

LDAP authorization is designed to integrate z/VSE into identity management systems, such as IBM Tivoli® products.

An LDAP sign-on solution overcomes the previous limitations in z/VSE that:

- ▶ VSE/ICCF user IDs were up to four characters long.
- ▶ z/VSE user IDs were 4 - 8 characters long.
- ▶ Passwords were up to eight characters long.

Consider the following points regarding an LDAP sign-on solution:

- ▶ User IDs and passwords can be up to 64 characters long.
- ▶ Users can be forced to use complex passwords. For example, passwords might have to contain a mixture of numeric and alphabetic characters, and uppercase and lowercase characters.
- ▶ A centralized management of company-wide user IDs and passwords within a company is possible. Each user can be forced to change their password that is used on all systems within a predefined time period; for example, every 12 weeks.

The major advantage of a centralized identity management system is that all updates for employees can be performed on one location, including enabling or disabling user IDs and granting or revoking access.

For example, if an employee leaves a company, the company's administrator must revoke access only for the person's user IDs in the identity management system; that is, typically the LDAP server. Therefore, the revoked user IDs can no longer be used for signing on to z/VSE without the requirement for the z/VSE administrator to perform any further actions.

## 3.2 Risks of the current situation

The identity management or user ID management often is separated between the host systems and the distributed systems, such as Windows, UNIX, and Linux (see Figure 3-1).

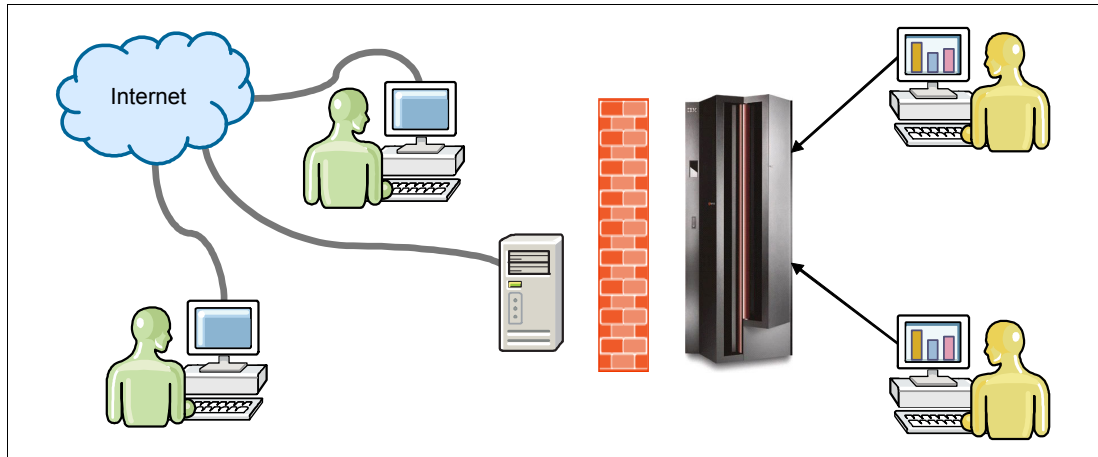


Figure 3-1 Separated identity management

The z/VSE and z/VM administrators manage the process of defining user IDs and appropriate access rights by using the tools that are available on z/VSE or z/VM. The security administrators of the distributed site manage the user IDs on the distributed site by using the available tools.

However, most z/VSE users also use distributed systems on other sites. Therefore, they have one or multiple user IDs on z/VSE, and in addition one or multiple user IDs on the distributed site. This issue results in one person having multiple user IDs on different systems. Normally, no automatic synchronization exists between the user IDs. If a change is made to one user ID on one system, the other user IDs on the other systems also must be changed manually. This manual task can be time-consuming and error-prone.

In many cases, the user ID does not easily tell you who is its owner; for example, user ID FRAN. Sometimes user IDs consist of the first few characters from the last name, an identification number, or the email address. In any case, documentation must exist to know what user IDs are owned by whom.

Corporate policies or rules can exist that define how a user ID or its password must be formed. Policies usually control how often a password must be changed, how many characters it must consist of, and how complex it must be (how many special characters, how many numbers, and so on). Different policies or rules can exist on different systems. Certain systems might not even allow enforcing certain policies.

Various security risks can be exposed if identity management is not performed carefully, including the following examples:

- ▶ If an employee leaves the company, you must deactivate all of the employee's user IDs on all affected systems. If you miss one, the person, or other persons that know that employee's password might still be able to use the system.
- ▶ If an employee moves to another department or job role, the permissions to access systems and resources on the systems might have to be adapted according to their new job role. If you miss adjusting a user ID on a server, the employee might still be able to access data that the employee should no longer be allowed access.

To reduce the risk of missing to update a user ID, use a centralized identity management system that stores and manages all user IDs for all systems. This approach enables you to have one single place where all identity-related information is stored, including user IDs, passwords, permissions, groups, roles, and more. At best, you have only one user ID per person, and all surrounding systems can work with that user ID.

With having such a centralized identity management system, the risk of missing a user ID when changing a system can be greatly reduced. All changes to user IDs automatically affect all systems, without any extra manual intervention. Corporate security policies can also be enforced much easier by a central identity management system.

Common security tasks, such as resetting a password, unlocking a user ID, and others can easily be provided through an automated self-service help desk, which reduces any manual intervention to a minimum.

Most of today's identity management systems use an LDAP server for storing identity-related information (see Figure 3-2).

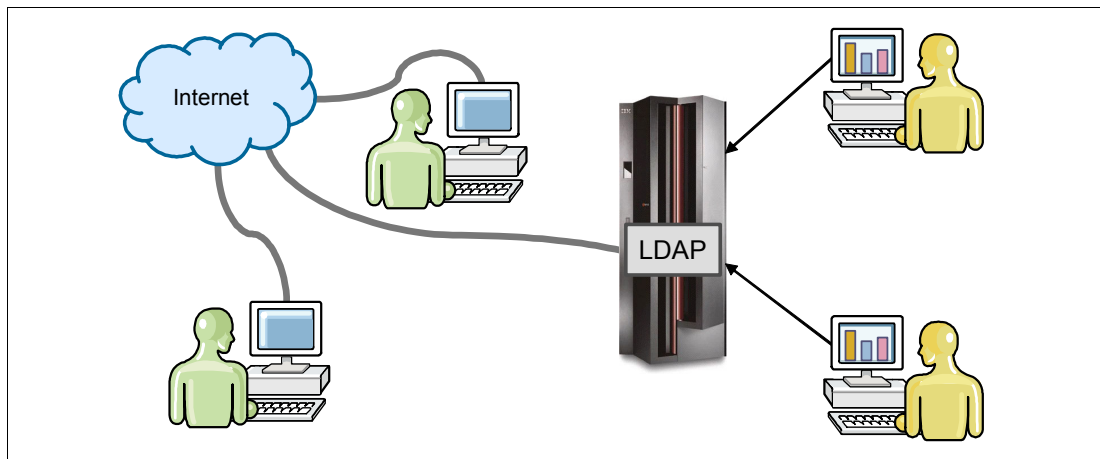


Figure 3-2 Identity management system using an LDAP server

## 3.3 LDAP terminology

When you start to use an LDAP server and clients, understanding several terms and processes is helpful.

### 3.3.1 Overview and terms

LDAP defines the content of messages that are exchanged between an LDAP client and an LDAP server. The messages specify the operations that are requested by the client (for example, search, modify, and delete), the responses from the server, and the format of data that is carried in the messages. Because LDAP messages are carried over TCP/IP (a connection-oriented protocol), operations are available to establish and disconnect a session between the client and server.

An overview of the main LDAP actors is shown in Figure 3-3.

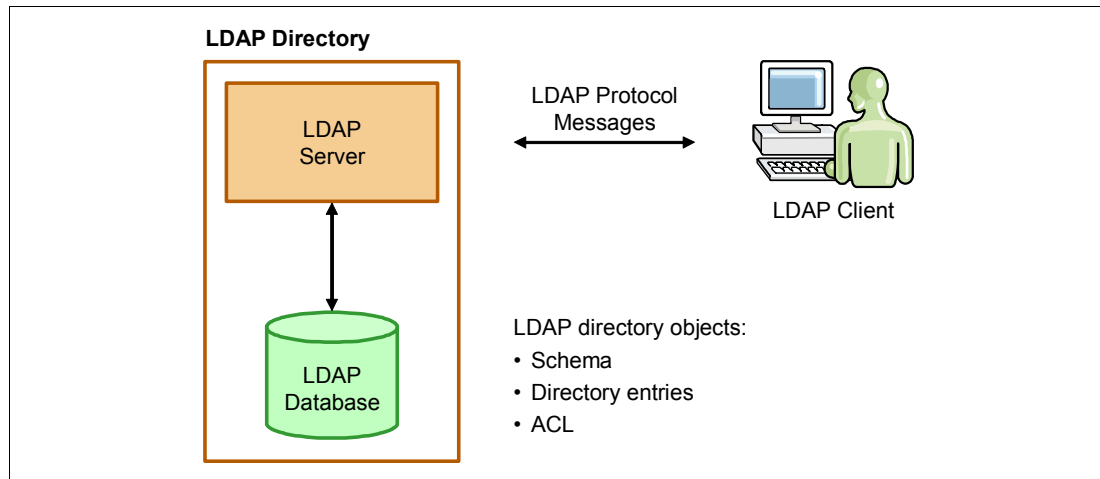


Figure 3-3 LDAP overview

However, for the designer of an LDAP directory, the structure of the messages that are sent and received over the wire are not of much interest. More important is the logical model that is defined by these messages and data types, how the directory is organized, what operations are possible, how information is protected, and so on.

The general interaction between an LDAP client and an LDAP server includes the following process:

1. The client establishes a connection with an LDAP server. This process is known as *binding* to the server. The client specifies the host name or IP address and TCP/IP port number where the LDAP server is listening.
2. The client can provide a user name and a password to authenticate properly with the server, or the client can establish an anonymous session with default access rights. The client and server can also establish a session that uses stronger security methods, such as encryption of data.
3. The client then performs operations on directory data. LDAP offers read and update capabilities, which allows directory information to be managed and queried. LDAP also supports searching the directory for data meeting arbitrary user-specified criteria.

Searching is a common operation in LDAP. A user can specify what part of the directory to search and what information to return. A search filter that uses boolean conditions specifies what directory data matches the search.

4. When the client is finished making requests, it closes the session with the server. This process is also known as *unbinding*.

The philosophy of the LDAP API is to keep simple things simple, which means that adding directory support to existing applications can be done with low overhead. Because LDAP was originally intended as a lightweight alternative to DAP for accessing X.500 directories, it follows an X.500 model.

The directory stores and organizes data structures that are known as *entries*. A directory entry often describes an object, such as a person, device, or location, as shown in Figure 3-4.

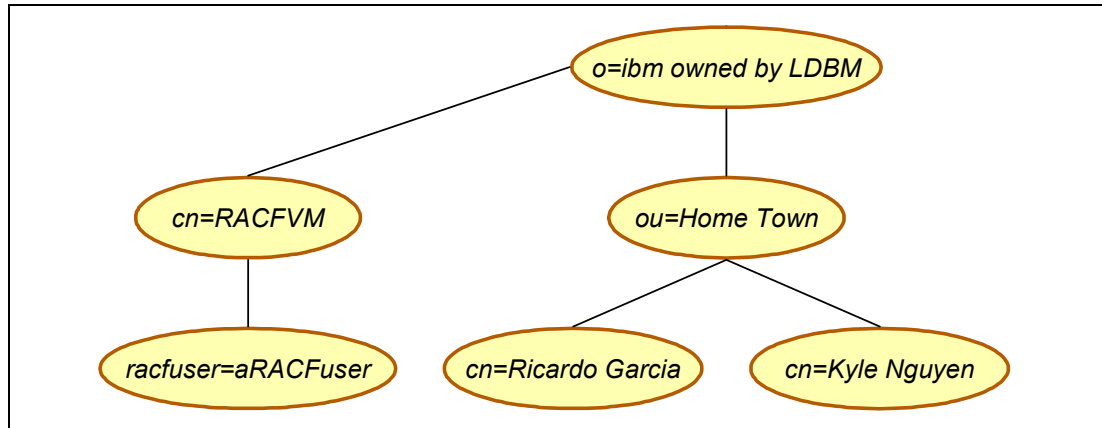


Figure 3-4 LDAP directory

Each entry includes a *distinguished name* (DN) that uniquely identifies it. The DN consists of a sequence of parts that is called relative distinguished names (RDNs), much like a file name consists of a path of directory names in many operating systems, such as z/OS, z/VM, Linux, UNIX, and Windows. The entries can be arranged into a hierarchical tree-like structure that is based on their distinguished names. This tree of directory entries is called the *Directory Information Tree* (DIT).

Each entry contains one or more attributes that describe the entry. Each attribute includes a type and a value. For example, the directory entry for a person might have an attribute that is named *telephoneNumber*. The syntax of the *telephoneNumber* attribute specifies that a telephone number must be a string of numbers that can contain spaces and hyphens. The value of the attribute is the person's telephone number, such as 800-555-1234.

A directory entry describes a particular object. An object class is a general description (sometimes called a template) of an object, as opposed to the description of a particular object. For example, the object class *person* has a *surname* attribute, whereas the object describing John Smith has a *surname* attribute with the value Smith.

The object classes that a directory server can store and the attributes they contain are described by schema. Schema define what object classes are allowed where in the directory, what attributes they must contain, what attributes are optional, and the syntax of each attribute. For example, a schema might define a *person* object class. The *person* schema might require that a person have a *surname* attribute that is a character string, specify that a person entry can optionally have a *telephoneNumber* attribute that is a string of numbers with spaces and hyphens, and so on.

## Object classes

An object class is an LDAP term that denotes the type of object that is represented by a directory entry or record. Several typical object types are *person*, *organization*, *organizational unit*, *domain component*, and *groupOfNames*. Also, object classes are available that define an object's relationship to other objects, such as object class *top* denotes that the object can have subordinate objects under it in a hierarchical tree structure. Some LDAP object classes can be combined; for example, an object class of *organizational unit* is most often also simultaneously defined as a *top* object class because it has entries beneath it.

## Attributes

The only thing an object class does is define the attributes, or types of data items that are in that type of object. Examples of typical attributes include cn (common name), sn (surname), givenName, mail, UID, and userPassword. Just as the object classes are defined with unique OIDs, each attribute also has a unique OID number that is assigned to it. LDAP V3 attributes follow a notation similar (ASN.1) to object classes.

## LDAP operations

LDAP defines operations for accessing and modifying directory entries, such as:

- ▶ Binding and unbinding
- ▶ Searching for entries meeting user-specified criteria
- ▶ Adding an entry
- ▶ Deleting an entry
- ▶ Modifying an entry
- ▶ Modifying the distinguished name or relative distinguished name of an entry (move)
- ▶ Comparing an entry

All directory servers use version 3 of LDAP.

LDAP V3 is documented in several IETF RFCs (2251, 2252, 2253, 2254, 2255, and 2256).

### 3.3.2 LDIF files

LDAP Data Interchange Format (LDIF) is used as input or output file during LDAP server commands. When an LDAP directory is loaded for the first time or when many entries must be changed at once, changing each single entry on a one-by-one basis is inconvenient. For this purpose, LDAP supports the LDAP Data Interchange Format (LDIF) that can be seen as a convenient, yet necessary, data management mechanism. It enables easy manipulation of mass amounts of data.

Figure 3-5 shows an LDAP server export followed by an import to another LDAP server.

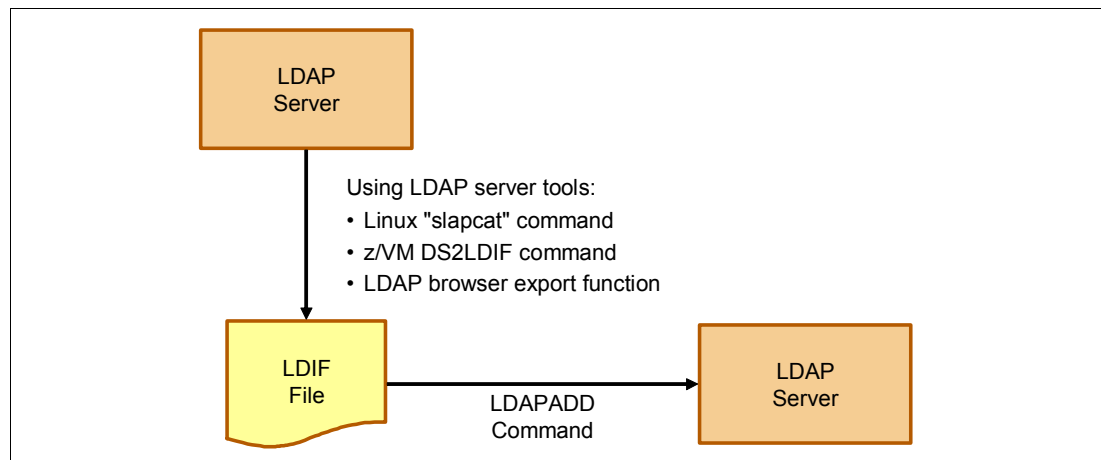


Figure 3-5 Export and import LDAP database

Depending on the client operating system and the server operating system, different tools are used to export or import LDAP server data. For example, on Linux, the main tool is slapcat. It is available in the openldap2 package. On Windows, tools, such as LDAP Browser/Editor by Jarek Gavor, includes a database export/import tool. With z/VM, DS2LDIF tools can also be used to copy an LDAP directory to a CMS files.

### 3.4 z/VM LDAP server

In addition to many other LDAP servers that are available as a commercial product or open source, z/VM provides an LDAP server that runs on z/VM. The z/VM LDAP server is a ported version of the IBM Tivoli Directory Server for z/OS.

Many z/VSE clients run their z/VSE systems as guests under z/VM. Therefore, choosing z/VM to host the LDAP server is a good choice.

The z/VM LDAP server provides different back ends, including the SDBM back end. The Secure Database Manager (SDBM) gives access to the data that is stored in the IBM z/VM RACF® database. Its main function is the directory authentication (bind) that is based on the user ID and the password. By using the SDBM back end, you can authenticate z/VSE users against the z/VM RACF database.

For more information about z/VM LDAP support, see *Security on z/VM*, SG24-7471.

### 3.5 LDAP sign-on of z/VSE

The z/VSE LDAP sign-on support is part of z/VSE since z/VSE V4R2. It consists of an LDAP client implementation that can connect to an LDAP server using TCP/IP. For security reasons, the communication between the LDAP client and the LDAP server can be encrypted by using Secure Sockets Layer (SSL) or Transport Layer Security (TLS). In addition, z/VSE provides a special sign-on program that performs the authentication through the LDAP server.

Figure 3-6 shows the modules that are part of the z/VSE LDAP sign-on support.

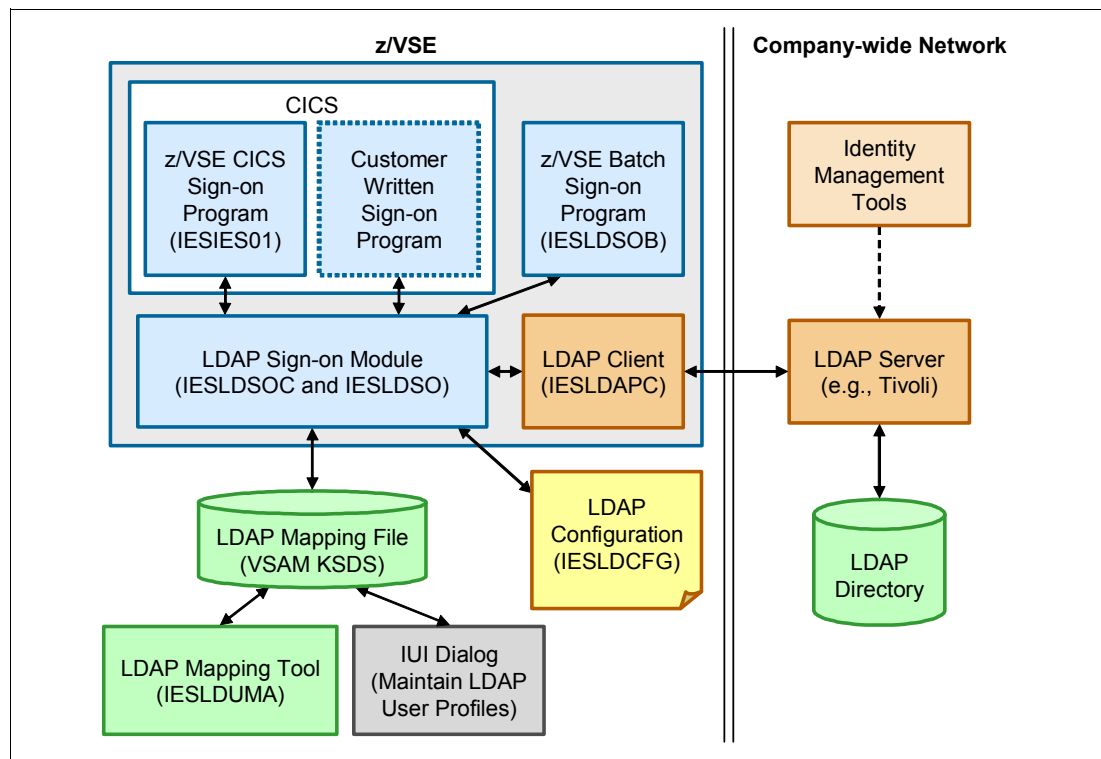


Figure 3-6 Components of the z/VSE LDAP sign-on support



The LDAP sign-on support sits on top of any z/VSE security manager. Therefore, it can be used with the BSM and with an external security manager, such as CA-Top Secret from Computer Associates or Alert from Connectivity Systems Inc. (CSI).

In principle, the sign-on process consists of the following steps:

1. The LDAP sign-on program connects to the LDAP server and authenticates by using the user ID and password that is entered by the user (both up to 64 characters) by performing LDAP bind and search operations.
2. If the LDAP authentication was successful, it looks up the user's record in the LDAP user mapping file and gets the associated z/VSE user ID and password (both up to eight characters) from the record. The LDAP mapping file is a KSDS VSAM cluster that contains the mapping between the long LDAP user IDs and the short VSE user IDs.
3. The LDAP sign-on program passes the corresponding z/VSE user ID and password to the underlying security manager (BSM or ESM) through the sign-on process (for example, EXEC CICS SIGNON).

The use of LDAP authentication enables users to sign on z/VSE by using a single, comprehensive, corporate-wide identity management system; for example, the IBM Tivoli Identity Manager. Any changes to a user ID done in the identity management system automatically affect z/VSE sign-on also. This approach allows for easily enforced corporate policies and rules, such as password renewal, password complexity rules, consistent access rules, and ease-of-use for basic users.

All security definitions regarding permissions and access to resources are done on z/VSE based on the associated short z/VSE user ID. If the security setup exists, it can be used without any change. However, the user signs on into z/VSE by using their long LDAP user ID and password.

The z/VSE LDAP sign-on support allows user IDs and passwords to be up to 64 characters. LDAP user IDs are usually case-sensitive.

User IDs or passwords on z/VSE must meet the following requirements:

- ▶ Up to four characters for VSE/ICCF user IDs
- ▶ A range of 4 - 8 characters for CICS user IDs
- ▶ Up to eight characters for the passwords

If both CICS and ICCF are used, user IDs of exactly four characters often result.

By using the z/VSE LDAP sign-on support, you integrate z/VSE into a corporate-wide identity management system, as shown in Figure 3-7 on page 74.

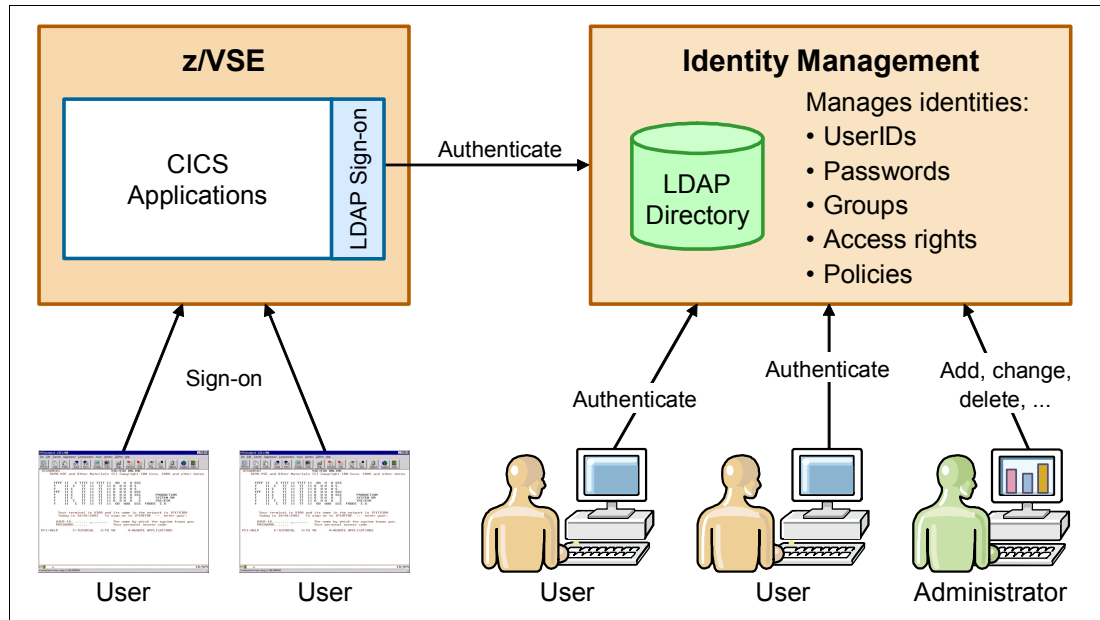


Figure 3-7 Identity management

### 3.5.1 LDAP user mapping file

To associate a long LDAP user ID with a short z/VSE user ID, the LDAP sign-on support uses an LDAP user mapping file. This mapping file is a VSAM KSDS cluster.

The mapping file contains records for users that are:

- ▶ LDAP-enabled

These user IDs are mapped to corresponding short z/VSE user IDs.

- ▶ Not LDAP-enabled

These user IDs are not mapped to short z/VSE user IDs; instead, these user IDs are passed directly to the underlying security manager without performing LDAP authentication. These users can sign on to z/VSE, even if the LDAP server is not operational.

A good practice is to configure the system administrators (such as SYSA) and possibly the operators to the *not LDAP-enabled* ID type, so that the users can sign on to z/VSE even if the LDAP server is not reachable, if the network is down, or if another problem does not allow z/VSE to perform LDAP authentication.

The LDAP user mapping file is maintained with a batch tool called IESLDUMA. You can add user IDs to the mapping file, change or delete user IDs, and list all user IDs that are contained in the mapping file. The IESLDUMA tool can be used by only an administrator type of users. If batch security is active in z/VSE, you specify this administrator user ID with the JCL ID statement before IESLDUMA is run. If batch security is not active, you must specify the administrator user ID by using the ID command of IESLDUMA, as the first command after IESLDUMA is started.

## 3.5.2 Strict mode

You can configure the LDAP sign-on support to operate in one of the following modes:

- ▶ Strict mode

In this mode, all users (those users who use a long user ID and those users who use a short user ID) are defined in the LDAP mapping file.

- ▶ Non-strict mode

In this mode, only those users who use a long user ID are defined in the LDAP mapping file. Sign-on requests for short user IDs are processed directly, if this short user ID is accepted by the security manager; for example, by the BSM. The advantage of using non-strict mode is that users who should be able to sign on to z/VSE when the LDAP server is not operational (for example, the SYSA user ID) do not have to be defined in the LDAP mapping file.

## 3.5.3 LDAP password cache

Performing authentication with a remote LDAP server can be time-consuming. It requires a network connection to the LDAP server to be established and several LDAP operations to be run. When the same user signs on to z/VSE within a short period, the LDAP password or other settings of the user ID in the LDAP directory are unlikely to change. Therefore, the z/VSE LDAP sign-on support provides an option for the use of a cache to store a secure hash (SHA-256) of the last successfully used LDAP password.

If the user signs on within a certain time, the password that is specified by the user is checked against the hash that is stored in the cache instead of performing a full authentication with the remote LDAP server. Such a cached password features an expiration period. You can configure after how many seconds the cache should expire. After the cache entry is expired, a full LDAP authentication is performed.

The use of the password cache is optional; you can also turn it off. The cache never stores the LDAP password; instead, a secure hash (SHA-256) is built from the password and stored in the cache entry. The password cannot be recovered from its hash because the hash function is a one-way function.

## 3.6 Configure and activate LDAP sign-on support

To use LDAP sign-on support with your z/VSE system, you must configure and activate it first. By default, LDAP sign-on support is not activated.

To activate LDAP sign-on support, create a configuration and specify several settings. These settings include the name of the user mapping file, the IP addresses or host names of the LDAP servers, settings for the LDAP authentication process, and settings for Secure Sockets Layer.

To configure LDAP sign-on support, you use the skeleton SKLD CFG in ICCF library 59. Note that certain values in that configuration are case-sensitive. You must switch to mixed case mode in the ICCF editor by using CASE M when editing this member.

### 3.6.1 LDAP configuration example skeleton

The skeleton that is shown in Example 3-1 generates a phase that is called IESLDCFG.PHASE, which is stored in PRD2.CONFIG. The sections of the example are discussed after the example.

*Example 3-1 LDAP configuration*

```
* $$ JOB JNM=LDCONFIG,CLASS=A,DISP=D
// JOB LDCONFIG GENERATE LDAP SIGNON CONFIG PHASE
* *****
* ASSEMBLE AND LINK THE LDAP CONFIG PHASE *
* *
* NOTE: THE CONTENTS OF THIS MEMBER IS CASE SENSITIVE ! *
* *****
// LIBDEF *,CATALOG=PRD2.CONFIG
// LIBDEF *,SEARCH=PRD1.BASE
// OPTION ERRS,SXREF,SYM,NODECK,CATAL,LISTX
    PHASE IESLDCFG,*,SVA
// EXEC ASMA90,SIZE=(ASMA90,64K),PARM='EXIT(LIBEXIT(EDECKXIT)),SIZE(MAXC
    -200K,ABOVE)'

IESLDCFG CSECT
IESLDCFG AMODE ANY
IESLDCFG RMODE ANY
* *****
* GENERAL SETTINGS:
* *****
*
* FLAGS. POSSIBLE VALUES SEE EQUATES BELOW
*
FLAGS                DC      XL4'00000001'
LDAP_AUTH_ENABLED    EQU     X'00000001'   LDAP AUTH IS ENABLED
USE_SSL              EQU     X'00000002'   SSL IS ENABLED
STRICT_MODE          EQU     X'00000004'   STRICT USER MAPPING
FAILURE_HASH         EQU     X'00000008'   FAILURE PASSWORD HASH USED
UPPERCASE_MODE       EQU     X'00000010'   UPPERCASE LDAP USERID & PWD
EBCDICCP_PER_TERM    EQU     X'00000020'   USE TERM.TABLE FOR CODEPAGE
SIGNON_MSG_SYSLOG    EQU     X'01000000'   SIGNON MESSAGES TO SYSLOG
SIGNON_MSG_SYSLST    EQU     X'02000000'   SIGNON MESSAGES TO SYSLST/TDQ
TRACE                EQU     X'80000000'   ENABLE TRACING
*
* USER MAPPING FILE NAME (DLBL)
*
USER_MAP_FILE_DLBL   DC      CL8'IESLDUM'
*
* EBCDIC CODEPAGE (IMPORTANT FOR @ CHAR). THE LDAP PROTOCOL USES UTF-8
* FOR DATA TRANSFER. THE LDAP CLIENT TRANSLATES THIS TO THE SPECIFIED
* EBCDIC CODEPAGE. (DEFAULT IS IBM-1047).
*
EBCDIC_CODEPAGE      DC      CL16'IBM-1047'
*
* VALIDITY PERIOD IN MINUTES FOR CACHING THE LDAP PASSWORD.
* TO DISABLE CACHING OF LDAP PASSWORDS SET THIS VALUE TO ZERO.
*
CACHE_EXPIRATION     DC      F'0'
*
* *****
* LDAP SERVER SETTINGS
* *****
*
* LDAP SERVER IP OR HOSTNAME.
```

```

*
* SPECIFICATION CONTAINS THE IP ADDRESS OR HOSTNAME OF ONE OR MORE
* LDAP SERVERS IN THE FORM <SERVER>:<PORT>.
* MULTIPLE SERVER ADDRESSES ARE SEPARATED BY BLANKS.
* IF THE PORT NUMBER IS OMITTED, THE DEFAULT PORT NUMBERS ARE USED:
*   389 - NON-SSL
*   636 - SSL
* PLEASE NOTE THAT MICROSOFT ACTIVE DIRECTORY MAY USE DIFFERENT PORT
* NUMBERS. THE AD GLOBAL CATALOG SERVER USES PORT 3268 PER DEFAULT.
*
LDAP_SERVERS      DC      CL256'my.ldap.server:389'
*
* *****
* SSL SPECIFIC SETTINGS. IF SPECIFIED, A SSL CONNECTION IS USED TO
* CONNECT TO THE LDAP SERVER.
* *****
*
* KEYRING LIBRARY
*
KEYRING_LIBRARY  DC      CL16'CRYPTO.KEYRING'
*
* NAME OF THE KEY MATERIAL IN THE KEYRING LIBRARY
*
KEYNAME          DC      CL8'LDAPKEY'
*
* SSL CIPHER SPECS TO USE
* SPECIFY ALL BLANKS TO USE THE SSL LIBRARY'S SUPPORTED CIPHERS
*
CIPHER_SPEC      DC      CL64'352F' 0A 09 08 02 01 ARE DEPRECATED
* FOR OPENSSL:
* CIPHER_SPEC      DC  CL64'C027C014C013C0126B673933163D3C352F'
*
* SSL SESSION TIMEOUT IN SECONDS
*
SESSION_TIMEOUT  DC      F'86400'
*
* *****
* LDAP AUTHENTICATION SETTINGS
* *****
*
* AUTHENTICATION METHOD:
* 1.) DIRECT: THE LDAP USER ID IS USED DIRECTLY AS THE USER NAME
*           PASSED TO BIND. THE DN_BIND_PATTERN BELOW IS USED TO
*           BUILD THE DISTINGUISHED NAME FOR BIND.
* 2.) SEARCH: THE LDAP USER ID IS NOT USED DIRECTLY FOR BIND.
*           INSTEAD A SEARCH IS PERFORMED FOR THE USER ID USING
*           A SPECIFIC ATTRIBUTE. THE DISTINGUISHED NAME OF THE
*           SEARCH RESULT IS USED TO PERFORM THE BIND.
*
AUTH_METHOD      DC      F'2'
AUTH_DIRECT      EQU     1      USE DIRECT BIND WITH USER ID
AUTH_SEARCH      EQU     2      USE SEARCH ON ATTRIBUTE
*
* DISTINGUISHED NAME PATTERN. AN OCCURANCE OF '%u' IS REPLACED
* WITH THE USER NAME.
*
DN_BIND_PATTERN  DC      CL64'cn=%u,dc=mycompany,dc=com'
*
* DISTINGUISHED NAME USED FOR BIND WHEN PERFORMING THE SEARCH.
* LEAVE IT BLANK FOR ANONYMOUS BIND

```

```

*
BIND_DN          DC    CL64''
*
* PASSWORD USED FOR BIND WHEN PERFORMING THE SEARCH.
* LEAVE IT BLANK FOR ANONYMOUS BIND
*
BIND_PWD         DC    CL64''
*
* USER ID ATTRIBUTE NAME USED WHEN PERFORMING THE SEARCH.
*
USER_ATTRIBUTE   DC    CL64'emailaddress'
*
* BASE DISTINGUISHED NAME USED WHEN PERFORMING THE SEARCH.
*
BASE_DN          DC    CL64'ou=myorgunit,o=mycompany.com'
*
* THE DEREFERENCING OPTION USED WHEN DOING THE SEARCH
*
SEARCH_DEREF     DC    F'0'
DEREF_NEVER      EQU   0      Deref NEVER
DEREF_SEARCHING  EQU   1      Deref SEARCHING
DEREF_FINDING    EQU   2      Deref FINDING
DEREF_ALWAYS     EQU   3      Deref ALWAYS
*
* THE SCOPE USED WHEN DOING THE SEARCH
*
SEARCH_SCOPE     DC    F'2'
SCOPE_BASE       EQU   0      SEARCH THE OBJECT ITSELF
SCOPE_ONELEVEL   EQU   1      SEARCH THE OBJECT'S IMMEDIATE CHILDREN
SCOPE_SUBTREE    EQU   2      SEARCH THE OBJECT AND ALL ITS DESCENDANTS
*
* ADDITIONAL SEARCH FILTER USED WHEN PERFORMING THE SEARCH.
* LEAVE IT BLANK IF NO ADDITIONAL FILTER IS REQUIRED
*
ADD_SEARCH_FILTER DC    CL128''
*
* THE TIMEOUT OPTION USED WHEN DOING THE SEARCH. A VALUE
* OF 0 MEANS NO TIME LIMIT. THE TIMEOUT IS SPECIFIED IN SECONDS.
*
SEARCH_TIMEOUT   DC    F'0'
*
*
* VALIDITY PERIOD IN MINUTES FOR CACHING THE LDAP PASSWORD. THIS
* PERIOD IS USED WHEN THE LDAP SERVER CANNOT BE REACHED.
* TO DISABLE FAILURE CACHING OF LDAP PASSWORDS SET THIS VALUE TO ZERO.
* TO ENABLE FAILURE CACHING; YOU ALSO NEED TO SET THE BIT FAILURE_HASH
* IN THE FLAGS FIELD.
*
FAILURE_CACHE_EXPI DC    F'0'
*
* MESSAGE DESTINATION (TRANSIENT DATA QUEUE) FOR THE SIGNON MESSAGES.
*
MSGSD_QUEUE      DC    CL4'CSML'
*
* *****
* ADDITIONAL SSL SPECIFIC SETTINGS.
* *****
*
* SSL SECURITY TYPES, E.G. 'SSL30' or 'TLSV1'
* OPENSSL ALSO SUPPORTS 'SSLV3', 'TLS31', 'TLSV1.2' OR 'ALL'.

```

```

* CSI ALSO SUPPORTS 'TLS12'.
*
SSL_SEC_TYPES      DC      CL16'TLSV1'
*
* *****
* SUPPORT FOR CONFIGURING THE EBCDIC CODEPAGE PER TERIMAL ID AND/OR
* NETNAME:
* IF USERS LOGON THROUGH TERMINALS WITH DIFFERENT CODEPAGE SETTINGS
* YOU MAY WANT TO CONFIGURE A CORESPONDING EBCDIC CODEPAGE FOR THE
* LDAP SIGNON SUPPORT. YOU CAN CONFIGURE THE EBCDIC CODEPAGE BASED
* ON THE 4 CHARACTER TERMINAL ID AND/OR THE 8 CHARACTER NETNAME.
* FOR BOTH YOU CAN USE PATTERS WITH WILDCARDS ('*' AND '?').
* THE TABLE BELOW IS SEARCHED FROM THE TOP TO THE BOTTOM. THE FIRST
* MATCH FOUND IS USED. IF NO MATCH IS FOUND, THE EBCDIC CODEPAGE
* SPECIFUIED IN FIELD EBCDIC_CODEPAGE IS USED.
* FOR BATCH SIGNON, THE CODEPAGE SPECIFIED IN FIELD EBCDIC_CODEPAGE
* IS USED.
* *****
*
* ADDRESS OF TERMINAL/NETNAME TABLE
*
ATERMTAB          DC      A(TERMTAB)
*
* *****
* MACRO DEFINITION USED FOR THE TERMINAL TABLE BELOW
* *****
      MACRO
&ENTRY  TERMCP &TERMID='*',&NETNAME='*',&EBCDICCP=
&ENTRY  DC      Y(L&SYSNDX-*)      LENGTH OF THIS ENTRY
      DC      CL4&TERMID          TERMINAL ID PATTERN
      DC      CL8&NETNAME         NETNAME PATTERN
      DC      CL16&EBCDICCP       EBCDIC CODEPAGE (ICONV FORMAT)
L&SYSNDX DC      OH              START OF NEXT ENTRY
      MEND
*
* *****
* START OF TERMINAL TABLE. ENTRIES ARE SPECIFIED AS FOLLOWS:
*      TERMCP TERMID='A*',NETNAME='D*',EBCDICCP='IBM-1047'
* *****
TERMTAB          DS      OF START OF TABLE
*
      TERMCP TERMID='A*',NETNAME='D*',EBCDICCP='IBM-1047' US ENGL.
      TERMCP TERMID='B*',NETNAME='N*',EBCDICCP='IBM-1141' GERMAN
      TERMCP NETNAME='X*',EBCDICCP='IBM-037' US
      TERMCP TERMID='T*',EBCDICCP='IBM-037' US
*
TERMTEND        DC      H'O'      END INDICATOR
*
* *****
* END OF SETTINGS
* *****
      END
/*
// IF $MRC GT 4 THEN
// GOTO NOLINK
// EXEC LNKEDT,PARM='MSHP'
* *****
* DONT FORGET TO NEWCOPY IESLDCFG IN ORDER TO ACTIVATE IT:
*      CEMT SET PROG(IESLDCFG) NEWCOPY

```

```

* *****
/. NOLINK
/*
/&
* $$ E0J

```

---

The following sections are shown in Example 3-1 on page 76:

- ▶ The general settings section contains a flag field, which allows you to turn on or off the LDAP sign-on support, enable the use of SSL, turn on or off the strict mode, and enable tracing for problem determination. You can replace the default name of the LDAP user mapping file IESLDUM with a new name.
 

You also can specify which EBCDIC code page should be used by z/VSE. The code page is important if you have special characters (for example @, [, ], {, }) in the user ID or password. The LDAP uses UTF-8, so all textual values must be converted from the specified EBCDIC code page to UTF-8 and vice versa. The last field in the general section specifies the expiration time for the LDAP password cache in seconds. Zero means that caching is disabled.
- ▶ The LDAP server or host name section is where you specify one or multiple IP addresses or host names of the LDAP servers to use for authentication. The servers are tried from the beginning to the end. If the first server is unavailable, the second server is tried, and so on. You can also specify the port number that is used by the LDAP server, which is separated by a colon. The default port numbers for LDAP are 389 and 636 for LDAP over SSL.
- ▶ The SSL-specific settings section is required only if SSL is enabled (see the flags in the general section). You must specify the keyring library, the name of the key to use, the cipher suites, and the SSL session timeout. For more information about how to set up SSL and create SSL keys and certificates, see Chapter 5, “Secure Sockets Layer with z/VSE” on page 197.
- ▶ The LDAP authentication settings section is where you configure how the LDAP authentication is performed. The following choices are available:
  - Direct BIND with LDAP user ID and password
 

This method uses a pattern to build an LDAP distinguished name (DN) with the user ID (for example, cn=%u,dc=ibm,dc=com, where %u is replaced with the user ID). The distinguished name is then used to perform a BIND operation with the password. This method requires that the user ID is part of the LDAP distinguished name.
  - Search for distinguished name using attribute
 

This method is used when the user ID is not part of the distinguished name. Therefore, an LDAP search operation is first performed to search for the associated entry. The search uses a filter such as uid=%u, where uid is the name of an attribute and %u is replaced with the user ID. The distinguished name of the search result is used to perform the final BIND operation with the specific password.

The setting for DN\_BIND\_PATTERN is used for direct authentication only. It specifies the pattern that is used to build the distinguished name with the user ID entered by the user.

The remaining settings are used for search authentication. BIND\_DN and BIND\_PWD specify the distinguished name and its password that is used to bind when performing the LDAP search operation. If these fields are all blank, an anonymous bind is used.



The field `USER_ATTRIBUTE` specifies the name of the attribute in the directory entry to search for. This attribute should uniquely identify an entry in the directory. `BASE_DN` specifies where to start the LDAP search. This can be the root of the directory tree, or a sub tree somewhere in the directory. The field `SEARCH_DEREF` controls how to handle references in the directory.

Entries can refer to other LDAP servers instead of storing the information. The `SEARCH_SCOPE` field controls how far to follow the tree when searching (for example, only direct children or the whole sub tree). The field `ADD_SEARCH_FILTER` allows you to specify another search filter, which can limit the search result even further. For example, you can use the search filter to restrict the search to only entries that are in certain groups. The last field, `SEARCH_TIMEOUT`, controls how long the search operation is allowed to take in seconds. Zero means no limit.

- ▶ Other SSL-specific settings specify the SSL security types that are used for TLS handshake. The value is for example 'TLSV1' or 'SSL30'. This field is only used when TLS is enabled (see field `FLAGS`).
- ▶ For configuring EBCDIC codepage for terminal or netname `ATERMTAB` specifies the address of a table that allows to specify the EBCDIC codepage per terminal-id or netname. The terminal-id's and netnames can be specified using wildcards, where \* can denote several characters and ? one character. See label `TERMTAB` for the start of this table. This table is only used when the support is enabled by way of the appropriate flag in the `FLAGS` fields. This support is useful for users from different countries that use different codepage settings in the terminal emulator. By configuring the appropriate EBCDIC codepage per group of terminals, LDAP sign-on can correctly handle special characters of different languages in the LDAP user-id or password.

For more information about the settings, see *z/VSE Administration*, SC34-2692.

When you specify your LDAP sign-on configuration settings, you must submit the skeleton to assemble and link the LDAP configuration phase. If you modify that skeleton later, you must also perform the following action in CICS:

```
CEMT SET PROG(IESLDCFG) NEWCOPY
```

### 3.6.2 Sign on to z/VSE with active LDAP sign-on support

With z/VSE, you can use LDAP sign-on by using the following methods:

- ▶ In a CICS sign-on panel (available since z/VSE V4R2)
- ▶ In a batch job (available since z/VSE V4R3)

## LDAP sign-on by way of the LDAP sign-on panel

If you use the z/VSE Interactive Interface sign-on panel for signing on to z/VSE, you see the LDAP sign-on panel IESADMSO3 with longer fields for user ID and password, as shown in Figure 3-8.

```
IESADMSO3                                z/VSE ONLINE
5609-ZV4 and Other Materials (C) Copyright IBM Corp. 2007 and other dates

                                ++
                                ++  VV  VV  SSSSS  EEEEEEE
                                ++  VV  VV  SSSSSS  EEEEEEE
zzzzzz                ++  VV  VV  SS  EE
zzzzz                 ++  VV  VV  SSSSSS  EEEEEEE
zz                    ++  VV  VV  SSSSSS  EEEEEEE
zz                    ++  VV  VV  SS  EE
zzzzzz                ++  VVVV  SSSSSS  EEEEEEE
zzzzzzzz              ++  VV  SSSSS  EEEEEEE

Your terminal is A000 and its name in the network is D38001
Today is 11/27/2008 To sign on to DBDCCICS -- enter your:

USER-ID. _____
PASSWORD _____

PF1=HELP          2=TUTORIAL    3=TO VM          4=REMOTE APPLICATIONS
                                     10=NEW PASSWORD
```

Figure 3-8 LDAP sign-on panel

If you are not using the Interactive Interface sign-on panel but your own sign-on program or panel, you might have to perform a few modifications to your sign-on program to add support for LDAP authentication. You do not have to rewrite the entire logic of the LDAP processing yourself; instead, you can call the z/VSE LDAP sign-on program IESLDSOC from your own sign-on program and so that it can perform the LDAP authentication, including the user mapping.

The z/VSE LDAP sign-on program can be called by using EXEC CICS LINK with a COMMAREA. It expects the long LDAP user ID and password as input and performs the LDAP authentications and the user mapping. It returns the status of the authentication, success or failure, and the corresponding short VSE user ID and password. Your own sign-on program can then pass the VSE user ID and password to the underlying security manager to perform the final sign-on.

The following changes might be required in your program:

- ▶ Increase the length of the fields for user ID and password.
- ▶ Call the z/VSE LDAP sign-on program IESLDSOC to perform the LDAP authentication.
- ▶ Add error handling and messages for errors that are returned from the z/VSE LDAP sign-on program IESLDSOC.

A sample CICS sign-on program is available in source code that can be used as a replacement for the CICS CESN sign-on program. This source code is available at this web page:

<http://www.ibm.com/zvse/downloads/samples.html#samplecode>

## LDAP sign-on by way of a batch job (available since z/VSE V4R3)

If you want to use LDAP sign-on in a batch job, you must include the following statements:

```
// EXEC IESLDSOB  
USER=xxx...  
PWD=xxx...  
/*
```

When this batch job is started, the z/VSE sign-on program IESLDSOB calls the LDAP sign-on module IESLDSO. The LDAP sign-on module searches the LDAP file for the record containing the user ID mapping. Depending upon whether the record is found, one of the following conditions apply:

- ▶ If the record that contains the user ID mapping is not found and LDAP is being operated in:
  - Strict mode, the sign-on attempt is rejected.
  - Non-strict mode and the user ID or password is greater than eight characters, the sign-on attempt is rejected.
  - Non-strict mode and the user ID and password are both less than or equal to eight characters, a mapping of user IDs does not occur. The sign-on attempt is then sent unchanged to the security manager; for example, to the Basic Security Manager (BSM).

For more information about strict mode and non-strict mode, see 3.5.2, “Strict mode” on page 75.

- ▶ If the record containing the user ID mapping is found and the user ID is not LDAP enabled, the mapping of the user ID does not occur. The sign-on attempt is then sent unchanged to the security manager.
- ▶ If the record containing the user ID mapping is found and the user ID is LDAP enabled, an LDAP authentication is performed with the remote LDAP server. Consider the following points:
  - If the LDAP authentication is successful, the user ID is mapped. The LDAP sign-on module IESLDSO returns the short user ID and short password (the “z/VSE” user ID and password) to the z/VSE sign on program IESLDSOB. IESLDSOB sends the sign-on attempt to the security manager.
  - If the LDAP authentication is not successful, the sign-on attempt is rejected.

Depending upon whether the sign-on request was accepted or rejected in the previous steps, the z/VSE sign-on program IESLDSOB continues to process the sign-on request by using the short z/VSE user ID and password in the same way as when processing an // ID statement.

## 3.7 Administering the LDAP user mapping file

In addition to configuring and activating the LDAP sign-on support, you must create the LDAP mapping information. The mapping information is stored in the LDAP user mapping file, which is a VSAM KSDS cluster. This file is automatically defined during base installation or Fast Service Upgrade (FSU) in the VSESP.USER.CATALOG. It does not contain any records by default.

For each user who uses an LDAP sign-on, you must add a mapping to the LDAP mapping file. To do so, use the LDAP mapping tool IESLDUMA. This batch tool enables you to maintain the contents of the LDAP mapping file.

In addition to adding LDAP user mappings, you can use the IESLDUMA tool to perform the following tasks:

- ▶ Change LDAP user mappings
- ▶ Delete LDAP user mappings
- ▶ List LDAP user mappings
- ▶ Export all definitions for migrating LDAP user mappings to a new system
- ▶ Reset the cached password hash for a user (supported from z/VSE 6.2 on)

You should add the following users as *not LDAP-enabled* (TYPE=VSE):

- ▶ Administrators, such as SYSA
- ▶ Operators
- ▶ Security administrators

User IDs that are to be used for LDAP authentication are referred to as being *LDAP-enabled*. User IDs that are not used for LDAP authentication are referred to as being *not LDAP-enabled*. These users can sign on to z/VSE, even if the LDAP server is not operational.

After a user ID is LDAP-enabled, this user should no longer be able to perform an LDAP sign-on using the short user ID as used in z/VSE. Doing so bypasses the company's security policies that are enforced by the LDAP authentication.

When enabling a user ID for LDAP authentication, a new z/VSE password can be randomly generated and a password change request is issued by using RACROUTE.

The randomly generated password is encrypted and stored in the LDAP mapping file. The user is never aware of the randomly generated password. Therefore, the user cannot perform an LDAP sign-on by using the short user ID, as used in z/VSE.

You should set the short user IDs of z/VSE that are LDAP-enabled to non-expiring. For such short user IDs, password expiration should be enforced by the LDAP server based on the long user ID and long password.

The LDAP mapping tool is started through the following statement, which reads control statements from SYSIPT:

```
// EXEC IESLDUMA
```

Because this tool provides security-sensitive services, it can be used by administrator type users only.

If batch security is active in z/VSE, you must specify an ID statement in the job that is running IESLDUMA. If batch security is inactive, you must specify the user ID and password of an administrator by using the ID command as the first command when IESLDUMA is run.

The LDAP mapping tool uses the following syntax:

```
// ID USER=nnn,PWD=nnn
// EXEC IESLDUMA
control statements
/*
```

The LDAP mapping tool provides the following commands:

ID	Authorizes an administrator to use the tool
ADD	Adds a user mapping
CHANGE	Changes a user mapping
DELETE	Deletes a user mapping
LIST	Lists all user mappings

EXPORT	Exports the user mappings
RESET	Clear the cached password hash for a user

From z/VSE 6.2 on, the commands allow wildcards for user ID specification, where '\*' can denote several characters and '?' one character. For example, LDAPUSER=john\* deletes all user IDs starting with "john".

Example 3-2 shows how to specify control statements by using the LDAP mapping tool.

*Example 3-2 LDAP mapping tool*

---

```
// JOB RUNLDUMA
// EXEC IESLDUMA
ID USER='SYSA' PWD='password'
ADD LDAPUSER='hugo@de.ibm.com' VSEUSER='HUGO' -
    TYPE=LDAP GENPWD PWDPATTERN='aaaaaaaa' OLDPWD='INITPWD' -
    DESC='Hugo Maier'
ADD LDAPUSER='HUGO' TYPE=VSE DESC='Hugo Schmidt'
DELETE LDAPUSER='HUGO'
CHANGE LDAPUSER='hugo@de.ibm.com' VSEUSER='FRAN' -
    TYPE=LDAP -
    DESC='Hugo Maier'
LIST
EXPORT
/*
/ &
```

---

For more information about the commands, see *z/VSE Administration*, SC34-2692.

### 3.7.1 Using the Maintain LDAP user profiles dialog

To maintain the LDAP mapping information z/VSE provides a dialog function.

The dialog provides a subset of the LDAP mapping tool IESLDUMA. Typically, you can add the LDAP-enabled user ID information for new users. With the RESET function (z/VSE 6.2) you can request that a user is forced to a full LDAP authentication for the next time they sign in.

Select MAINTAIN LDAP USER PROFILES by using fast path **217**, as shown in Figure 3-9.

```

IESADMLUPM          MAINTAIN LDAP USER PROFILES

START.... _____
VSE USERID.... _____
OPTIONS:  1 = ADD      2 = CHANGE      3 = DISPLAY      5 = DELETE
              7 = RESET
OPT  LDAP USERID                                USER
      TYPE
  1  hugo@de.ibm.com                             LDAP
  _  kdwacker@de.ibm.com                         LDAP

PF1=HELP          3=END
                  9=PRINT          10=EXPORT      11=RESET ALL

```

Figure 3-9 Maintain LDAP user profiles dialog

To add LDAP-enabled user IDs, specify **1** in the option field and press Enter (see Figure 3-10).

```

IESADMLUPA          ADD OR CHANGE LDAP USER PROFILE

LDAP USERID.. hugo@de.ibm.com

DESCRIPTION.. Hugo Maier

VSE USERID..... FRAN      Assigned VSE user-ID. 1-8 characters
VSE PASSWORD.....         Specifies VSE password. 3-8 char or blank
GENERATE PASSWORD.. _      1 - Forces generation of random VSE passw
                          2 - Use current password
PASSWORD PATTERN... _____ Specifies a pattern for password generation
                          Required if password is generated
                          d - decimal digit (0-9)
                          c - character (A-Z)
                          a - decimal digit (0-9) or character (A-Z)
                          x - special character (@, # or $)
                          other - place is filled with specified char
                          blank - place is not filled with a char

PF1=HELP          3=END          5=PROCESS

```

Figure 3-10 Add or change LDAP user profile

Figure 3-10 shows an ID from the selected model user; overwrite the fields with the information for the new user.

## 3.8 LDAP sample setup

This section describes one example of an LDAP setup on the IBM intranet. The setup uses the IBM Blue® Pages server as the LDAP server. IBM Blue Pages is an online directory listing all IBM employees with some personal information.

The setup consists of the following steps:

1. Modify the LDAP configuration phase.
2. Map an IBM intranet user to a VSE user.
3. Modify the TCP/IP setup to access the LDAP server.

### 3.8.1 Modifying the LDAP configuration phase

In this sample setup, the following parameters in job skeleton SKLDCFG were changed:

CACHE_EXPIRATION	DC	F'2'
LDAP_SERVERS	DC	CL256'bluepages.ibm.com:389'
AUTH_METHOD	DC	F'2'
USER_ATTRIBUTE	DC	CL64'emailaddress'
BASE_DN	DC	CL64'ou=bluepages,o=ibm.com'

The CACHE\_EXPIRATION parameter now allows caching of an LDAP password for 2 minutes. That means, if the same LDAP user logs on again within 2 minutes, the entered password is checked against the cached password.

The LDAP\_SERVERS parameter specifies the DNS name and port number of the IBM Blue Pages server. We use AUTH\_METHOD set to 2, which is the search authentication. Hereby, the LDAP user ID is not used directly for the BIND. Instead, a search is performed for the user ID using a specific attribute. In this example, it is the email address attribute of the IBM Blue Pages entry. The distinguished name (DN) of the search result is then used to perform the bind.

The BASE\_DN parameter specifies the tree node, where the LDAP server shall start its search for the specified user attribute. In this example, the entire ibm.com subtree is not searched. Instead, the search is limited to entries within the *bluepages* subtree.

The LDAP configuration phase in CICS now must be refreshed, as shown in the following example:

```
CEMT SET PROG(IESLDCFG) NEWCOPY
```

### 3.8.2 Mapping an intranet user ID to a z/VSE user ID

The IBM intranet user ID jschmidb@de.ibm.com is now mapped to a z/VSE user ID using the IESLDUMA utility. It is important that you define the user with TYPE=LDAP, as shown in Example 3-3.

*Example 3-3 Update job for user ID mapping*

---

```
* $$ JOB JNM=RUNLDUMA,CLASS=0,DISP=D
// JOB RUNLDUMA
// EXEC IESLDUMA
ID USER='JSCH' PWD='VSEPASSW'
ADD LDAPUSER='jschmidb@de.ibm.com' VSEUSER='JSCH' DESC='JOERG SCHMIDBAUER' -
    TYPE=LDAP VSEPWD='VSEPASSW'
LIST
```

```
EXPORT
/*
/&
* $$ E0J
```

---

### 3.8.3 Modifying the TCP/IP setup

Because TCP/IP for VSE/ESA must access the LDAP server, you must define a domain name server. Then, ensure that the LDAP server can be reached from z/VSE. You can use console commands that are shown in Example 3-4.

*Example 3-4 Define domain name server and test connection*

---

```
msg f7
AR 0015 1140I  READY
F7-0112 IPN300A Enter TCP/IP Command
112 set dns1=9.152.120.241
F7 0109 IPN254I DNS 1 address is 9.152.120.241, Timeout is 1200
F7-0112 IPN300A Enter TCP/IP Command
112 ping bluepages.ibm.com
F7-0112 IPN300A Enter TCP/IP Command
F7 0109 0432: TCP915I  PINGING 009.017.186.253 (bluepages.ibm.com)
F7 0109 0432: TCP910I  PING 1 was successful, milliseconds: 00138.
F7 0109 0432: TCP910I  PING 2 was successful, milliseconds: 00138.
F7 0109 0432: TCP910I  PING 3 was successful, milliseconds: 00138.
F7 0109 0432: TCP910I  PING 4 was successful, milliseconds: 00138.
F7 0109 0432: TCP910I  PING 5 was successful, milliseconds: 00138.
F7 0109 0432: TCP910I  PING Complete
```

---

You are now ready to log on to z/VSE by using the intranet user ID and intranet password.

**Note:** You should permanently define the DNS entry in your IPINIT member.

### 3.8.4 Setting up for SSL

This section shows how to set up LDAP with SSL.

To activate SSL, complete the following steps:

1. Set up a z/VSE keyring by using the Keyman/VSE tool. This process is described in 5.1, “Generating the server key and certificates” on page 198.
2. Activate SSL in the LDAP configuration phase.

To activate SSL in the LDAP configuration phase, you first must set the SSL flag in the FLAGS parameter of the LDAP configuration, as shown in Example 3-5.

*Example 3-5 Set the flag*

---

FLAGS	DC	XL4'00000001'	
LDAP_AUTH_ENABLED	EQU	X'00000001'	LDAP AUTH IS ENABLED
USE_SSL	EQU	X'00000002'	SSL IS ENABLED
STRICT_MODE	EQU	X'00000004'	STRICT USER MAPPING
FAILURE_HASH	EQU	X'00000008'	FAILURE PASSWORD HASH USED
UPPERCASE_MODE	EQU	X'00000010'	UPPERCASE LDAP USERID & PWD
EBCDICCP_PER_TERM	EQU	X'00000020'	USE TERM.TABLE FOR CODEPAGE



SIGNON_MSG_SYSLOG	EQU	X'01000000'	SIGNON MESSAGES TO SYSLOG
SIGNON_MSG_SYSLST	EQU	X'02000000'	SIGNON MESSAGES TO SYSLST/TDQ
TRACE	EQU	X'80000000'	ENABLE TRACING

Next, change the port number of your LDAP server to the SSL port. This port number is 636 in Example 3-6.

*Example 3-6 Change the port number*

SPECIFICATION CONTAINS THE IP ADDRESS OR HOSTNAME OF ONE OR MORE LDAP SERVERS IN THE FORM <SERVER>:<PORT>. MULTIPLE SERVER ARE SEPARATED BY BLANKS. IF THE PORT NUMBER IS OMITED, THE DEFAULT PORT NUMBERS ARE USED:

389 - NON-SSL

**636 - SSL**

PLEASE NOTE THAT MICROSOFT ACTIVE DIRECTORY MAY USE DIFFERENT PORT NUMBERS. THE AD GLOBAL CATALOG SERVER USES PORT 3268 PER DEFAULT.

LDAP_SERVERS	DC	CL256'bluepages.ibm.com:636'
--------------	----	------------------------------

Finally, specify your z/VSE keyring parameters, as shown in Example 3-7.

*Example 3-7 Specify keyring parameters*

KEYRING_LIBRARY	DC	CL16'CRYPTO.KEYRING'
KEYNAME	DC	CL8'LDAP01'
CIPHER_SPEC	DC	CL64'010208090A62'
SESSION_TIMEOUT	DC	F'86440'

Be aware that this setup depends on your LDAP environment and might be different from the one described here.

Now, catalog the IESLDCFG phase again and issue the following command:

```
CEMT SET PROG(IESLDCFG) NEWCOPY
```

### 3.8.5 Observations

This section describes several observations that we made during our tests.

When the sign-on process fails, check the CICS job output for details about the failure. To obtain more trace information, you can turn on the trace in the LDAP configuration phase by setting the leftmost bit to a value of 1 in the FLAGS parameter (see Example 3-8).

*Example 3-8 Setting the trace on by adjusting the FLAGS bit*

FLAGS	DC	XL4'00000001'	
LDAP_AUTH_ENABLED	EQU	X'00000001'	LDAP AUTH IS ENABLED
USE_SSL	EQU	X'00000002'	SSL IS ENABLED
STRICT_MODE	EQU	X'00000004'	STRICT USER MAPPING
FAILURE_HASH	EQU	X'00000008'	FAILURE PASSWORD HASH USED
UPPERCASE_MODE	EQU	X'00000010'	UPPERCASE LDAP USERID & PWD
EBCDICCP_PER_TERM	EQU	X'00000020'	USE TERM.TABLE FOR CODEPAGE
SIGNON_MSG_SYSLOG	EQU	X'01000000'	SIGNON MESSAGES TO SYSLOG
SIGNON_MSG_SYSLST	EQU	X'02000000'	SIGNON MESSAGES TO SYSLST/TDQ
TRACE	EQU	X'80000000'	ENABLE TRACING

**Note:** You should be aware that the CICS log shows the entered passwords in readable format.

## Sign-on failed

The symptom and possible reason are listed in this section.

### *Symptom*

The message and output are listed:

- ▶ The following message appears in the LDAP sign-on window:  
"SIGN-ON FAILED. INFORMATION LOGGED."
- ▶ The CICS job output shows:  
IESC3001E SEVERE ERROR WHILE PERFORMING LDAP AUTHENTICATION  
RC='11' FDBK='91' OPERATION='BIND(2)'  
IESV0089I FOLLOWING MESSAGE(S) FROM PROGRAM 'IESIES01', OFFSET X'2E14'.  
IESA0951I CICS SIGNON FOR USER 'jschm...' AT TERMINAL 'A000' FAILED.
- ▶ When the trace is turned on, the following output shows:  
-> ldap\_connect: called  
ldap\_connect: Resolving host 'bluepages.ibm.com'.  
ldap\_connect: Host 'bluepages.ibm.com' cannot be resolved.  
ldap\_bind\_s rc=91

### *Possible reason*

The LDAP server cannot be reached by TCP/IP for VSE/ESA because of a missing DNS definition. Ensure that a domain name server is defined and the LDAP server can be pinged from z/VSE. For more information, see 3.8.3, "Modifying the TCP/IP setup" on page 88.

## User ID is not defined to the online system

The symptom and possible reason are listed.

### *Symptom*

The following message and output are produced:

- ▶ The following message appears on the LDAP sign-on window:  
"USER ID 'jschm...' IS NOT DEFINED TO THE ONLINE SYSTEM."
- ▶ The CICS job output includes following message:  
Signon at netname D0910001 has failed. User JSCHMIDB not recognized.

### *Possible reason*

The LDAP user is defined with TYPE=VSE, which means that it is treated as a normal z/VSE user. Use the IESLDUMA utility to delete the definition and redefine it with TYPE=LDAP.

## **SSL sign-on failed**

The symptom and possible reason are listed.

### ***Symptom***

The message and output are:

- ▶ LDAP sign-on with SSL fails. The following message appears on the LDAP sign-on window:  
"SIGN-ON FAILED. INFORMATION LOGGED."
- ▶ The trace in the CICS job output shows:  
ldap\_connect: Connecting to 'bluepages.ibm.com' (9.17.186.253) on port 389.  
ldap\_connect: start SSL handshake  
ldap\_connect: SSLHandle =FFFFFF3E4  
ldap\_connect: SSL handshake failed.

### ***Possible reason***

In this example, the wrong port number is used. Port 389 is the non-SSL port, and 636 is the SSL-port of the IBM Blue Pages server.





## Cryptography on z/VSE

This chapter provides an overview of cryptography and its implementation in z/VSE. We describe how to prepare a z/VSE system to increase the security of your environment by using encryption technology. Because cryptographic operations are processor-intensive by nature, we also describe how to set up your environment to realize the most benefit from the cryptographic hardware capabilities of IBM Z.

Encryption technology covers many areas, such as software-based and hardware-based encryption, securing network connections, and encryption of stored data.

Sometimes, hardware- and software-based encryption are used together. For example, the SSL/TLS support in z/VSE can use software implementations of encryption algorithms, but can also take advantage of cryptographic hardware that is provided by the System z processor. The same is true for Encryption Facility for z/VSE, which provides encryption of z/VSE data sets and tapes.

In addition to the most widely used SSL/TLS connections between a web server and a web browser, SSL/TLS is used in z/VSE by the following products:

- ▶ CICS Web Support
- ▶ LDAP
- ▶ Secure FTP
- ▶ Secure Telnet
- ▶ VSE/POWER PNET<sup>1</sup>
- ▶ WebSphere MQ
- ▶ z/VSE connectors
- ▶ z/VSE VTAPE support
- ▶ OpenSSL on z/VSE
- ▶ EZA interface

Protecting information from unintended use is not limited to the active data on your system. Backup on tapes or data transfer with tapes can be critical, especially if the tapes get in the wrong hands. We show you what tape encryption can do for you and how you can make use of it.

---

<sup>1</sup> For more information about PNET SSL, see *VSE/POWER Networking*, SC33-8249.

In addition, z/VSE can also use the Full Disk Encryption capabilities as provided by the IBM DS8000® series storage controller. Because this support is transparent to z/VSE, it is not described in this book.

This chapter includes the following topics:

- ▶ 4.1, “Cryptography introduction” on page 95
- ▶ 4.2, “Configuring cryptographic hardware” on page 102
- ▶ 4.3, “Hardware-based tape encryption with z/VSE” on page 126
- ▶ 4.4, “Example of TS1120 installation” on page 134
- ▶ 4.5, “Software-based encryption with Encryption Facility for z/VSE V1R1” on page 145
- ▶ 4.6, “Software-based encryption with Encryption Facility for z/VSE V1R2” on page 160
- ▶ 4.7, “z/VSE Navigator GUI for Encryption Facility” on page 193

## 4.1 Cryptography introduction

Before the computer era, cryptography was primarily used to transform any kind of sensitive message into a non-readable form (encryption) and vice versa (decryption), making it impossible for interceptors to get access to the confidential message text. Hereby, the mechanism of encrypting and decrypting information is called a *cipher*.

Some early ciphers were based on the rearrangement of the order of characters in a message (transposition ciphers), but others, for example, replace characters with other characters (substitution ciphers). However, with the help of computers and mathematical methods, all of these ciphers are relatively easy to crack, because they all reveal statistical information about the plaintext that is imposed by the nature of the used national language.

Modern ciphers feature one important difference to the old ciphers. Early ciphers were based on the secrecy of the encryption mechanism; but the algorithm of modern ciphers is completely known by the public. The only secret is a random number or a set of numbers, called the *encryption key*. With the help of this key, which is different with each new encryption process, the plain message text is transformed through the non-secret algorithm into a non-readable cipher text. Therefore, one challenge of modern cryptography is the generation of random numbers that cannot be guessed by any attacker.

### 4.1.1 Modern cryptography

In today's computing world, cryptography is used for message encryption, decryption, and ensuring the integrity of a secret message. It is also used to verify the identity of the two communication partners that are exchanging secret information. We must distinguish the following basic scenarios:

- ▶ Exchanging secret information online over a network (SSL/TLS, Secure FTP, Secure Telnet, and others)
- ▶ Exchanging secret data on physical media (tapes, CD-ROMs, USB sticks, portable disks, and others)

In both scenarios, data encryption and decryption is usually performed with a symmetric cipher. That means, the same key is used for encryption and decryption (see Figure 4-1).

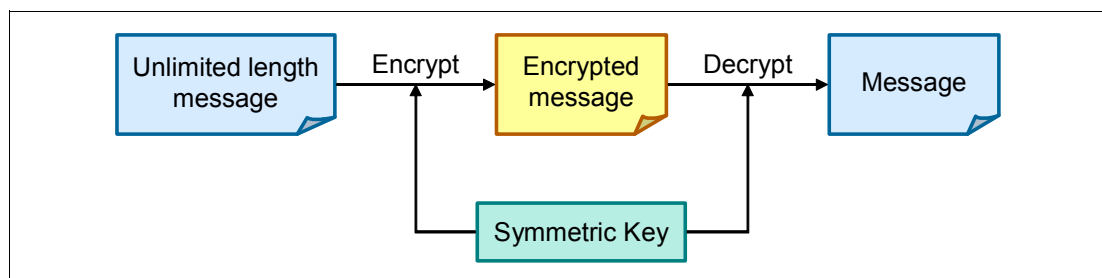


Figure 4-1 Symmetric encryption

Symmetric ciphers can be divided into two categories: stream ciphers and block ciphers. A stream cipher can process any specific number of bytes; a block cipher encrypts a series of plaintext blocks with a fixed length.

Symmetric ciphers feature one major disadvantage. Encrypting specific plaintext blocks with a specific key might result in recognizable data patterns in the cipher text because each piece of plaintext results in the same piece of ciphertext. To avoid such patterns, so called encryption modes (modes of operation) were introduced.

### 4.1.2 Encryption modes

When encrypting data by using block ciphers, chaining modes are used to ensure better security. *Chaining* refers to encrypting input data block-by-block, taking the result of a previous block into the processing of the next block. This way, each encrypted block depends on all previous blocks from the beginning of the input data.

The following chaining modes are available:

- ▶ Cipher block chaining (CBC)
- ▶ Cipher feedback (CFB)
- ▶ Output feedback (OFB)
- ▶ Counter mode (CTR)
- ▶ Galois Counter mode (GCM)

In many cryptographic systems, the plaintext is compressed before being encrypted, which further eliminates any visible patterns in the encrypted data and leads to less data that must be encrypted.

However, to solve the problem of transferring a specific key securely over a network when establishing a secure connection, another concept had to be introduced.

### 4.1.3 Verifying the identity of communication partners

In 1977, three researchers at MIT (Ron Rivest, Adi Shamir, and Leonard Adleman) invented a cryptography algorithm, named RSA, which is the initials of their surnames. RSA features two keys: one is used for encryption and the other is for decryption. Both keys, called an RSA key pair, are generated through one mathematical calculation. Because of the two keys, RSA is called an asymmetric cipher.

The basic idea behind RSA is to keep one of the keys private (the private key), and the other key can be sent to any user (the public key). Therefore, anyone can use a public key to encrypt data and send the encrypted message to the owner of the corresponding private key over a network. By definition, the encrypted message can only be decrypted with the help of the private key, which is always only in the possession of the owner of this RSA key pair.

Therefore, only the owner of this RSA key pair can decrypt the message and RSA solves the problem of verifying the identity of a communication partner. The person who encrypts a message with a public key can be sure that only the owner of the related private key can decrypt it. Figure 4-2 shows how asymmetric encryption works.

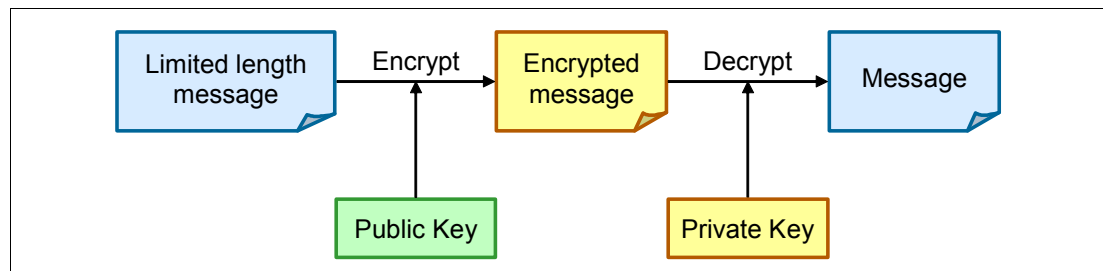


Figure 4-2 Asymmetric encryption



One disadvantage of asymmetric encryption is that the plain input message cannot be longer than the key length of the underlying RSA key pair. In practice, a few bytes are even reserved for internal processing purposes. Therefore, encrypting a large message by using RSA results in repeated encryption steps until the input message is fully processed.

Another disadvantage of asymmetric encryption is poor performance. In practice, asymmetric encryption is always used with symmetric encryption. Data is encrypted by using a fast symmetric cipher and the few symmetric key bytes are encrypted by using asymmetric encryption.

Typical RSA key lengths are 1024, 2048, and 4096 bits, which are all supported on z/VSE. Because of the huge processing overhead, we highly recommend the use of a Crypto Express feature for cryptographic acceleration. RSA keys with 512 bits are considered to be highly unsecure and should no longer be used.

With increasing RSA key lengths, Elliptic Curve Cryptography (ECC), provides the same security level with much smaller key sizes, which is used more often today. OpenSSL on z/VSE supports ECC acceleration with Crypto Express4S and later.

#### 4.1.4 Ensuring data integrity

One problem remains: ensuring the integrity of a secret message. This problem is solved by hash functions, which take any kind of data of any length as input and produce one single fixed length number, called the message digest, hash value, or fingerprint as output.

The ideal hash function features three main properties:

- ▶ Calculating a hash for any data is extremely easy.
- ▶ Constructing a text that has a hash is extremely difficult, or almost impossible.
- ▶ Two separate messages, however similar, are extremely unlikely to have the same hash.

The concept of applying the hash function is shown in Figure 4-3.

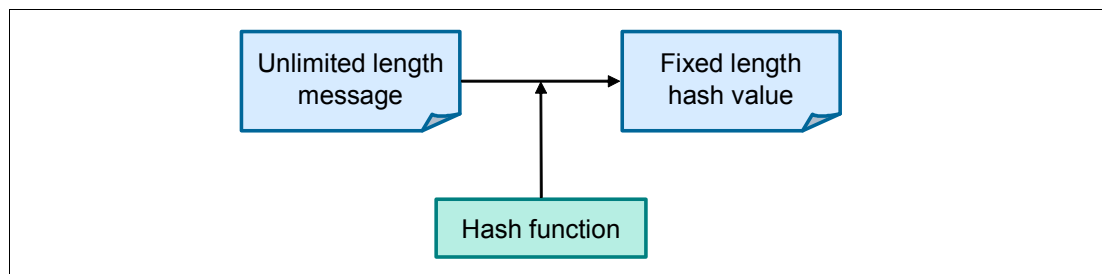


Figure 4-3 Hash function

In modern cryptography, a hash value is calculated from the plaintext message. Then, the message is encrypted, and finally, the hash value is appended to the encrypted message. The receiver of the encrypted message first decrypts the encrypted message and again calculates the hash value by using the same hash function. Having two hash values that match proves that the message was not altered between sender and receiver.

Typical hash functions and their digest lengths are listed in Table 4-1.

Table 4-1 Hash functions

Hash function	Digest length	Supported by z/VSE	Hardware support
MD5	16 bytes	Yes	No
RIPED-160	20 bytes	Yes (OpenSSL)	No
SHA-1	20 bytes	Yes	Yes, since z890/990
SHA-224	28 bytes	Yes	Yes, since IBM z9®
SHA-256	32 bytes	Yes	Yes, since z9
SHA-384	48 bytes	Yes	Yes, since IBM z10™
SHA-512	64 bytes	Yes	Yes, since z10

All three concepts together (data encryption, verifying the identity of communication partners, and ensuring data integrity) are used in modern cryptographic systems.

### 4.1.5 Secure Sockets Layer and Transport Layer Security

Secure Sockets Layer (SSL) and Transport Layer Security (TLS) are cryptographic protocols that provide communications security over a computer network. Symmetric, asymmetric, and hash algorithms are used to establish secure network connections by verifying the identity of communication partners, encrypting data that is sent over the network, and providing data integrity. SSL/TLS connection properties are specified by an SSL/TLS cipher suite, which consists of an asymmetric algorithm, a symmetric algorithm with an encryption mode, and a hash function.

The following mechanisms are widely used for establishing a secure SSL/TLS connection:

- ▶ Plain RSA-based key exchange. A randomly created symmetric session key is encrypted with the server's public RSA key and then exchanged between client and server. This mechanism includes a general weakness, because the session key is part of the session data. If the server's private RSA key is compromised in future, past sessions that might be recorded and saved can then be decrypted.
- ▶ Diffie-Hellman based SSL/TLS key agreement (DH). The symmetric session key is never sent over the network. Instead, it is calculated by client and server independently based on DH parameters that are not secret. For performance reasons, the DH key exchange is often used with ECC. This mechanism provides "forward secrecy", because even when an ECC key is compromised, past sessions cannot be decrypted.

The following SSL/TLS implementations are available on z/VSE 6.2:

- ▶ SSL4VSE as part of TCP/IP for z/VSE V2.2 from Connectivity Systems International (CSI). This SSL/TLS implementation is internally used for all TCP/IP for z/VSE applications.
- ▶ OpenSSL, which is part of z/VSE base and used by IPv6VSE from Barnard Software, Inc. For more information about OpenSSL, see *z/VSE TCP/IP Support*, SC34-2706, and *Enhanced Networking on IBM z/VSE*, SG24-8091.

LE/C applications can choose which SSL/TLS implementation to use by using the LE/C multiplexer (EDCTCPMC in ICCF library 62).

CICS Web Support (CWS) can specify which SSL/TLS implementation to use by way of JCL (SETPARM) variables. For more information about the use of OpenSSL in CICS, see *CICS Transaction Server V2.2 Enhancements Guide*, SC34-2685.

EZA applications can use the EZA multiplexer to specify the SSL/TLS implementation. For more information about the use of OpenSSL for EZA applications, see *z/VSE TCP/IP Support*, SC34-2706.

The most relevant SSL/TLS cipher suites in z/VSE V6R2 and their support by OpenSSL and TCP/IP for z/VSE V2.2 are listed in Table 4-2.

Table 4-2 Relevant cipher suites in z/VSE

Hex code	OpenSSL	TCP/IP for z/VSE V2.2
0A	DES-CBC3-SHA	SSL_RSA_WITH_3DES_EDE_CBC_SHA
2F 35	AES128-SHA AES256-SHA	TLS_RSA_WITH_AES_128_CBC_SHA TLS_RSA_WITH_AES_256_CBC_SHA
3C 3D	AES128-SHA256 AES256-SHA256	RSA_AES128CBC_SHA256 RSA_AES256CBC_SHA256
15 16 33 39 67 6B	EDH-RSA-DES-CBC-SHA EDH-RSA-DES-CBC3-SHA DHE-RSA-AES128-SHA DHE-RSA-AES256-SHA DHE-RSA-AES128-SHA256 DHE-RSA-AES256-SHA256	-
C011 C012 C013 C014 C027	ECDHE-RSA-RC4-SHA ECDHE-RSA-DES-CBC3-SHA ECDHE-RSA-AES128-SHA ECDHE-RSA-AES256-SHA ECDHE-RSA-AES128-SHA	-

Consider the following points:

- ▶ Cipher suite 0A should only be used for backward compatibility. The Triple-DES algorithm is less secure and even slower than AES.
- ▶ Cipher suites 2F and 35 can be used when TLSv1.2 is unavailable.
- ▶ Cipher suites 3C and 3D are only available with TLSv1.2.
- ▶ The EDH and DHE cipher suites (Diffie-Hellmann) perform poorly and should only be used if ECDHE is not possible.
- ▶ The ECDHE cipher suites (Elliptic Curve and Diffie Hellman) should preferably be used for high security and performance.

#### 4.1.6 Use of certificates

A certificate is similar to an ID card or driver's license. A certificate includes an issuer of the certificate (the certificate authority, or CA), a subject (the owner of the certificate), a validity period, and other properties that belong to the certificate.

One important certificate property is the public key of the certificate owner. That is, the public key is the central point of a certificate. All other properties of a certificate are necessary only to prove the identity of the certificate owner.

The issuer of the certificate signs the certificate by calculating a hash value of all certificate properties, including the public key, and encrypting the resulting hash value with its private key. The result is now called the signature.

Through the corresponding public key of the issuer, anyone can now verify that the signature was calculated by this issuer. The issuer, in turn, guarantees the correctness of the certificate, which means that this public key really belongs to the certificate owner.

For example, web browsers feature preinstalled lists of public keys of many known certificate authorities (CAs), so that verifying the signature of any received certificate when an SSL session is established is possible.

The receiver of a certificate can now trust that the contained public key is indeed the public key of the certificate subject if the signature is made by a well-known CA.

### 4.1.7 Comparison of key sizes

Symmetric ciphers feature relatively short key sizes compared to the RSA key sizes. The reason is that the various algorithms have different strengths in terms of their ability to resist against attacks. To increase the strength of an algorithm, you often increase the key size. The equivalent algorithm strengths for different key sizes (in bits) are listed in Table 4-3.

Table 4-3 Equivalent algorithm strength for different key sizes<sup>2</sup>

RSA	ECC	Symmetric	Hash	Strength
		RC4		
		DES	MD5	
			SHA1	<80
1024	160			80
2048	224	Triple-DES	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

Today, you typically want a security level of 128 bits, which is a good compromise between performance and security. You choose AES-128 for data encryption and SHA-256 for creating digital fingerprints. In SSL/TLS, you choose ECC with an elliptic curve of 256 bits.

When RSA is used, you choose 2048 bits for better performance or 4096 bits if you have high security requirements. Unfortunately, no commonly used RSA key length that directly relates to a security level of 128 bits is available. Most applications choose 2048 bits.

The following sections introduce more terms and concepts that are important to understand how cryptography is supported by z/VSE.

<sup>2</sup> Parts of this table are taken from RFC4880.

## 4.1.8 Password-based encryption

In this chapter thus far, we introduced an encryption key as a random number, which is generated in advance of the encryption process. Another way of producing an encryption key is with the help of a password.

The encryption key, often also called the *data key* or *session key*, is generated from a specific secret password and two other parameters, which are not secret. These parameters are a so-called iteration count, and another random number (the *salt* value), which is different for each encryption process.

Generating the encryption key from a password and iteration count uses the following overall process:

1. Concatenate the password bytes with a random number, the salt value, which is usually an 8-byte length.
2. Apply a hash function; for example SHA-1, to the resulting byte string, which returns a 20-byte hash value.
3. Repeatedly apply the hash function to the obtained hash value until iteration count times are reached.
4. Take the final hash value as the encryption key.

This process has the advantage that the key is not dependent on only the characters that you enter by using your keyboard. This example also has the disadvantage that it can produce encryption keys only up to a 20-byte length, which is not sufficient in most cases. For a triple-DES key (3 x 8 bytes), 24 bytes are needed.

However, the example shows that when encrypting the same data twice with the same password and iteration count, the resulting encrypted data is completely different. This result occurs because of the randomly created salt value, which is different for each encryption process and leads to a completely different encryption key.

The example is known as the *password-based key derivation function* PBKDF1 and is described in RFC 2898 (Password-Based Cryptography Specification). Some modifications lead to other PBKDF functions, which can produce keys of any length.

To regenerate the same encryption key at the recipient's site, these extra parameters are often stored with the encrypted data; for example, in some kind of a header structure, which can also contain other control data. The same PBKDF function must be applied to these values on both sides.

## 4.1.9 Public key encryption

Public key encryption does not mean that your data is encrypted by using an RSA public key. This process is far too time-consuming because of the high computational effort for asymmetric encryption. Instead, a symmetric encryption key, which is created randomly or from a password, is used for encrypting your data. Only the encryption key is then encrypted by using the public key of the recipient of the encrypted data.

When the TS1140 tape drive with encryption support is used, the term key encrypting key (KEK) is used for a public key to encrypt a session key.

The encrypting site always uses the public key of the intended receiving site of the encrypted data to protect the data key. The decrypting site uses its private key to decrypt the encryption key. The encrypting and decrypting site can be identical.

z/VSE functions and their used concepts are listed in Table 4-4.

Table 4-4 Cryptographic functions in z/VSE

Function	Media	Concept
Tape drive TS1140	Physical tapes	Public key encryption
Encryption Facility for z/VSE	Physical tapes, virtual tapes, disks	Public key encryption, password-based encryption
SSL/TLS, secure FTP, secure Telnet, SSL with CICS Web Support, SSL/TLS with IBM MQSeries®, and others.	Network	Public key encryption

z/VSE uses a subset of the encryption methods that were described in this section, and which can be implemented in hardware or software. In the following sections, we describe which methods are implemented in hardware and which in software and how z/VSE makes use of them.

## 4.2 Configuring cryptographic hardware

This section describes how access to cryptographic hardware is set up through the Hardware Management Console (HMC) and the Support Element (SE).

In March 2003, VSE/ESA V2R7M0 introduced the support for cryptographic hardware with the PCICA accelerator card. As a result, RSA operations can be performed through the PCICA card, symmetric algorithms, such as DES, TDES (triple-DES), SHA-1, and MD5, still must be run by the software implementations that are provided by TCP/IP for VSE/ESA. Therefore, only the SSL/TLS handshaking process and special functions, such as generating a certificate request, were hardware-accelerated because an RSA operation is performed here only.

Crypto Express2 and the CP Assist for Cryptographic Function (CPACF) feature were then supported since z/VSE V3R1. With the introduction of CPACF, symmetric crypto algorithms were also supported by the hardware. In contrast to crypto cards, CPACF is not a pluggable adapter, but it is a microcode extension, which provides more processor instructions to perform symmetric crypto functions. Therefore, with z/VSE V3R1 and later, the entire SSL/TLS process is supported by crypto hardware.

As of this writing, z/VSE only supports clear-key cryptography where all types of keys are stored in the operating system's file system. Cryptographic operations occur in processor main storage, which is a possible security leak. For secure-key cryptography, use Linux on Z or z/OS.

We use the following hardware and software in our examples:

- ▶ IBM z14™, SE 2.14.0
- ▶ z/VSE V6R2M0 running in an LPAR
- ▶ Multiple Crypto Express features assigned to the z/VSE LPAR
- ▶ TCP/IP for z/VSE V2.2 as part of z/VSE V6R2 GA version

## 4.2.1 Hardware overview

Cryptographic hardware can be pluggable crypto cards and features that are provided by processor chips in the machine.

### Crypto cards

The evolution of crypto cards on IBM mainframes started in the early 2000s with the Peripheral Component Interconnect® (PCI) Cryptographic Coprocessor (PCICC) and PCI Cryptographic Accelerator (PCICA), which were available for IBM zSeries 800 (z800) and z900 mainframes. They were followed by the PCIX Cryptographic Coprocessor (PCIXCC) on z990 and the first Crypto Express cards on z890/z990 and z9.

The current generation of Crypto Express (CEX) features is the Crypto Express6S (CEX6S), which is available for the IBM z14 and are shown in Figure 4-4.

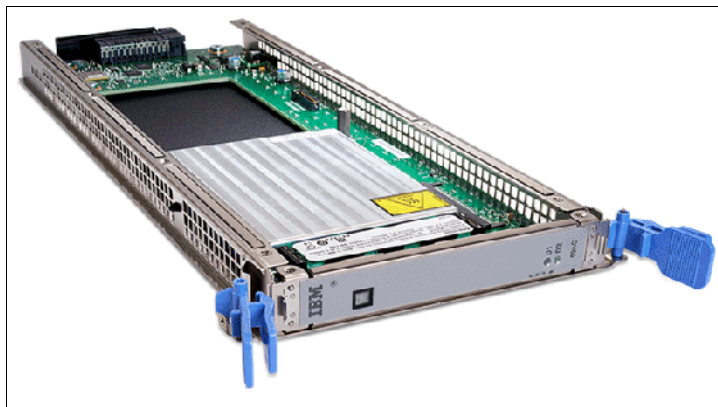


Figure 4-4 Crypto Express6S

The first Crypto Express cards, up to the Crypto Express3 (CEX3), included two independent crypto processors (APs) on one card, where Crypto Express4S (CEX4S) onwards provide only one AP. However, the number of CEX4S, 5S, and 6S features per mainframe is twice as many as for the earlier cards.

Figure 4-4 also shows a Crypto Express6S, which is supported by z/VSE V5R2M0 with APAR DY47715, z/VSE V6R1M0 with DY47716, and z/VSE V6R2M0 GA version.

Each card can be configured by using the HMC in the following modes, where each mode provides special functionality:

- ▶ Common Cryptographic Architecture (CCA) coprocessor mode. The CCA coprocessor provides the full set of functionality, including support for secure master keys in the hardware. It can perform RSA sign/verify, RSA key generation, true random number generation, and more. Cards that are configured in CCA coprocessor mode are named CEX6C.
- ▶ Accelerator mode. The accelerator mode was specially designed for RSA acceleration. A Crypto Express2 card in accelerator mode can perform RSA sign/verify approximately three times faster than in CCA coprocessor mode. This performance gain decreased with each new card generation and higher RSA key lengths. Cards that are configured in accelerator mode are named CEX6A.
- ▶ EP11 coprocessor mode. The EP11 mode was introduced with the Crypto Express4S and provides a Public Key Cryptography Standards (PKCS) #11 programming interface. As of this writing, it is not used on z/VSE. Cards configured in EP11 mode are named CEX6P.

Crypto requests are sent to an AP by way of one of these queues, and replies are sent back from the card on the same queue. The number of AP queues depends on the mainframe model.

Up to the IBM zEnterprise® EC12 (zEC12) and zEnterprise BC12 (zBC12), the number of AP queues is always 16, where with the IBM z13®, the number increased to 85. You can serve up to 85 LPARs concurrently with one card. You assign AP queues to logical partitions (LPARs) using the HMC panels.

Access to crypto cards is possible through the z/VSE crypto device driver only. The application programming interface (API) is available for vendors.

For more information about IBM z14, see *IBM z14 Configuration Setup*, SG24-8460.

## CPACF

CPU Assist for Cryptographic Function (CPACF) is a microcode extension to the processor. It has the feature code 3863. CPACF provides more instructions to perform symmetric crypto functions, such as DES, TDES, AES, hash functions, and random number generators (PRNG, DBRG). A set of query functions are available that indicate the availability of particular algorithms dependent on the processor.

For more information about CPACF instructions, see *z/Architecture Principles of Operation*, SA22-7832, which is available online at:

<http://www.ibm.com/systems/z/os/zvse/documentation/#vse>

**Note:** Feature code 3863 must be enabled as the prerequisite to use CPACF functions. The CPACF is enabled by using the appropriate *CPACF enablement* diskette. One diskette can enable the CPACF on all the processors on each of the nodes in the system.

## 4.2.2 Planning your crypto configuration

On z/OS, CCA coprocessors are used for processing specific functions that are available in CCA coprocessor mode only; for example, secure key functions and banking functions, such as PIN verification. On z/VSE, these coprocessor functions are not used. z/VSE uses the accelerator mode for processing RSA encryptions and decryptions, which are started during an SSL handshake. CCA coprocessors are also used for RSA acceleration, random number generation, and Elliptic-Curve / Diffie-Hellman.

When z/VSE runs in an LPAR, we recommend assigning one accelerator and one CCA coprocessor to the z/VSE LPAR. When running under z/VM, we recommend the use of APDEDICATE to assign one accelerator and one CCA coprocessor to the z/VSE guest. With this setup, the VSE cryptographic device driver always uses the best device for a request.

Figure 4-5 on page 105 shows a scenario with several cards getting accessed from different operating systems. The z/OS LPAR has access to APs 1, 2, 3, and 4 through cryptographic domain (AP queue) 0. z/VSE has access to APs 2, 3, 5, and 6 through cryptographic domain 15. Therefore, z/OS and z/VSE share the crypto devices 2 and 3, but z/OS uses AP queue 0 to enqueue crypto requests to the device but z/VSE uses AP queue 15. The assignment of LPAR3 and LPAR4 is not shown in detail.



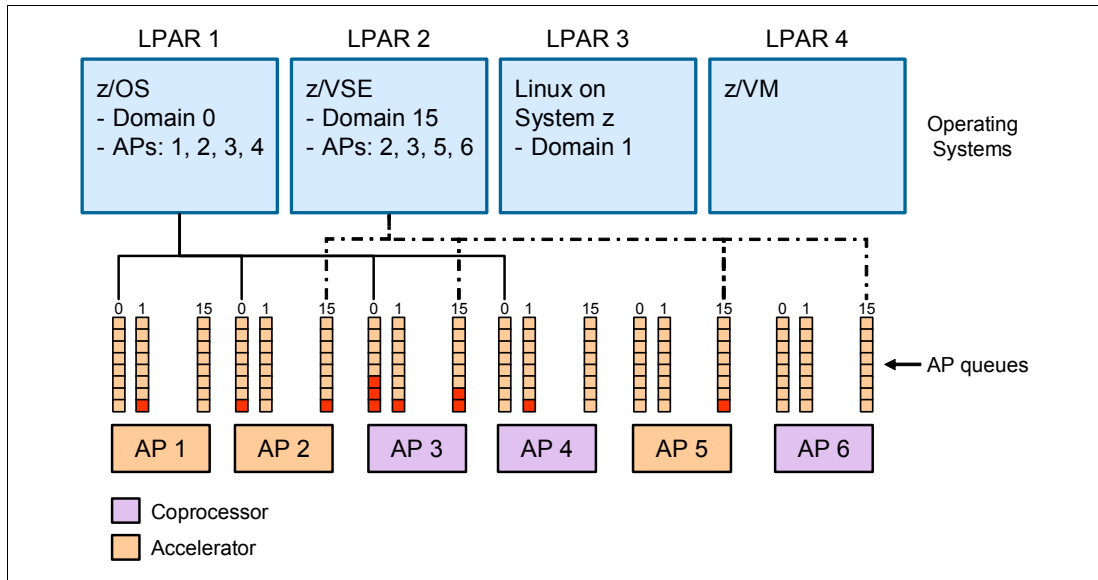


Figure 4-5 Assignment of APs to LPARs

The following section describes how to set up access to cryptographic hardware through the HMC and the SE.

### 4.2.3 Configuring LPAR activation profile

Crypto cards and CPACF do not require any hardware configuration in z/VSE; therefore, they do not have an ADD statement in the IPLPROC and no Interactive Interface dialogs are used to define them. Crypto hardware is sensed during IPL, or through operator command, and used transparently to applications.

The following sections describe how to view the machine's cryptographic configuration and how to assign crypto devices to a specific LPAR.

#### Setting up the cryptographic domain

The term *cryptographic domain* is a synonym to the term *AP queue* and denotes an input queue that is provided by a crypto card for receiving requests. Each AP queue can hold up to eight elements until it is full. Setting up the cryptographic domain for an LPAR is the first task you must complete to provide access to crypto cards.

In the first step, we access the machine's main cryptographic configuration, which is defined in the LPAR activation profile.

After logging on to the Support Element, select your LPAR and select **Operational Customization** → **Customize/Delete Activation Profiles**, as shown in Figure 4-6 on page 106.

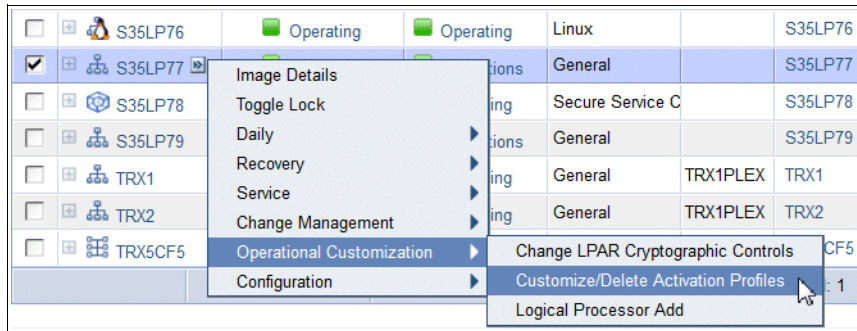


Figure 4-6 Setting Customize/delete Activation Profiles option

The Customize/Delete Activation Profiles List panel opens, as shown in Figure 4-7.

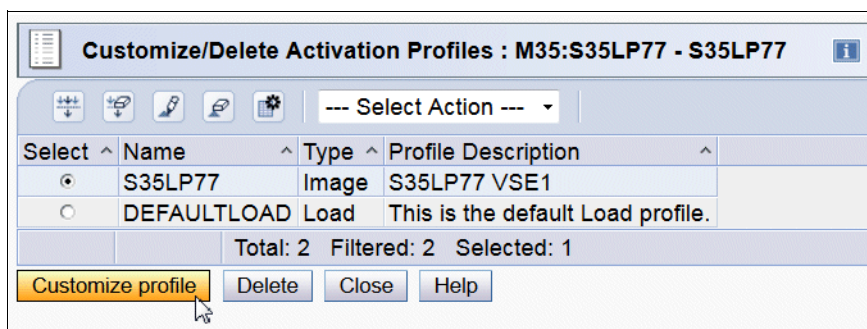


Figure 4-7 Customize/delete activation profiles list

Select the profile that you want to customize and click **Customize profile**.

On the next panel, select **Crypto**.

The following controls are configured here:

- ▶ Control Domain Index
 

This control identifies the domains that can be administered from this LPAR when it is acting as a TKE host. As of this writing, TKEs are not supported by z/VSE.
- ▶ Usage Domain Index
 

This control is the AP queue number that is used by this LPAR to access crypto devices.
- ▶ Cryptographic Candidate List
 

This control indicates the AP numbers of the crypto devices that are eligible to be accessed by this LPAR.
- ▶ Cryptographic Online List
 

This control indicates AP numbers of the crypto devices that are put online at LPAR activation.

In the example, the z/VSE LPAR accesses crypto devices at AP 0 - 7 and 11 by way of AP queue (crypto domain) 82. All crypto devices are put online at LPAR activation because they are in the Candidate and Online list, as shown in Figure 4-8 on page 107.

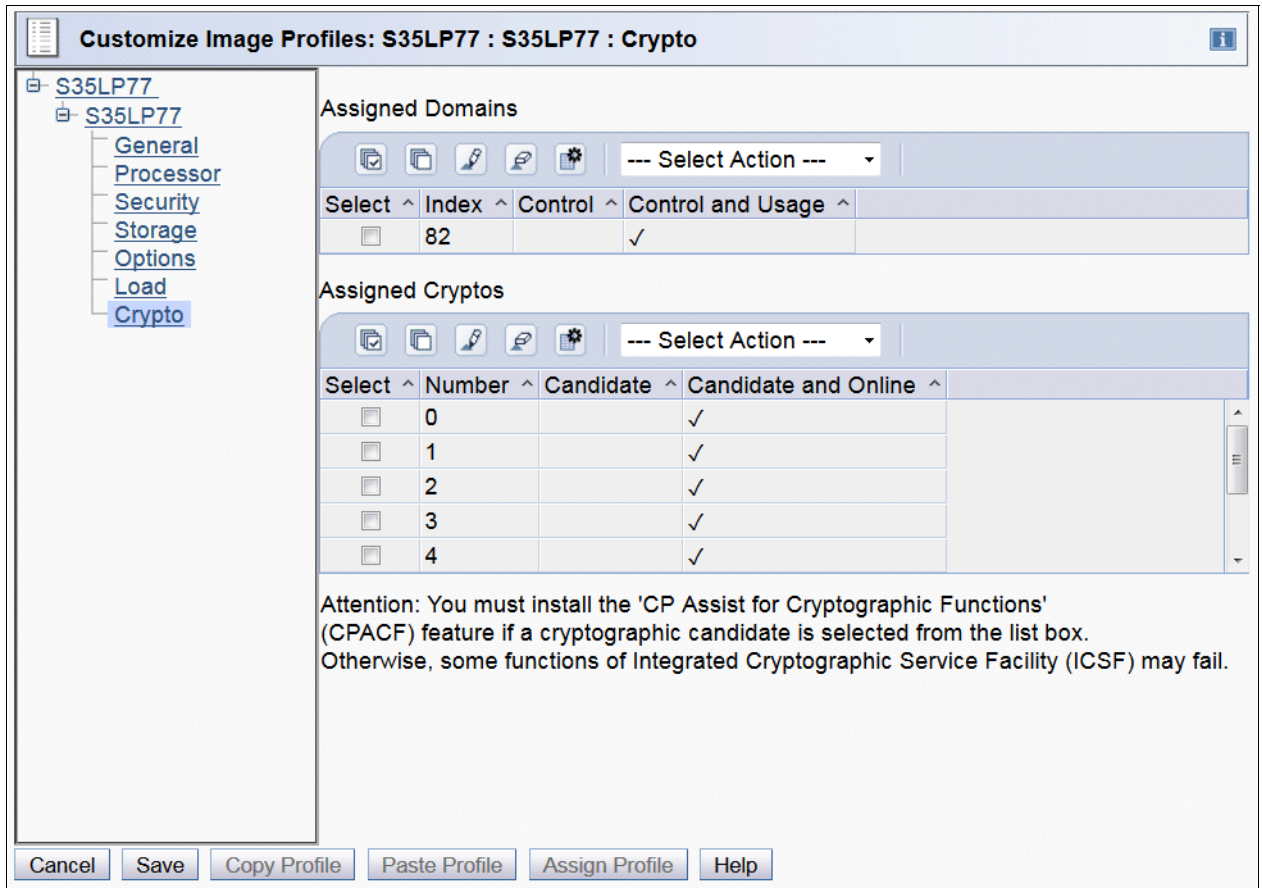


Figure 4-8 Customize image profiles

Click **Save** to continue.

#### 4.2.4 CPC cryptographic configuration

In the next step, we view the cryptographic configuration of the entire machine. Open the System details menu of the CPC at the SE workplace, as shown in Figure 4-9.

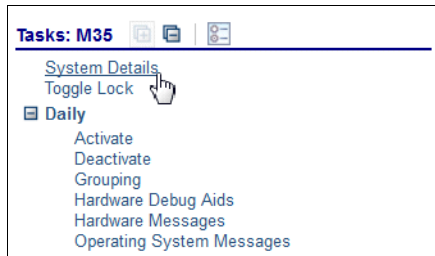


Figure 4-9 Select System Details in the SE workplace

In the System Details window, you can verify that the CPACF feature is installed (see Figure 4-9 on page 107).

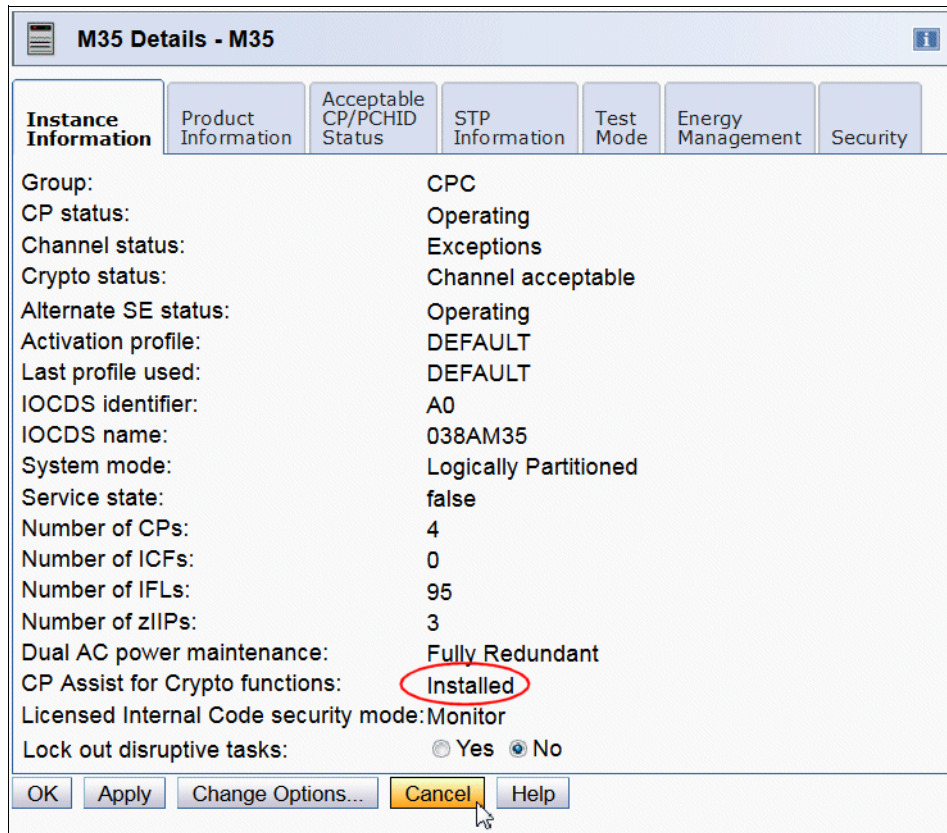


Figure 4-10 Instance information

Click **Cancel**.

In the SE workplace, click **Cryptos**, as shown in Figure 4-11, to display the available crypto devices.

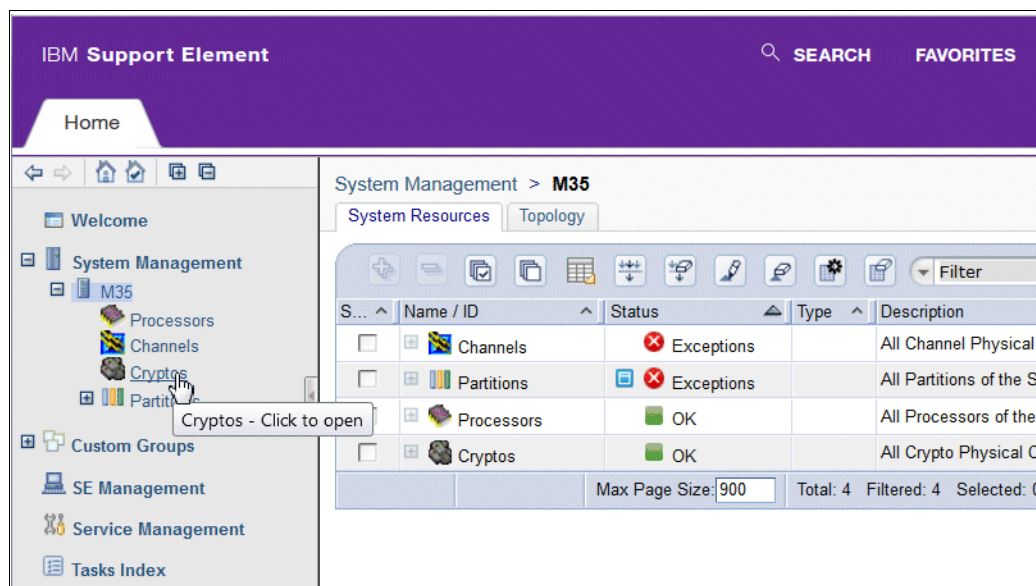


Figure 4-11 Cryptos in the CPC Work Area

On this machine, a mix of 12 Crypto Express5S and 6S cards is used. Although the panel that is shown in Figure 4-12 shows all crypto cards that are available on this machine, in the next section we view the crypto cards that are available from the z/VSE LPAR.

Select	PCHID	ID	Status	State	Location	Type
<input type="checkbox"/>	0144	00	Operating	Online	Z22B-LG21	Crypto Express5S
<input type="checkbox"/>	0150	01	Operating	Online	Z22B-LG25	Crypto Express5S
<input type="checkbox"/>	01C4	02	Operating	Online	Z15B-LG21	Crypto Express5S
<input type="checkbox"/>	01D0	03	Operating	Online	Z15B-LG25	Crypto Express5S
<input type="checkbox"/>	0284	04	Operating	Online	Z01B-LG02	Crypto Express6S
<input type="checkbox"/>	021C	05	Operating	Online	Z08B-LG09	Crypto Express6S
<input type="checkbox"/>	0250	06	Operating	Online	Z08B-LG25	Crypto Express6S
<input type="checkbox"/>	02EC	07	Operating	Online	Z01B-LG33	Crypto Express6S
<input type="checkbox"/>	015C	08	Operating	Online	Z22B-LG28	Crypto Express6S
<input type="checkbox"/>	01C8	09	Operating	Online	Z15B-LG22	Crypto Express6S
<input type="checkbox"/>	01CC	10	Operating	Online	Z15B-LG23	Crypto Express6S
<input type="checkbox"/>	01DC	11	Operating	Online	Z15B-LG28	Crypto Express6S

Max Page Size: 900 Total: 12 Filtered: 12 Selected: 0

Figure 4-12 Primary Support Element Workplace showing cryptos

## 4.2.5 LPAR cryptographic configuration

On the SE workplace, select your LPAR and click **Cryptos**, as shown Figure 4-13.

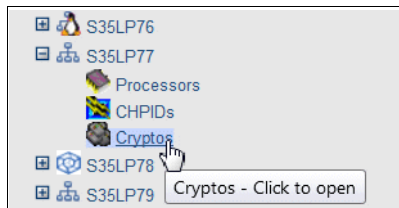


Figure 4-13 Select Cryptos of your LPAR

The list of LPAR cryptos shows the 9 out of 12 available crypto devices of this LPAR, (see Figure 4-14 on page 110).



System Management > M35 > Partitions > S35LP77 > Cryptos

Cryptos Topology

Select	ID	PCHID	Status	State	Type
<input type="checkbox"/>	00	0144	Operating	Online	Crypto Express5S Accelerator
<input type="checkbox"/>	01	0150	Operating	Online	Crypto Express5S CCA Coprocessor
<input type="checkbox"/>	02	01C4	Operating	Online	Crypto Express5S CCA Coprocessor
<input type="checkbox"/>	03	01D0	Operating	Online	Crypto Express5S EP11 Coprocessor
<input type="checkbox"/>	04	0284	Operating	Online	Crypto Express6S Accelerator
<input type="checkbox"/>	05	021C	Operating	Online	Crypto Express6S CCA Coprocessor
<input type="checkbox"/>	06	0250	Operating	Online	Crypto Express6S EP11 Coprocessor
<input type="checkbox"/>	07	02EC	Operating	Online	Crypto Express6S CCA Coprocessor
<input type="checkbox"/>	11	01DC	Operating	Online	Crypto Express6S EP11 Coprocessor

Max Page Size: 900 Total: 9 Filtered: 9 Selected: 0

Figure 4-14 Cryptos Work Area

## 4.2.6 Hardware crypto device driver in z/VSE

This section describes how z/VSE recognizes cryptographic hardware and how crypto-related information can be displayed on the operator console.

The z/VSE crypto device driver runs as a subtask IJBCRYPT of the job SECSERV in the security server partition FB, by default<sup>3</sup>. It is part of the Basic Security Manager (BSM). Cryptographic hardware is sensed during IPL when the BSM is initialized. When no crypto cards are assigned, the device driver finishes processing and the subtask is stopped. When at least one usable crypto card is available, the device driver keeps running.

You can obtain a list of available hardware crypto related operator commands, as shown in Example 4-1.

### Example 4-1 Crypto device driver help command output

```
msg fb,data=help
...
FB 0011 HARDWARE CRYPTO COMMANDS:
FB 0011  APBUSY=NN .....: SET AP CRYPTO WAIT ON BUSY (0..99)
FB 0011  APRETRY=NN .....: SET AP CRYPTO RETRY COUNT (0..99)
FB 0011  APREM AP=N1 (N2 N3..): REMOVE (DISABLE) ONE OR MORE APS
FB 0011  APADD AP=N1 (N2 N3..): ADD (ENABLE) ONE OR MORE APS
FB 0011  APQUE .....: SHOW STATUS OF ASSIGNED AP QUEUE
FB 0011  APHIST (AP=NN) .....: SHOW HISTORY OF ONE OR ALL APS
FB 0011  APWAIT=NN .....: SET AP CRYPTO POLLING TIME (0..99)
FB 0011  APSENSE .....: START SENSING OF CRYPTO HARDWARE
FB 0011  APTTRACE=N .....: SET AP CRYPTO TRACE LEVEL (0..3)
FB 0011  APEAI .....: ENABLE AP-QUEUE INTERRUPTS
FB 0011  APDAI .....: DISABLE AP-QUEUE INTERRUPTS
FB 0011  APSTAT AP=NN .....: DISPLAY CRYPTO ADAPTER STATUS
FB 0011  ( STATCARD | .....: OF THE CARDS FIRMWARE (DEFAULT), OR
FB 0011  STATCCA ) .....: OF THE CARDS CCA LEVEL
FB 0011  APCLEAR .....: CLEAR INTERNAL STATS COUNTERS
FB 0011  APTERM .....: TERMINATE DEVICE DRIVER
```

<sup>3</sup> For more information about security server, see 2.1.3, "Security server partition" on page 15.

The commands are described in *z/VSE Administration*, SC34-2692. For more information about crypto support when running as guest under z/VM, see 4.2.8, “Cryptography for guests on z/VM”. For more information about crypto support when an External Security Manager (ESM) is used, see 4.2.9, “Cryptography when using an external security manager” on page 124.

Example 4-2 shows the messages that are issued when IPLing z/VSE with the crypto hardware.

*Example 4-2 Crypto hardware messages during z/VSE startup*

---

```
FB 0115 1J022I CPU CRYPTOGRAPHIC ASSIST FEATURE AVAILABLE.
FB 0115 1J054I FOUND A CRYPTO EXPRESS5S CARD AT AP 0
FB 0115 1J054I FOUND A CRYPTO EXPRESS5S CARD AT AP 1
FB 0115 1J054I FOUND A CRYPTO EXPRESS5S CARD AT AP 2
FB 0115 1J054I FOUND A CRYPTO EXPRESS5S CARD AT AP 3
FB 0115 1J054I FOUND A CRYPTO EXPRESS6S CARD AT AP 4
FB 0115 1J054I FOUND A CRYPTO EXPRESS6S CARD AT AP 5
FB 0115 1J054I FOUND A CRYPTO EXPRESS6S CARD AT AP 6
FB 0115 1J054I FOUND A CRYPTO EXPRESS6S CARD AT AP 7
FB 0115 1J054I FOUND A CRYPTO EXPRESS6S CARD AT AP 11
FB 0115 1J005I HARDWARE CRYPTO DEVICE DRIVER INITIALIZED SUCCESSFULLY.
FB 0115 1J006I USING AP QUEUE 82
```

---

Example 4-3 shows the messages that are displayed on a system where no crypto cards are installed.

*Example 4-3 Crypto hardware messages with CPACF but no crypto cards*

---

```
FB 0095 1J022I CPU CRYPTOGRAPHIC ASSIST FEATURE AVAILABLE.
FB 0095 1J017I CRYPTO HARDWARE NOT INSTALLED OR NOT DEFINED.
```

---

The **STATUS** command of the SECSERV job can be used to view the device driver status.

Example 4-4 shows the nine assigned APs (crypto cards) that are accessible through AP queue (crypto domain) 82 as configured in section “Setting up the cryptographic domain” on page 105.

*Example 4-4 Output of the security server command STATUS=CR*

---

```
msg fb,data=status=cr
AR 0015 1I40I  READY
FB 0011 BST223I CURRENT STATUS OF THE SECURITY TRANSACTION SERVER:
FB 0011 CRYPTO DEVICE DRIVER STATUS:
FB 0011  AP CRYPTO SUBTASK STARTED ..... : YES
FB 0011  MAX REQUEST QUEUE SIZE ..... : 0
FB 0011  MAX PENDING QUEUE SIZE ..... : 0
FB 0011  TOTAL NO. OF AP REQUESTS ..... : 0
FB 0011  NO. OF POSTED CALLERS ..... : 0
FB 0011  AP-QUEUE INTERRUPTS AVAILABLE ..... : YES
FB 0011  AP-QUEUE INTERRUPTS STATUS ..... : DISABLED
FB 0011  AP CRYPTO POLLING TIME (1/300 SEC).. : 1
FB 0011  AP CRYPTO WAIT ON BUSY (1/300 SEC).. : 75
FB 0011  AP CRYPTO RETRY COUNT ..... : 5
FB 0011  AP CRYPTO TRACE LEVEL ..... : 3
FB 0011  TOTAL NO. OF WAITS ON BUSY ..... : 0
FB 0011  CURRENT REQUEST QUEUE SIZE ..... : 0
```

```

FB 0011  CURRENT PENDING QUEUE SIZE ..... : 0
FB 0011  ASSIGNED APS : CEX2C / CEX2A ..... : 0 / 0
FB 0011                CEX3C / CEX3A ..... : 0 / 0
FB 0011                CEX4C / CEX4A / CEX4P : 0 / 0 / 0
FB 0011                CEX5C / CEX5A / CEX5P : 2 / 1 / 1
FB 0011                CEX6C / CEX6A / CEX6P : 2 / 1 / 2
FB 0011  AP 0 : CEX5A  - ONLINE
FB 0011  AP 1 : CEX5C  - ONLINE
FB 0011  AP 2 : CEX5C  - ONLINE
FB 0011  AP 3 : CEX5P  - ONLINE
FB 0011  AP 4 : CEX6A  - ONLINE
FB 0011  AP 5 : CEX6C  - ONLINE
FB 0011  AP 6 : CEX6P  - ONLINE
FB 0011  AP 7 : CEX6C  - ONLINE
FB 0011  AP 11 : CEX6P - ONLINE
FB 0011  ASSIGNED AP QUEUE (CRYPTO DOMAIN)... : 82
FB 0011  NO. OF AVAILABLE CRYPTO DOMAINS .... : 85
FB 0011  END OF CRYPTO DEVICE DRIVER STATUS

```

---

The following other important values are shown in the example's STATUS output:

► **MAX REQUEST QUEUE SIZE**

This value is the maximum number of RSA requests that are waiting to be processed by the crypto device driver since its last restart. Multiple z/VSE applications can issue RSA requests simultaneously, even from multiple TCP/IP stacks on the same z/VSE system. The request queue serializes the requests for processing by the device driver.

► **MAX PENDING QUEUE SIZE**

This value is the maximum number of requests currently being processed by a crypto card. In some situations, the device driver might be under heavy load by incoming new requests so that replies cannot be immediately returned to callers. A crypto card might be busy so that some time interval exists until a reply can be dequeued.

► **TOTAL NO. OF AP REQUESTS**

This value is the total number of processed RSA requests since the device driver was started.

► **NO. OF POSTED CALLERS**

This value is the total number of callers that received a reply. Normally, this value is identical to the total number of AP requests, unless a reply cannot be delivered, such as when the calling application ends.

► **AP CRYPTO POLLING TIME**

This value is the time interval between enqueueing an RSA request block to a crypto card and the first attempt to dequeue a response. The time interval is specified in 1/300th seconds. The default is 1/300th second. Higher values decrease processor load, but increase elapsed job time; lower values increase processor load and decrease elapsed job time.

## 4.2.7 Disabling a crypto device

Disabling a crypto device has the following meanings:

- Disabling the device in z/VSE, which means that the related device is no longer used by z/VSE, but its hardware status remains unchanged.



- ▶ Disabling the device for your LPAR by way of the HMC or SE. For more information, see “Disabling the device for your LPAR” on page 114.
- ▶ Disabling the device for the entire machine, which affects all LPARs that are using this device. For example, this configuration is necessary when changing the device type from accelerator to coprocessor, which is described in “Disabling the device for the entire machine” on page 117.

These options are described next.

### Disabling the device in z/VSE

If you no longer want to use one or more devices in z/VSE, use the crypto device driver’s APREM command, as shown in Example 4-5. For more than one AP, specify the AP numbers separated by spaces. For more information about all crypto commands, see *z/VSE Administration*, SC34-2692.

*Example 4-5 Removing APs from use by z/VSE*

---

```
msg fb,data=aprem ap=0 1 2
AR 0015 1I40I  READY
FB 0011 1J026I AP 0  DISABLED SUCCESSFULLY.
FB 0011 1J026I AP 1  DISABLED SUCCESSFULLY.
FB 0011 1J026I AP 2  DISABLED SUCCESSFULLY.
```

---

The crypto device driver ensures that any pending replies are still delivered, but the devices are no longer used for further requests. When you now issue the **STATUS=CR** command, these APs appear as “disabled by operator”, as shown in Example 4-6.

*Example 4-6 STATUS=CR output showing disabled devices*

---

```
msg fb,data=status=cr
...
FB 0011      AP 0 : CEX5A  - DISABLED BY OPER
FB 0011      AP 1 : CEX5C  - DISABLED BY OPER
FB 0011      AP 2 : CEX5C  - DISABLED BY OPER
FB 0011      AP 3 : CEX5P  - ONLINE
FB 0011      AP 4 : CEX6A  - ONLINE
FB 0011      AP 5 : CEX6C  - ONLINE
FB 0011      AP 6 : CEX6P  - ONLINE
FB 0011      AP 7 : CEX6C  - ONLINE
```

---

To show any pending replies, you can use the **APQUE** command, as shown in Example 4-7

*Example 4-7 View pending replies*

---

```
msg fb,data=apque
AR 0015 1I40I  READY
FB 0011 1J045I NUMBER OF REQUESTS BEING PROCESSED BY AP QUEUE 82:
FB 0011  AP 0 : 0
FB 0011  AP 1 : 0
FB 0011  AP 2 : 0
FB 0011  AP 3 : 0
FB 0011  AP 4 : 0
FB 0011  AP 5 : 0
FB 0011  AP 6 : 0
FB 0011  AP 7 : 0
```

---

For reenabling these APs, use the **APADD** command in the same way as APREM.

**Hint:** When your z/VSE is running and crypto workload is present, use the APREM command to first remove an AP from use by z/VSE before turning the AP off by way of the SE or HMC.

### Disabling the device for your LPAR

To disable a crypto device for your LPAR, select your LPAR on the SE workplace and click **Cryptos**. Then, select the device in the Cryptos list, and select **Crypto Service Operations** → **Configure On/Off**, as shown in Figure 4-15.

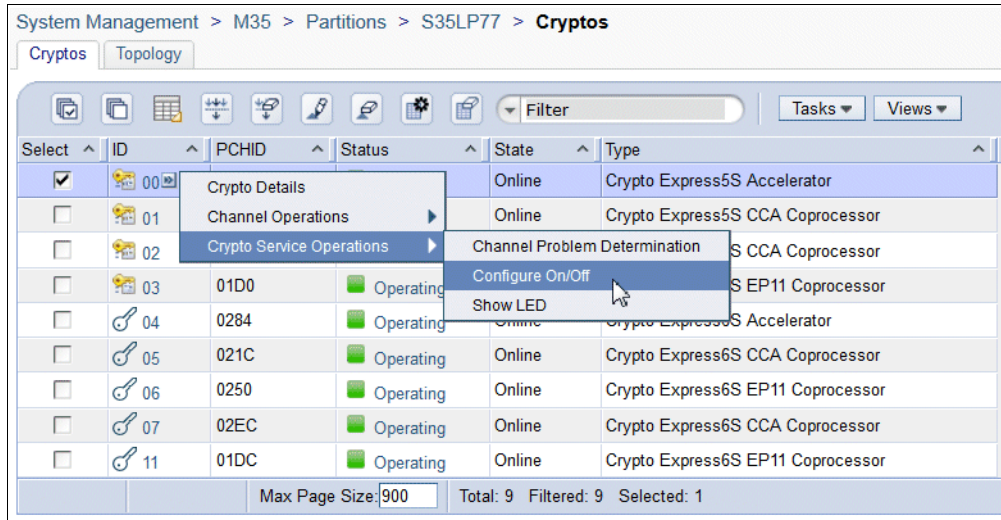


Figure 4-15 Select Configure On/Off

This operation disables the crypto device for this specific LPAR only. The device might still be online in other LPARs. Therefore, the process of enabling or disabling a device is relatively fast, compared to the time required for activating a device the first time after a machine started.

In the Configure On/Off panel, select **Toggle** from the Select Action drop-down menu, as shown in Figure 4-16.

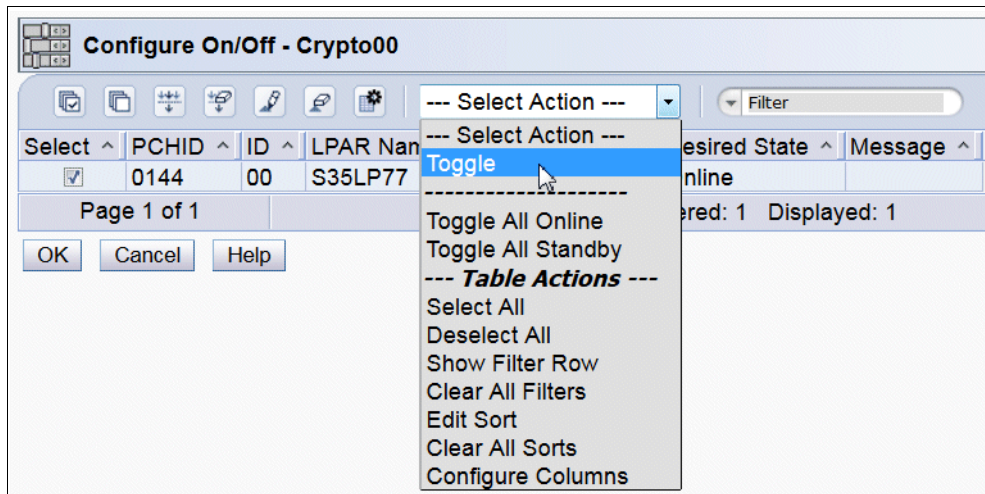


Figure 4-16 Configure Channel Path On/Off - change Desired State

The wanted state changes to Standby. Click **OK** to close this window and confirm that you want to apply your change, as shown in Figure 4-17.

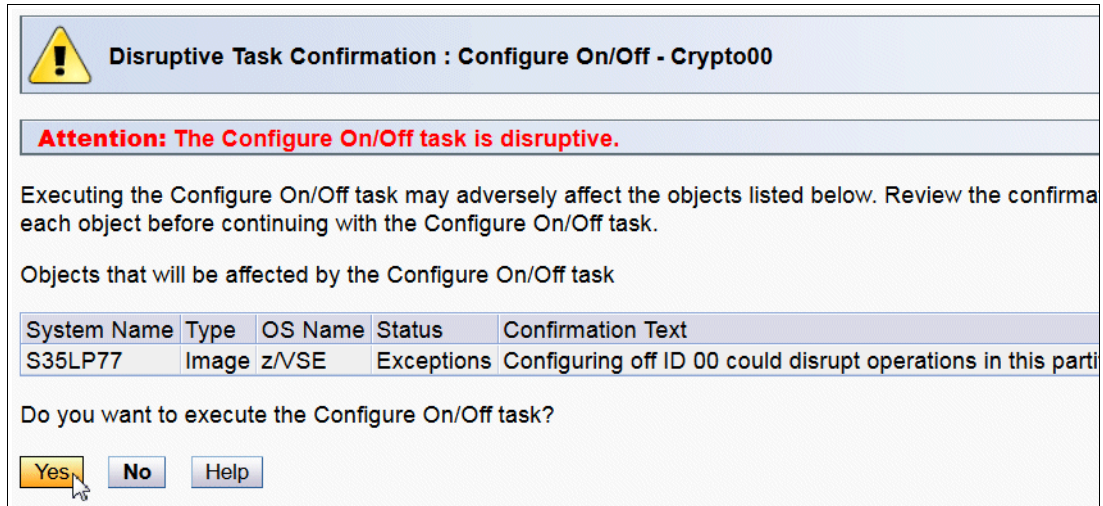


Figure 4-17 Configure Channel Path On/Off - apply new Desired State

After the operation completes, click **OK** to close the Configure Channel Path On/Off Progress panel, as shown in Figure 4-18.

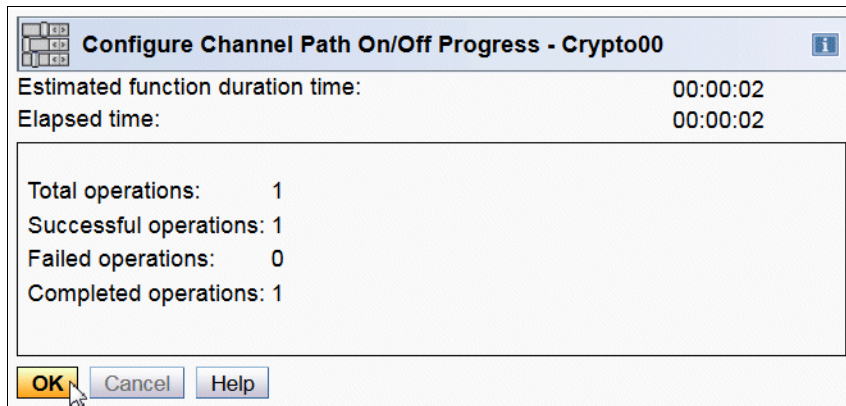


Figure 4-18 Configure Channel Path On/Off Progress

Figure 4-19 shows that the device is now stopped.

Select	ID	PCHID	Status	State	Type
<input checked="" type="checkbox"/>	00	0144	Stopped	Standby	Crypto Express5S Accelerator
<input type="checkbox"/>	01	0150	Operating	Online	Crypto Express5S CCA Coprocessor
<input type="checkbox"/>	02	01C4	Operating	Online	Crypto Express5S CCA Coprocessor
<input type="checkbox"/>	03	01D0	Operating	Online	Crypto Express5S EP11 Coprocessor
<input type="checkbox"/>	04	0284	Operating	Online	Crypto Express6S Accelerator
<input type="checkbox"/>	05	021C	Operating	Online	Crypto Express6S CCA Coprocessor
<input type="checkbox"/>	06	0250	Operating	Online	Crypto Express6S EP11 Coprocessor
<input type="checkbox"/>	07	02EC	Operating	Online	Crypto Express6S CCA Coprocessor
<input type="checkbox"/>	11	01DC	Operating	Online	Crypto Express6S EP11 Coprocessor

Figure 4-19 Cryptos Work Area shows new status

You can now run the **apsense** command to refresh the crypto configuration in z/VSE, as shown in Example 4-8.

*Example 4-8 Security server command APSENSE*

```
msg fb,data=apsense
AR 0015 1I40I  READY
FB 0115 1J022I CPU CRYPTOGRAPHIC ASSIST FEATURE AVAILABLE.
FB 0115 1J054I FOUND A CRYPTO EXPRESS5S CARD AT AP 1
FB 0115 1J054I FOUND A CRYPTO EXPRESS5S CARD AT AP 2
FB 0115 1J054I FOUND A CRYPTO EXPRESS5S CARD AT AP 3
FB 0115 1J054I FOUND A CRYPTO EXPRESS6S CARD AT AP 4
FB 0115 1J054I FOUND A CRYPTO EXPRESS6S CARD AT AP 5
FB 0115 1J054I FOUND A CRYPTO EXPRESS6S CARD AT AP 6
FB 0115 1J054I FOUND A CRYPTO EXPRESS6S CARD AT AP 7
FB 0115 1J054I FOUND A CRYPTO EXPRESS6S CARD AT AP 11
FB 0115 1J031I HARDWARE CRYPTO DEVICE DRIVER REFRESHED.
```

AP 0 is no longer recognized and the **status** output also no longer shows AP 0, as shown in Example 4-9.

*Example 4-9 Output of security server command STATUS=CR*

```
msg fb,data=status=cr
...
FB 0011  ASSIGNED APS : CEX2C / CEX2A ..... : 0 / 0
FB 0011                CEX3C / CEX3A ..... : 0 / 0
FB 0011                CEX4C / CEX4A / CEX4P : 0 / 0 / 0
FB 0011                CEX5C / CEX5A / CEX5P : 2 / 0 / 1
FB 0011                CEX6C / CEX6A / CEX6P : 2 / 1 / 2
FB 0011  AP 1 : CEX5C  - ONLINE
FB 0011  AP 2 : CEX5C  - ONLINE
FB 0011  AP 3 : CEX5P  - ONLINE
FB 0011  AP 4 : CEX6A  - ONLINE
FB 0011  AP 5 : CEX6C  - ONLINE
FB 0011  AP 6 : CEX6P  - ONLINE
```

```

FB 0011    AP 7 : CEX6C  - ONLINE
FB 0011    AP 11 : CEX6P - ONLINE
...

```

### Disabling the device for the entire machine

Now, we put a device offline in all LPARs that can access to it and later reconfigure its device type.

**Attention:** This action is a disruption for all LPARs that have this AP assigned. If it is a CCA or EP11 coprocessor with master keys, these keys are zeroized. In our example, we use AP 11, which is currently an EP11 coprocessor and not actively being used by other LPARs (although other LPARs have this AP assigned).

On the SE workplace, click **Cryptos** and select the device you want to put offline, as shown in Figure 4-20. Then, select **Crypto Service Operations** → **Configure On/Off**.

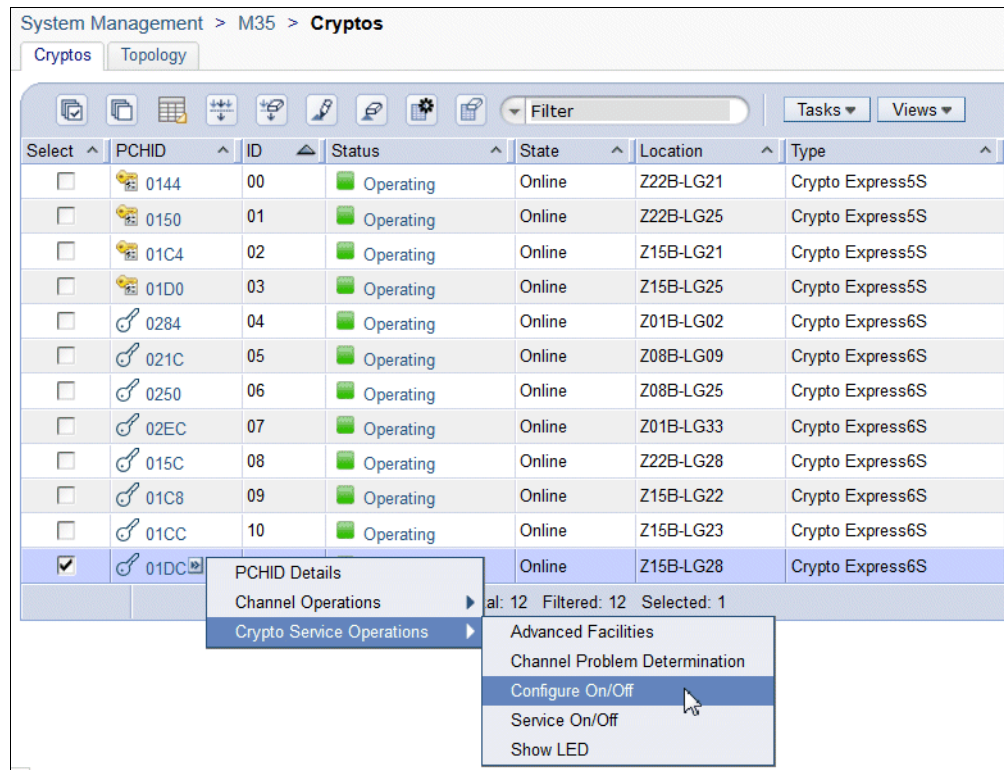


Figure 4-20 Configure On/Off



On the next panel, select **Toggle All Standby** from the Select Action drop-down menu, as shown in Figure 4-21. AP 11 is put in standby mode in all LPARs that use it.

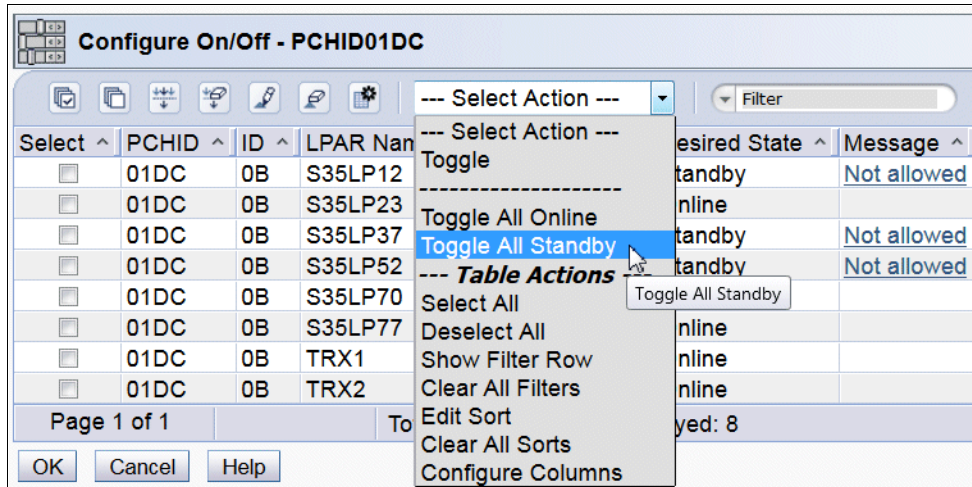


Figure 4-21 Toggle All Standby

The device now appears stopped, as shown in Figure 4-22.

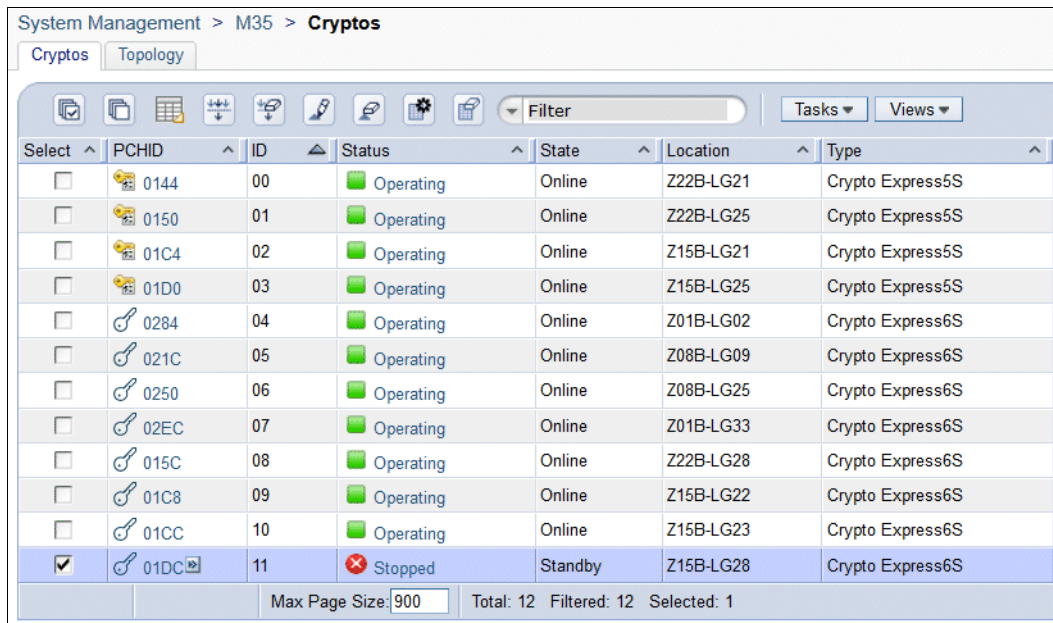


Figure 4-22 AP11 is now stopped

Next, we reconfigure the stopped EP11 coprocessor as accelerator.

## Reconfiguring as accelerator

In the SE workplace, select the CPC icon, and then, select **Configuration** → **Cryptographic Configuration** on the right side of the window, as shown in Figure 4-23.

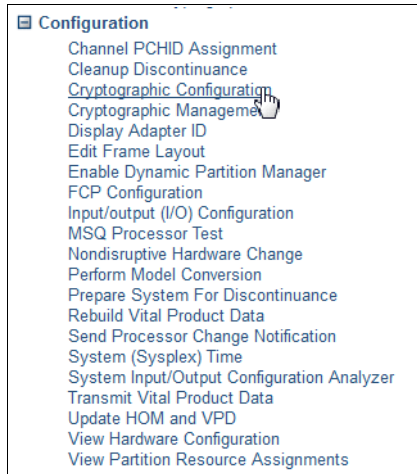


Figure 4-23 Select Cryptographic Configuration

Select the device number and click **Crypto Type Configuration**, as shown in Figure 4-24.

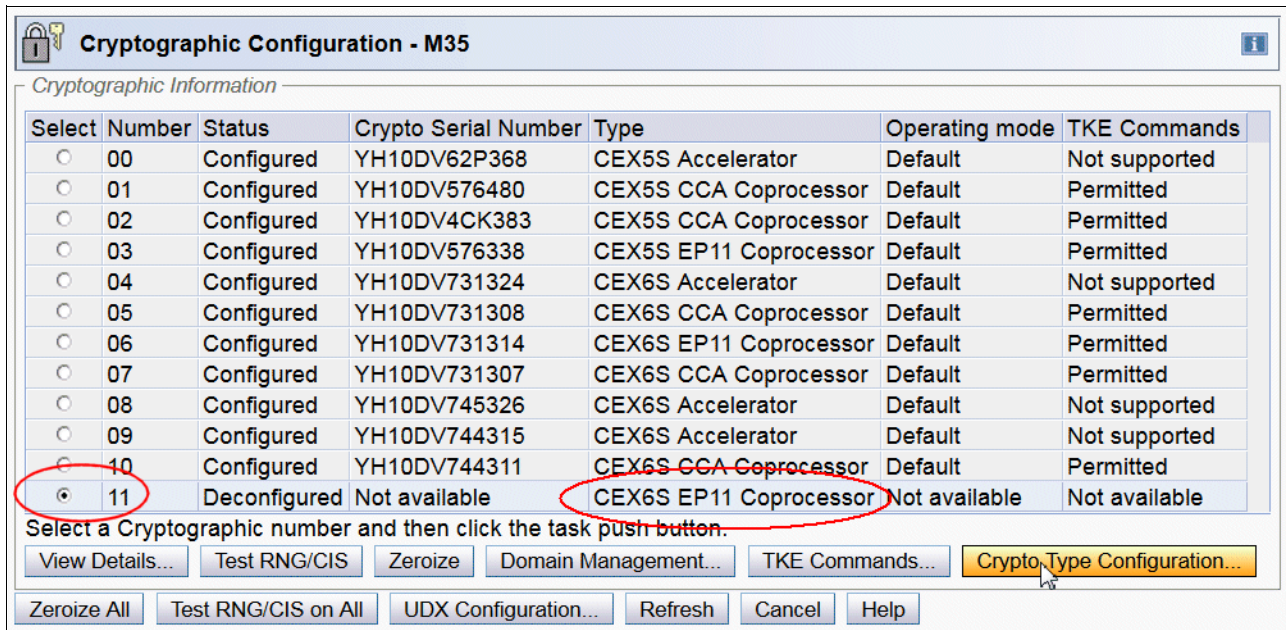


Figure 4-24 Cryptographic Configuration

In the Crypto Type Configuration window, select **Accelerator** and click **OK**, as shown in Figure 4-25.

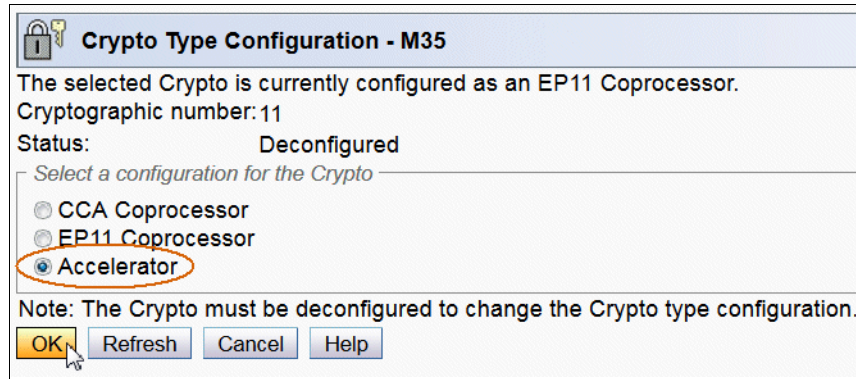


Figure 4-25 Crypto Type Configuration

For z/VSE, you can ignore the warning that is shown in Figure 4-26 because z/VSE does not support secret keys that are stored in the cryptographic hardware. However, for z/OS and Linux on Z, be aware that stored secret keys are lost when performing the reconfiguration.

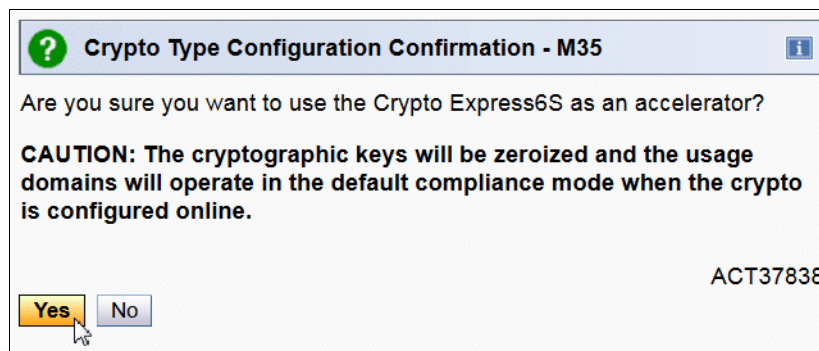


Figure 4-26 Crypto Type Configuration Confirmation

Click **Yes** to continue.

Figure 4-27 shows that accelerator configuration was successful. Click **OK**.

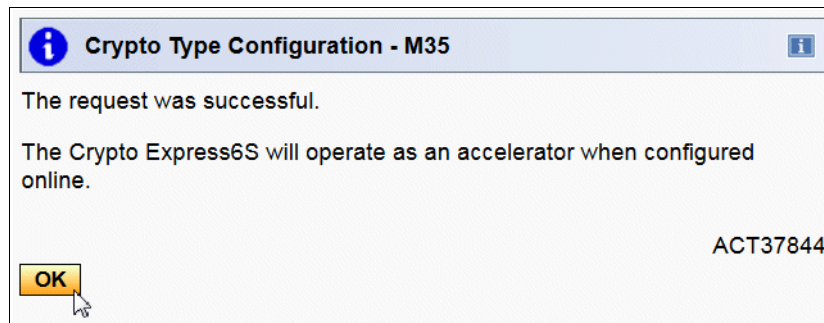


Figure 4-27 Crypto Type Configuration completion

Click **OK** to continue. Close the Crypto Type Configuration window by clicking **Cancel**. Also, close the Cryptographic Configuration window by clicking **Cancel** because all operations are now performed.



## Enabling the device after the mode change

In the SE workplace, click **Cryptos** and select **AP 11** to put it back online, as shown in Figure 4-28. Then, select **Crypto Service Operations** → **Configure On/Off**.

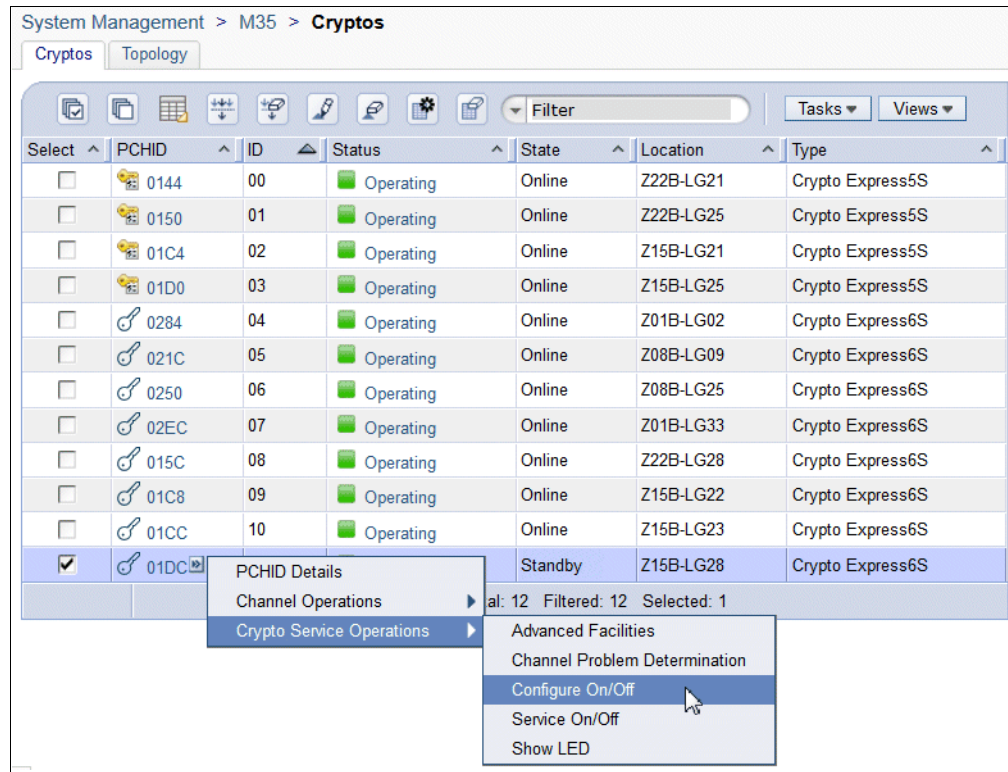


Figure 4-28 Select Cryptos

Toggle the wanted state to **Online** and click **OK**, as shown in Figure 4-29.

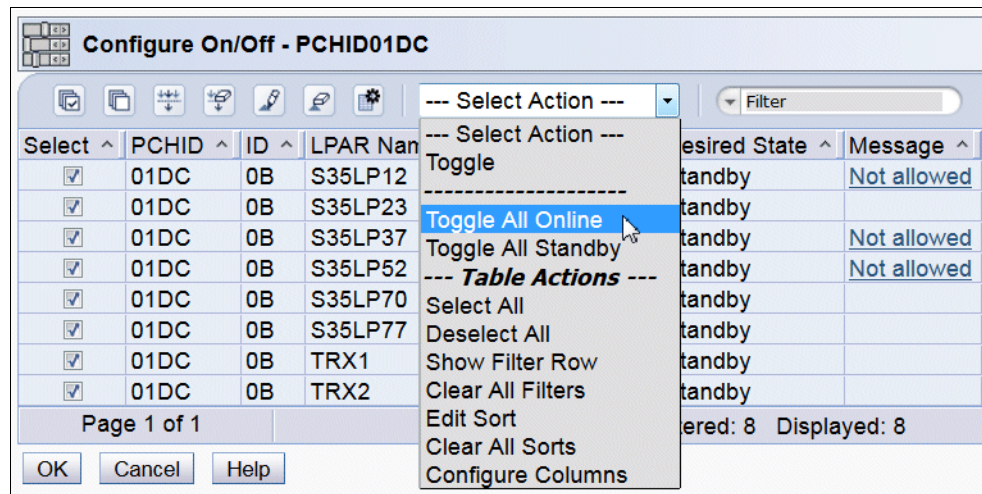


Figure 4-29 Select Configure On/Off

After the device initializes again, it is now usable as an accelerator. The device initialization process can take several minutes because the device is now physically initialized.

Figure 4-30 shows the new status of the device. To display the device status, click the CPC icon in the SE workplace and select **Configuration** → **Cryptographic Configuration**, as shown in Figure 4-23 on page 119.

Select	Number	Status	Crypto Serial Number	Type	Operating mode	TKE C
<input type="radio"/>	00	Configured	YH10DV62P368	CEX5S Accelerator	Default	Not s
<input type="radio"/>	01	Configured	YH10DV576480	CEX5S CCA Coprocessor	Default	Permi
<input type="radio"/>	02	Configured	YH10DV4CK383	CEX5S CCA Coprocessor	Default	Permi
<input type="radio"/>	03	Configured	YH10DV576338	CEX5S EP11 Coprocessor	Default	Permi
<input type="radio"/>	04	Configured	YH10DV731324	CEX6S Accelerator	Default	Not s
<input type="radio"/>	05	Configured	YH10DV731308	CEX6S CCA Coprocessor	Default	Permi
<input type="radio"/>	06	Configured	YH10DV731314	CEX6S EP11 Coprocessor	Default	Permi
<input type="radio"/>	07	Configured	YH10DV731307	CEX6S CCA Coprocessor	Default	Permi
<input type="radio"/>	08	Configured	YH10DV745326	CEX6S Accelerator	Default	Not s
<input type="radio"/>	09	Configured	YH10DV744315	CEX6S Accelerator	Default	Not s
<input type="radio"/>	10	Configured	YH10DV744311	CEX6S CCA Coprocessor	Default	Permi
<input checked="" type="radio"/>	11	Configured	YH10DV747302	CEX6S Accelerator	Default	Not s

Select a Cryptographic number and then click the task push button.

View Details... Test RNG/CIS Zeroize Domain Management... TKE Commands... Crypto Type

Zeroize All Test RNG/CIS on All UDX Configuration... Refresh Cancel Help

Figure 4-30 AP 11 now online as accelerator

To reflect the mode change in z/VSE, you first run the **apsense** command. In Example 4-10, the **status** command shows the new hardware configuration.

Example 4-10 Security server command STATUS=CR

```

msg fb,data=status=cr
...
FB 0011    AP  0 : CEX5A  - ONLINE
FB 0011    AP  1 : CEX5C  - ONLINE
FB 0011    AP  2 : CEX5C  - ONLINE
FB 0011    AP  3 : CEX5P  - ONLINE
FB 0011    AP  4 : CEX6A  - ONLINE
FB 0011    AP  5 : CEX6C  - ONLINE
FB 0011    AP  6 : CEX6P  - ONLINE
FB 0011    AP  7 : CEX6C  - ONLINE
FB 0011    AP 11 : CEX6A  - ONLINE
...

```

AP 11 is now usable by z/VSE as an accelerator.

## 4.2.8 Cryptography for guests on z/VM

When running under z/VM, crypto hardware that is assigned to the z/VM LPAR is not automatically available for guest systems. Even when no crypto cards are available, you should have a CRYPTO directory statement for getting access to CPACF. The CPACF availability depends on the activation of hardware feature code 3863.

Example 4-11 shows how you can use the CMS command `q crypto` to query the hardware crypto settings.

*Example 4-11 Query crypto*

---

```
q crypto
00: Processor 00 Crypto Unit 0 usable
00: Processor 01 Crypto Unit 1 usable
00: There is no user enabled for PKSC Modify
00: All users with directory authorization are enabled for key entry
00: Crypto Adjunct Processor is installed
```

---

Figure 4-11 also shows two crypto cards with APs 0 and 1.

You can use the `q virtual crypto CP` command to query the hardware crypto settings for the z/VSE guest system from the z/VSE operator console, as shown in Example 4-12.

*Example 4-12 Query virtual crypto*

---

```
* cp q virtual crypto
AR 0015 No CAM or DAC Crypto Facilities defined
AR 0015 AP 0E Queue 13 shared
AR 0015 1I40I READY
```

---

In this example, crypto device 0E (14) is available through AP queue 13 in this z/VSE system.

**Notes:** Consider the following points:

- ▶ z/VM hides CCA coprocessor cards from guest systems if accelerator cards are also available. Also, when more than one card of the same type is assigned to the z/VM LPAR, z/VM shows only one card to its guest systems and performs the load balancing.
- ▶ The `q virtual crypto` command only queries the availability of crypto cards. No information is provided about the availability of CPACF.

You can assign a domain to one particular z/VM guest with command, as shown in the following example:

```
CRYPTO DOMAIN 5
```

z/VM virtualizes the assignment of AP queue numbers for guest systems; therefore, it is normal for the z/VSE guest to see a different queue number when it is IPLed. For more information about hardware cryptography support in z/VM, see *CP Planning and Administration*, SC24-6171, which is available at:

<http://www.vm.ibm.com/pubs/hcsg0b01.pdf>

## 4.2.9 Cryptography when using an external security manager

If you use an external security manager and not the BSM, the following implementation details of the hardware crypto support are important and must be observed.

The hardware crypto support is activated by the startup job SECSERV (Security Server), which is part of the BSM. It runs in partition FB by default. If SECSERV is not started because you are using an external security manager, the hardware crypto support is not available. However, the hardware crypto task can be started manually in any partition with a job stream, as shown in Example 4-13.

### Example 4-13

---

```
* $$ JOB JNM=IJBHCOPR,DISP=D,CLASS=S
// JOB IJBHCOPR - OPERATOR INTERFACE FOR CRYPTO
// EXEC IJBHCOPR
/*
/&
* $$ EOJ
```

---

To activate the hardware crypto support, complete the following steps:

1. Shut down TCP/IP (CSI or BSI) and your TCP/IP applications.
2. Start the job stream as shown in Example 4-13 or a similar job.
3. Restart TCP/IP and your TCP/IP applications.

Phase IJBHCOPR replaces the previous method of starting the crypto device driver by way of phase IJBCRYPT. With IJBHCOPR, you have a full operator interface, such as the SECSERV application, which is available for BSM users.

When IJBHCOPR is started, available crypto hardware is detected and shown on the operator console. To obtain a list of available crypto commands, use the **HELP** command, as shown in the following example:

```
msg nn,data=help
```

**Note:** Do not use IJBHCOPR with the SECSERV application of BSM, or the IJBCRYPT phase, which is still available for backward compatibility.

## 4.2.10 Changing the status of hardware-based encryption

TCP/IP for z/VSE transparently uses hardware-based encryption when available. With a \$SOCKOPT phase, you can control whether encryption is performed in hardware or software. Jobs that are described in this section can be used to enforce or disable hardware-based encryption.

### Suppressing the CP assist feature

Example 4-14 shows a job to suppress the use of the CP assist feature.

#### Example 4-14 Job to suppress CP assist feature

---

```
* $$ JOB JNM=SOCKOPT1,CLASS=0,DISP=D
// JOB SOCKOPT
*
* * CREATE A $SOCKOPT.PHASE THAT SUPPRESSES THE
* * USE OF THE KMC INSTRUCTION
```

```

*
// OPTION CATAL
// LIBDEF *,SEARCH=PRD1.BASE
// LIBDEF *,CATALOG=PRD1.BASE
// EXEC ASMA90,SIZE=ASMA90,PARM='SZ(MAX-200K,ABOVE)'
      PUNCH    ' PHASE $SOCKOPT,* '
$SOCKOPT CSECT
      SOCKOPT  CSECT,SSLFLG2=$OPTSNZA
      END      $SOCKOPT

/*
// EXEC LNKEDT,SIZE=512K
/*
/&
* $$ E0J

```

---

### Forcing the CP assist feature

The job that is shown in Example 4-15 forces the use of the CP assist feature.

*Example 4-15 Job to force the CP assist feature*

---

```

* $$ JOB JNM=SOCKOPT2,CLASS=0,DISP=D
// JOB SOCKOPT2
*
* * THIS JOB WILL CREATE A $SOCKOPT.PHASE THAT FORCES THE
* * USE OF THE KMC INSTRUCTION
*
// OPTION CATAL
// LIBDEF *,SEARCH=PRD1.BASE
// LIBDEF *,CATALOG=PRD1.BASE
// EXEC ASMA90,SIZE=ASMA90,PARM='SZ(MAX-200K,ABOVE)'
      PUNCH    ' PHASE $SOCKOPT,* '
$SOCKOPT CSECT
      SOCKOPT  CSECT,SSLFLG2=$OPTSFZA
      END      $SOCKOPT

/*
// EXEC LNKEDT,SIZE=512K
/*
/&
* $$ E0J

```

---

### Suppressing the use of crypto cards

The job that is shown in Example 4-16 suppresses the use of crypto cards. RSA keys with 2048 bits or greater require a crypto card. TCP/IP for z/VSE can perform RSA only up to 1024 bits in software.

*Example 4-16 Job to suppress using crypto cards*

---

```

* $$ JOB JNM=$SOCKOPT,CLASS=0,DISP=D
* $$ LST CLASS=A
// JOB $SOCKOPT
// OPTION CATAL
// LIBDEF *,SEARCH=PRD1.BASE
// LIBDEF *,CATALOG=PRD1.BASE
// EXEC ASMA90,SIZE=ASMA90,PARM='SZ(MAX-200K,ABOVE)'
      PUNCH    ' PHASE $SOCKOPT,* '

```

```

$SOCKOPT CSECT
      SOCKOPT CSECT,           Generate a csect           X
      SSLFLG2=$OPTSNHC,       SSL do not use hw-crypto   X
      END $SOCKOPT
/*
// EXEC LNKEDT,SIZE=512K
/*
/&
* $$ E0J

```

---

The crypto cards are used by default when they are available.

### 4.2.11 AP-queue Adapter Interruption Facility

The AP-queue Adapter Interruption Facility (AP interrupts) was introduced with IBM z10, and provides a second way of obtaining responses from a crypto card than polling. At the time of this writing, AP interrupts were not available when running under IBM z/VM.

AP interrupts allow the application to be posted when a reply from a crypto card is available for dequeuing, which minimizes the elapsed time between enqueueing a request and dequeuing the related reply. AP interrupts also tend to minimize processor (CPU) load.

By using the crypto device driver, you can enable or disable AP interrupts with the APEAI and APDAI commands. The default setting is to use polling with a polling time interval of 1/300 sec. Decreasing the polling time interval to zero also decreases elapsed job time, but increases CPU load.

## 4.3 Hardware-based tape encryption with z/VSE

Tape encryption is used to protect sensitive data that leaves your data center on physical tapes or is stored on tape for archive purposes. The IBM System Storage® Tape Encryption solution is provided by the TS1120 and later family of tape drives.

Encryption is done by the tape drive. Key management is done through the Encryption Key Manager (EKM), which is a Java application that acts as a key server. The advantage of IBM Tape Encryption is that data is encrypted after compression, and no other software program costs are incurred. IBM Tape Encryption saves space on tape cartridges and saves more hardware investments.

The hardware-based tape encryption solution is complemented by a software-based encryption solution that is described in 4.5, “Software-based encryption with Encryption Facility for z/VSE V1R1” on page 145.

The important terms that are used in this section are listed in Table 4-5.

*Table 4-5 Important terms of tape encryption*

Term	Meaning
Encryption Key Manager (EKM)	A Java application that maintains and serves keys in a TS1120 and later environment.
Encryption key or data key (DK)	An AES-256 key that is randomly generated by the EKM and used for data encryption.

Term	Meaning
Key encrypting key (KEK)	An RSA key that is maintained by the EKM and is used to encrypt a data key.
Externally encrypted data key (EEDK)	A data key was encrypted by a KEK.
Key encrypting key label (KEKL)	The label that is used to identify an RSA key in the EKM keystore. It can be a string or a hash value that is calculated from the binary key material.
Keystore	A Java keystore file, which is password protected.
Crypto services	Crypto-related services (encryption, decryption, creation of random numbers, and others) that are provided by the Java runtime environment that is used by the EKM. Depending on which platform the EKM runs, crypto services might be hardware-accelerated.

### 4.3.1 Encrypting data

Data encryption is performed by the tape drive by using a randomly created 256-bit AES key. When data compression is enabled, the tape drive first compresses the data before encrypting it.

The following process is used for encrypting data:

1. The tape cartridge is loaded; encryption is specified.
2. Key encrypting key labels (KEKs) are passed to the CU (if specified).
3. The tape drive requests a data key from the EKM.
4. EKM generates a data key and encrypts it with a key encrypting key (KEK).
5. The encrypted data key is transferred to the tape drive.
6. The tape drive writes encrypted data and stores the encrypted data key on the cartridge.

Figure 4-31 on page 128 shows the flow of data and keys between the keystore, crypto service, EKM, and the tape drive.

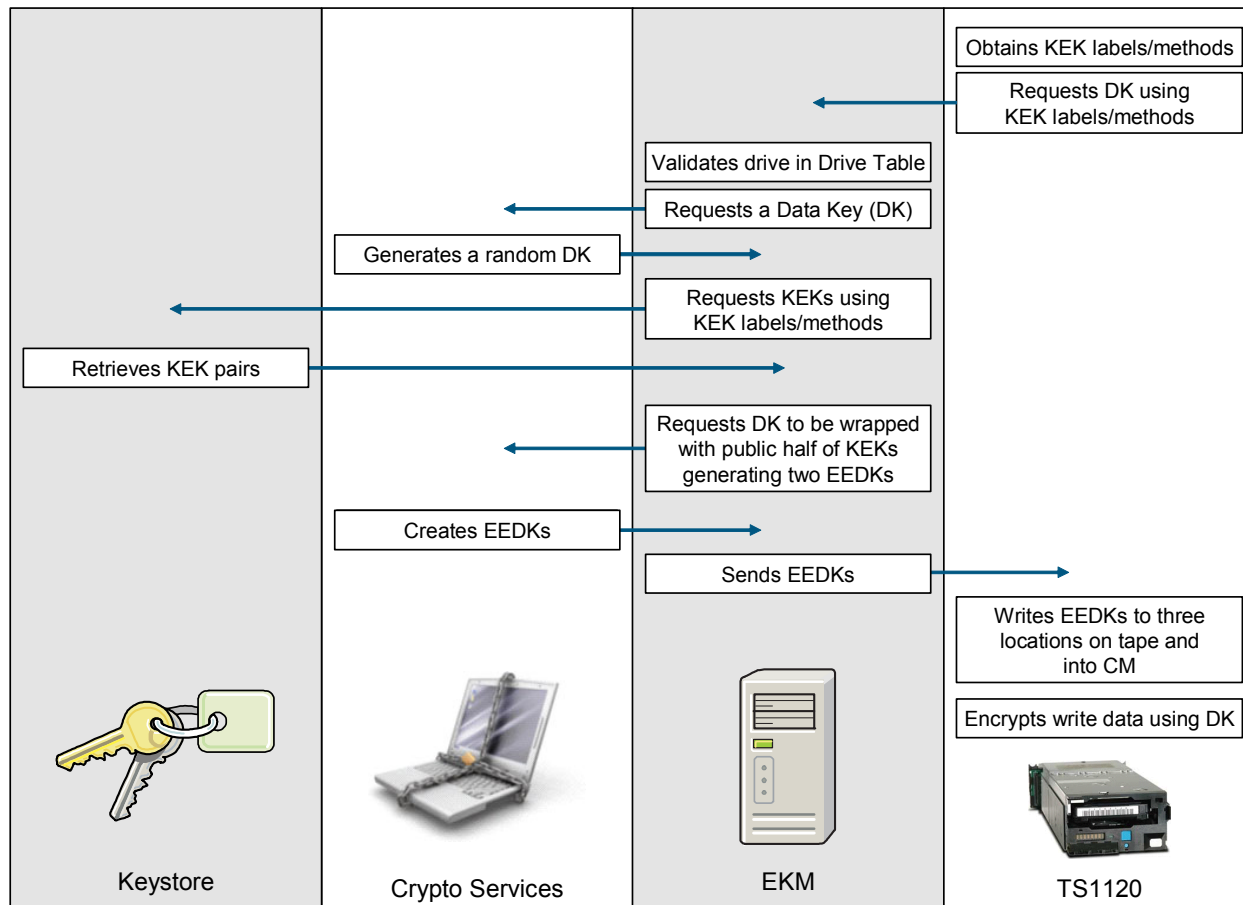


Figure 4-31 Data and encryption key flow

When data is to be encrypted, the application specifies up to two KEKs that identifies the RSA keys that are used to encrypt a randomly created data key. If the application does not specify key labels (KEKs), the EKM uses its default keys as specified in the EKM configuration file.

The EKM validates that the requesting drive is in its list of known drives. After validation, the EKM sends the encrypted data key (EEDK) to the tape drive. The drive stores the EEDKs in the cartridge memory (CM) and various locations on the magnetic tape.

The following modes can be used for creating the EEDK:

- ▶ CLEAR or LABEL mode
  - The KEK is stored in the EEDK.
- ▶ HASH mode
  - A hash value of the key material is stored in the EEDK.

When sharing tapes with clients and business partners, use the HASH mode to avoid problems with different key labels at different locations. The HASH mode enables each party to use any KEK when a certificate is imported into their keystore. The alternative is to use the CLEAR or LABEL mode and then have each party agree on a unique KEK for the same key on all locations.



## 4.3.2 Decrypting data

When decrypting data, specifying the KEKL is not necessary. The key exchange is tried automatically when an encrypted tape is detected by the hardware.

The following process is used for decrypting data:

1. An encrypted tape is mounted for a read request. The drive recognizes that the tape is encrypted.
2. The tape drive reads the EEDKs from the cartridge and requests the corresponding private key of the KEK that is used to encrypt the data key. When CLEAR or LABEL is specified at the encrypting site, the key label as stored on the cartridge must match with the appropriate KEK in the EKM keystore. When HASH is specified, the EKM finds the right KEK by its hash value.
3. The EKM decrypts the data key in the EEDK through the cryptographic services.
4. The EKM sends the data key to the drive in a secure way.
5. The drive decrypts the data on the tape.

## 4.3.3 z/VSE considerations

An overview of the hardware-based tape encryption in z/VSE is shown in Figure 4-32.

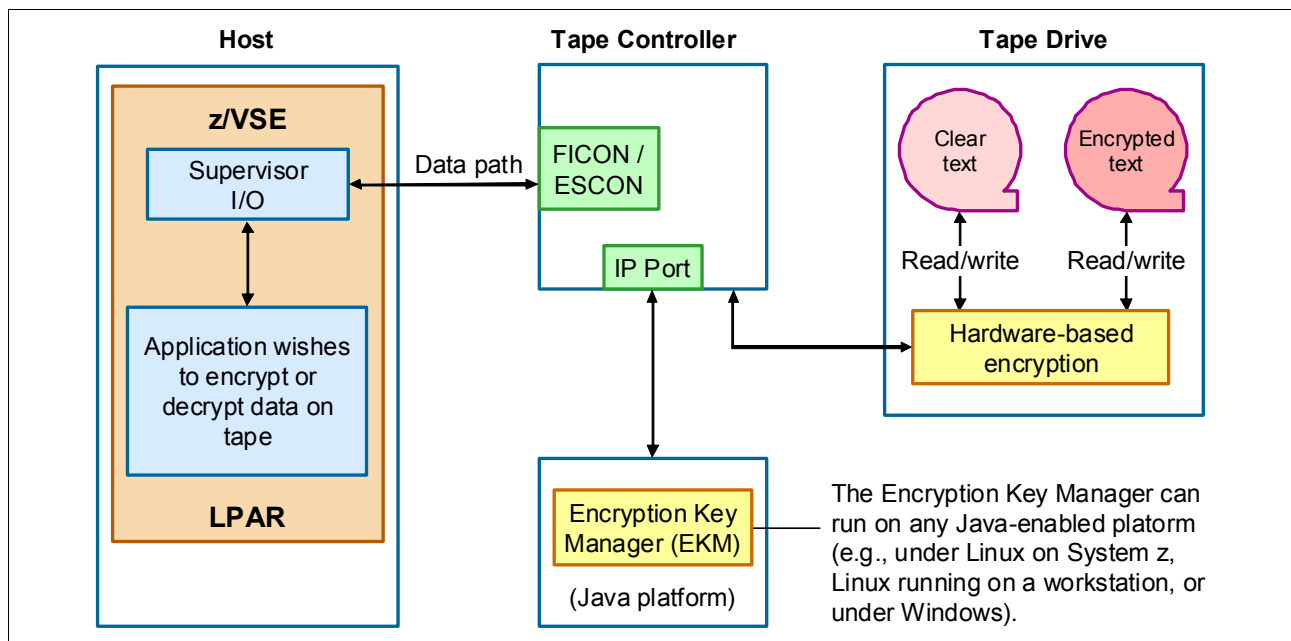


Figure 4-32 Hardware-based tape encryption in a z/VSE environment

The key encrypting key labels are passed through the z/VSE Supervisor to the tape drive's control unit to the (EKM).

*Key negotiation* occurs between the tape drive and the EKM, during which the EKM validates or supplies encryption keys with the tape drive. The tape drive and EKM communicate through the TCP/IP protocol.

If the key-verification process is successful, the data on the tape cartridge is encrypted. If it is not successful, an error message is returned.

## 4.3.4 Hardware and software requirements

To use hardware-based tape encryption with z/VSE, be aware of hardware and software requirements that are described in this section.

### Hardware requirements

IBM System Storage TS1120 Tape Drive (3592 E05), TS1130 Tape Drive (3592 E06), or later must be installed, as shown in Figure 4-33.

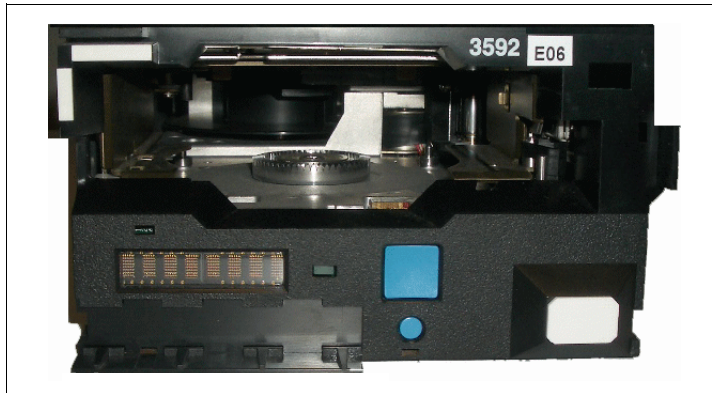


Figure 4-33 IBM System Storage TS1130 Tape Drive

The IBM TS1120, IBM TS1130, or later Tape Drive can be installed in a rack, inside an IBM TS3400 Tape Library or later, or in an installed IBM 3494 Tape Library, frame models L22, D22, or D24 or later.

You must have an encryption-capable tape drive installed and configured, such as the IBM System Storage TS1120, TS1130, or later, attached to a 3592 J70 or TS1120 C06 tape controller or later.

Encryption must be enabled and activated in the control unit and tape drive. The encryption capability is implemented through tape drive hardware, and microcode additions and changes.

All 3592 media (MEDIA 5 - MEDIA 10), including WORM cartridges, can be encrypted.

### Encryption Key Manager requirements

EKM must be implemented in at least one of the Java runtime environments before you can encrypt any tape. EKM is part of the IBM Java environment and uses the IBM Java Security components for its cryptographic capabilities. The keystore that is used by EKM is defined as part of the Java Cryptography Extension (JCE) and an element of the Java Security components, which are, in turn, a part of the Java runtime environment.

For more information about how to implement the EKM, see *IBM Encryption Key Manager component for the Java platform, Introduction, Planning, and User's Guide*, GA76-0418.

## 4.3.5 Writing and reading encrypted data in z/VSE

In this section, we describe how to write or read encrypted data in z/VSE.

### Writing encrypted data

To write encrypted data to tape, you must set the encryption mode.

## Encryption modes

The encryption mode is a 1-byte field that determines how the data on the tape should be written. Supported encryption modes include the following examples:

<b>X'03'</b>	Encryption Write Mode for the TS1120
<b>X'04'</b>	Encryption Write Mode for the TS1130 or TS1140 (3592 Model E07)
<b>X'0B'</b>	Encryption and IDRC (compression) Write Mode for the TS1120
<b>X'0C'</b>	Encryption and IDRC (compression) Write Mode for the TS1130 or TS1140 (3592 Model E07)
<b>X'23'</b>	Encryption with unbuffered Write Mode for the TS1120
<b>X'24'</b>	Encryption with unbuffered Write Mode for the TS1130
<b>X'2B'</b>	Encryption and IDRC (compression) and unbuffered Write Mode for the TS1120
<b>X'2C'</b>	Encryption and IDRC (compression) and unbuffered Write Mode for the TS1130 or TS1140 (3592 Model E07)

You can set the encryption mode as permanent when you specify the mode during IPL with the ADD statement, as shown in the following example:

```
ADD cuu,TPA,mode
```

If you want to specify a temporary or permanent encryption mode on job base, use the JCL statement ASSGN, as shown in the following example:

```
ASSGN SYSxxx,cuu,mode
```

## Key encrypting key labels

Jobs for backing-up to tape with encryption (LIBR, FCOPY, and VSAM backup) can include a JCL key encrypting key labels (KEKL) statement.

**Note:** If your job does not contain a KEKL statement, the EKM uses the defaults that you previously generated and stored in the EKM.

The JCL KEKL statement can have one of the following formats:

```
// KEKL UNIT={cuu|SYSnnn},KEKL1='KEKL1',KEM1={L|H},KEKL2='KEKL2',KEM2={L|H}[ ,REKEY]
// KEKL UNIT={cuu|SYSnnn},KEKL1='KEKL1',KEM1={L|H}[ ,REKEY]
// KEKL UNIT={cuu|SYSnnn},CLEAR
```

The following parameters are used:

► KEKL1, KEKL2

The KEKL1 parameter is the label for the first key encrypting key to be used by the EKM to encrypt the data to be stored on the tape. The parameter must be enclosed in single quotation marks. If you do not specify a KEKL1, z/VSE uses the default KEKL1 and KEKL2 that are stored by the EKM. Consider the following rules:

- You cannot specify a KEKL2 without specifying a KEKL1.
- If you specify a KEKL1 but not a KEKL2, z/VSE uses the value of KEKL1 for KEKL2.
- If you do not specify a KEKL1 and a KEKL2, z/VSE uses the default KEKL1 and KEKL2 that are stored by the EKM.

► KEM1, KEM2

This parameter specifies how the label for the first key encrypting key (KEKL1) is encoded by the EKM and stored on the tape cartridge. The following values can be used:

- L: Encoded as the specified label
- H: Encoded as a hash of the public key

► REKEY

This parameter enables a tape cartridge that was encrypted to have its data key re-encrypted by using one or two new key encrypting keys. These keys are specified by one new KEKL or multiple new KEKLs: KEKL1/KEM1 and possibly KEKL2/KEM2. This approach enables a tape cartridge to be *re-keyed* without having to copy the data to another volume. That is, the same data key is encrypted by using new key encrypting keys, with the following condition: If a REKEY request is submitted against a volume that is not positioned at Load Point (LP), z/VSE forces a rewind of the tape before the REKEY is processed.

► CLEAR

This parameter indicates that the information that was established by a KEKL statement is cleared.

**Note:** You might have to reset the KEKL (the default KEKL, or the KEKL from a previous KEKL statement) on a previously-encrypted volume. To do so, you must run a **WRITE** command (for example, writing a tape mark) from the beginning-of-tape (BOT) with encryption mode *not* active.

Example 4-17 shows how you can create an encrypted backup tape of a z/VSE library.

*Example 4-17 Batch job to create an encrypted tape*

---

```
// JOB ENCRYPT
// ID USER=user-ID,PWD=password
// ASSGN SYS005,480,03
// KEKL UNIT=480,KEKL1='HUSKEKL1',KEM1=L,KEKL2='HUSKEKL2',KEM2=L
// EXEC LIBR
BACKUP LIB=PRD2 TAPE=SYS005
/*
/ &
```

---

### Reading the contents of an encrypted tape

To read the encrypted data, the job uses the keys that are stored on the tape for the key negotiation. No KEKL statements are required. If KEKL statements are included in the job, they are ignored.

However, the following prerequisites must be met to read an encrypted tape:

- The tape was encrypted by using keys that are known by the currently-connected EKM.
- The encryption keys were not deleted from the currently-connected EKM.

## 4.3.6 Recognizing an encrypted tape

You can easily determine whether a tape is encrypted. Use the console command **QT** to query the tape information, as shown in Example 4-18.

*Example 4-18 Display the details of an encrypted tape*

---

```
QT A83
AR 0015 CUU CODE DEV.-TYP VOLID USAGE MED-TYP STATUS POSITION
AR 0015 A83 5603 3592-E05 PAUL01 BG CST5 /E RESERVED 8 BLK
AR 0015 CU 3592-C06 LIB 3494-L10 (GALL88)
AR 0015 FAST-ACC.SEG.= 0 MB FILES = 2
AR 0015 KEKL1:KEY_LABEL_001
```

---

AR 0015 KEKL2:KEY\_LABEL\_002  
AR 0015 1I40I READY

---

In this example you see:

- ▶ CODE 5603 indicates:
  - A tape drive that uses the TPA is attached to z/VSE (the **56** part of 5603).
  - This tape drive is assigned to encryption mode (the **03** part of 5603).
- ▶ 3592-E05 is the device type for the TS1120 (IBM 3592 E05) tape drive.
- ▶ CST5 /E is the media type for 300 GB 3592 cartridges. The /E indicates *encrypted*. It is displayed only after at least one successful write operation from z/VSE by using encryption on this device.
- ▶ KEY\_LABEL\_001 is the label for the first key encrypting key to be used by the EKM to encrypt the data encryption key.
- ▶ KEY\_LABEL\_002 is the label for the second key encrypting key to be used by the EKM to encrypt the data encryption key.

### 4.3.7 More information about using hardware-based tape encryption

In this section, we provide more information about the use of z/VM and handling of encrypted tapes.

#### Running z/VSE as a guest under z/VM

The z/VSE and z/VM operating systems support hardware-based tape encryption.

Use hardware-based tape encryption on *either* z/VSE or z/VM, *not on both* operating systems. Otherwise, errors can occur because different key encrypting key labels are used.

#### Positioning of an encrypted tape

If the tape that is specified in the CUU device address is at load point, the new mode setting is immediately effective.

If the tape that is specified in the CUU device address is not at load point, the new mode setting is effective the next time a write occurs at load point.

Consider the following points for the encryption mode setting (for example, X'03'):

- ▶ If the tape was at load point, the tape is written as encrypted.
- ▶ If the tape was not at load point, the tape continues writing in the current mode.
- ▶ If the first file written to a tape is encrypted, all subsequent files that are written to that same tape cartridge are encrypted by using the same data key.

#### Overwriting an encrypted tape

If an encrypted volume is processed, but the key is unknown to the EKM, access might fail with the following message:

```
0P68I Key Exchange Error
```

To avoid this problem, you can write a beginning-of-tape (BOT) mark.

## Multivolume file processing

To process a multivolume file on an alternate volume, you must specify the same KEKL as was specified for the original volume.

## 4.4 Example of TS1120 installation

This section describes the setup of a TS1120 tape drive with encryption support and the setup and configuration of an EKM on an Intel Linux platform.

The following software is used in the example setup:

- ▶ z/VSE V4R2M0
- ▶ TCP/IP for VSE/ESA 1.5F
- ▶ Java 1.5.0 for Linux from IBM (java-1.5.0-ibm-1.5.0.3.3-3jpp)
- ▶ EKM version 2.1-20080730
- ▶ IBM Key Management tool as part of IBM Java

### 4.4.1 Installing the prerequisite programs

The programs that are listed in this section are required for the setup.

#### IBM Java

You can download various IBM JDKs and JREs from this web page:

<https://developer.ibm.com/javasdk/downloads/>

Part of the IBM Java installation is the IBM Key Management tool, which is used later to set up the necessary keys and certificates.

#### Encryption Key Manager

EKM is included in the Tivoli Key Lifecycle Manager product. For more information about how to obtain the product, see the product's related documentation.

Start the EKM by using the following shell command:

```
java -cp IBMKeyManagementServer.jar com.ibm.keymanager.EKMLaunch  
KeyManagerConfig.properties &
```

#### IBM key management

The IBM key management tool iKeyman is part of the IBM Java installation and is included in the EKM java .jar file.

In Windows, the key management GUI is started with the `ikeyman.exe` in folder `jre/bin` in the Java installation directory.

On Linux, the tool is started with:

```
java -cp IBMKeyManagementServer.jar com.ibm.gsk.ikeyman.Ikeyman &
```

## Java policy files

Regardless which version of an IBM SDK you use, you must replace the `US_export_policy.jar` and `local_policy.jar` files in your `java_home/usr/java5/jre/lib/security` directory with new files. These files can be downloaded from the following web page:

<https://ibm.biz/BdZ5J4>

These new files install the unrestricted policy files that EKM requires to serve AES keys.

For more information about installing these prerequisites programs on platforms other than linux, see *IBM Encryption Key Manager component for the Java platform, Introduction, Planning, and User's Guide*, GA76-0418.

## 4.4.2 Setting up the TS1120

The TS1120 setup consists of the TS1120 hardware setup and the C06 controller configuration.

### Tape drive controller setup

The TS1120 tape drive controller is an IBM AIX® box that must be configured to recognize the encryption option of the tape drive. Therefore, you can use the SCSI/FCP Configurator. A new IML of the controller is necessary after enabling the encryption option.

### Checking for encryption capability

From the operating system perspective, whether the TS1120 encryption capability is active can be determined by using several methods.

#### *Using the Q TAPES command on z/VM*

Under z/VM, run the following `q tapes` command to display the drive properties:

```
q tapes details 811
TAPE 0811 SEQUENCE NUMBER A2631 LIBPORT 1 ENCRYPTION CAPABLE
```

The output must display ENCRYPTION CAPABLE.

#### *Using the CP command Q V*

In CP or on the z/VSE console, run the CP command `q v` to display drive properties:

```
* cp q v 811
AR 0015 TAPE 0811 ON DEV 0811 3590 R/W SUBCHANNEL = 00C6 ENCRYPTION CAPABLE
```

Again, the output must display ENCRYPTION CAPABLE.

#### *Using the QT command on z/VSE*

For more information about the QT command, see 4.3.6, “Recognizing an encrypted tape” on page 132.

## 4.4.3 Setting up the EKM

At this point, we assume that the IP address of the EKM platform was specified in the TS1120 hardware setup.

Because two sets of RSA keys are necessary for the EKM operation, you should set up the following separate keystores:

- ▶ Keys to establish the SSL connection between EKM and the tape drive controller
- ▶ Keys used to encrypt data keys on a tape cartridge (key encrypting keys, KEKs)

Although all keys can be in a single keystore, this approach can be problematic when an SSL key runs out of its validity period. In this case, the SSL connection between EKM and tape drive is no longer established. The *IBM Encryption Key Manager component for the Java platform, Introduction, Planning, and User's Guide*, GA76-0418 includes the following guideline:

“As certificates in the keystore age, Encryption Key Manager SSL operations can risk failure due to expired certificates. The risk of failed SSL operations can be reduced by configuring the Encryption Key Manager to use two separate keystores, one for storing the keys and certificates for the TS1120 or TS1130 drives, and the other to hold keys and certificates for SSL operations. When the certificates in the SSL keystore expire, they can be deleted and recreated without affecting the Encryption Key Manager's ability to serve keys to TS1120 or TS1130 Tape Drives.”

The following sections describe how to set up an EKM keystore.

## Defining an EKM user under Linux

For EKM installations under Linux, add a user `ekm` to the system that is used to run and configure the EKM. A typical EKM installation features the structure that is shown in Example 4-19.

*Example 4-19 Structure of EKM installation*

---

```
/home/ekm/EKM

/home/ekm/EKM/ClientKeyManagerConfig.properties
/home/ekm/EKM/KeyManagerConfig.properties
/home/ekm/EKM/IBMKeyManagementServer.jar

/home/ekm/EKM/ekmstop.sh
/home/ekm/EKM/ekmstatus.sh
/home/ekm/EKM/ekmstart.sh

/home/ekm/EKM/log
/home/ekm/EKM/log/debug.log
/home/ekm/EKM/log/native_stdout.log
/home/ekm/EKM/log/native_stderr.log
/home/ekm/EKM/log/debug_cli.log

/home/ekm/EKM/metadata
/home/ekm/EKM/metadata/EKMData.xml
/home/ekm/EKM/audit
/home/ekm/EKM/audit/kms_audit.log
/home/ekm/EKM/KeyGroups.xml
/home/ekm/EKM/filedrive.table
/home/ekm/EKM/keys
/home/ekm/EKM/keys/SSLKeys.jck
/home/ekm/EKM/keys/EKMKeys.jck
```

---



## Setting up the keystore

Complete the following steps to set up the keystore:

1. Start the IBM key management tool:  

```
java -cp IBMKeyManagementServer.jar com.ibm.gsk.ikeyman.Ikeyman
```
2. Select **Create a new key database file**, as shown in Figure 4-34.

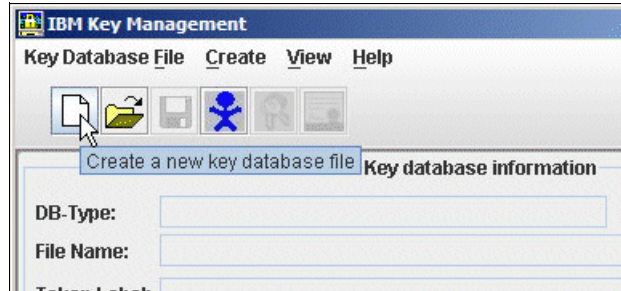


Figure 4-34 Creating a new key database file

3. In the next window, select type **JCEKS**, and click **OK**, as shown in Figure 4-35.

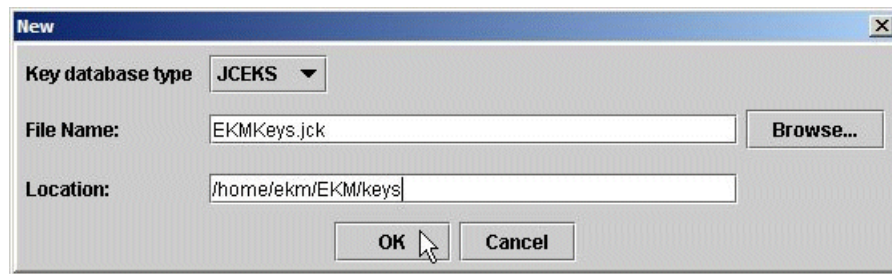


Figure 4-35 New key database definition

4. In the next window, enter the password for the new key database file and click **OK**.
5. In the Key Management main window, select **Personal Certificates**.
6. Create a self-signed certificate by entering Enter personal data for the new certificate, (see Figure 4-36 on page 138).

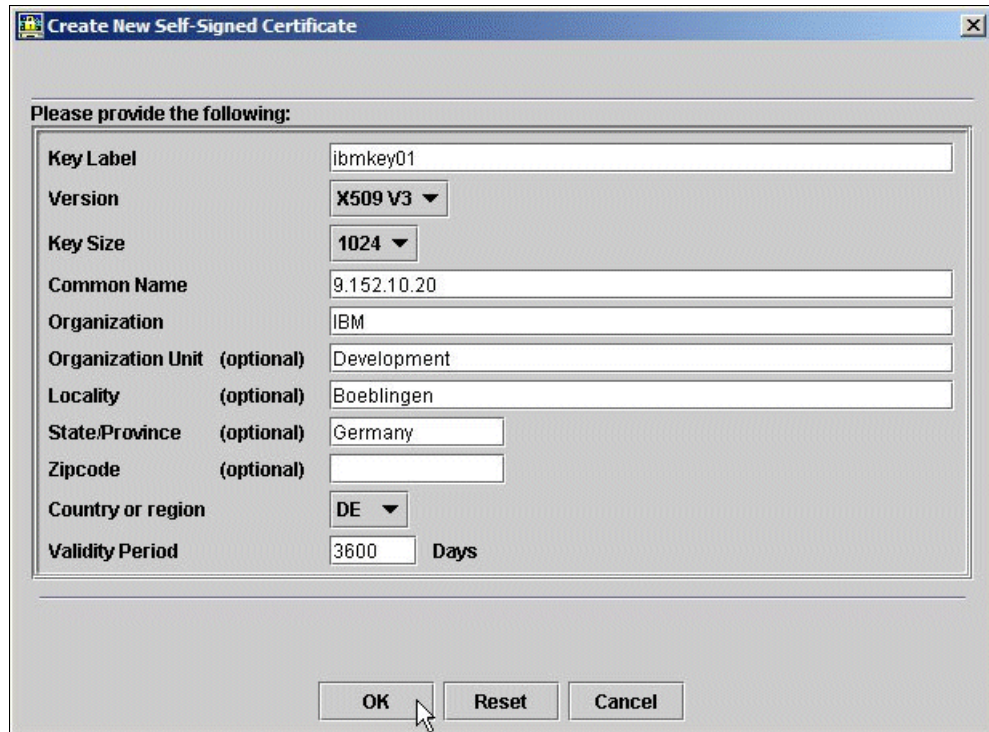


Figure 4-36 Specify certificate information

A keystore can also be created by using the **keytool** executable. The following command string creates a certificate with a validity period of 3600 days (10 years):

```
keytool -keystore SSLKeys.jck -storetype jceks -genkey -alias sslcert -keyalg RSA -keysize 1024 -validity 3600
```

You must ensure that certificate validity periods are carefully tracked and maintained in your company. When creating a certificate, decide how long the certificate will be valid. Short validity periods increase flexibility and security; long validity periods decrease the extra amount of work that is needed for renewing certificates. In our example, a long validity period of 10 years is specified, which might be not appropriate for your company.

**Note:** Key pairs (that is, self-signed certificates) must not be exported from a keystore to transfer a key to another keystore. In this case, the private key is lost. Copy the entire keystore file to another EKM platform to have the same keys on both platforms. If you want to include the keys from a specific keystore into another keystore, open the target keystore, select **Personal Certificates**, and click **Import** to import the entire source keystore.

### Considerations for key validity periods

The SSL/TLS connection between EKM and the tape drive controller cannot be established by using expired SSL/TLS keys. Therefore, have the SSL/TLS keys in a separate keystore.

This consideration does not apply to KEKs that are used to encrypt the data key on a cartridge. You can create an encrypted tape with an expired KEK and decrypt an encrypted tape with an expired KEK.

## Setting up a second EKM

For a production environment, use two EKM installations for failover reasons. If one EKM cannot be reached by the tape drive, the controller tries the second EKM transparently. Both EKMs must be configured identically, except for the individual IP address.

The IP addresses of both EKMs must be specified in the TS1120 hardware configuration.

This setup requires the same data and configuration files on both servers. This setup can be achieved with the EKM sync-function. However, the sync-function only synchronizes the drive table and the configuration files. Keyring, certificates, and the key groups XML file must be copied manually. The synchronization settings are defined in the file `KeymanagerConfig.properties` of the primary EKM server.

**Note:** All manual changes always must be performed on EKM. Automatic synchronization with EKM2 keeps the two EKMs synchronized.

The active EKM (usually EKM1) is selected with the KVM switch at the TS3500 control unit.

## Backup of EKM configuration

The following data should be saved on a separate data store (CD-ROM, network drive, and others) to set up a recovery EKM server in an emergency situation:

- ▶ EKM configuration files
- ▶ Tape drive table
- ▶ Key groups (XML file)
- ▶ Keystore (all keys, all certificates)

With the help of this alternative EKM server, a possibility is to read encrypted backup tapes if the primary EKMs do not work.

## Modifying the EKM configuration files

The following EKM configuration files are available:

- ▶ The `KeymanagerConfig.properties` file contains EKM server definitions
- ▶ The `ClientKeyManagerConfig.properties` file contains EKM client (command-line interface) definitions

### *KeymanagerConfig.properties*

The EKM configuration file `KeymanagerConfig.properties` is in the EKMS folder. The parameters that can be modified are listed in Table 4-6 on page 140.

Table 4-6 EKM configuration parameter

Parameter	Description
Sync.timeinhours = 24	Synchronization of data is performed in intervals of 24 hours.
Sync.action = merge   rewrite	Defines how data gets synchronized. Both values (merge or rewrite) cause the configuration file to be overwritten.
Sync.ipaddress = ipaddr	Defines IP address of alternate EKM server (EKM2).
Sync.type = all   drivetab   config	Defines which data to synchronize. The sync events are written to the log in folder Audit. Here, you can check whether the synchronization was successful.
Drive.default.alias1=mykey01	Defines the first default key, which is used when no KEKL statement is in the z/VSE JCL.
Drive.default.alias2=mykey02	Defines the second default key, which is used when no KEKL statement is in the z/VSE JCL.
Admin.ssl.keystore.name= /home/ekms/EKMSKeys/MyKeyring01.jck	Defines the keystore name.
TransportListener.ssl.truststore.name= /home/ekms/EKMSKeys/MyKeyring01.jck	Defines the transport listener truststore name.
TransportListener.ssl.keystore.name= /home/ekms/EKMSKeys/MyKeyring01.jck	Defines the transport listener keystore name.

On Linux, the EKM is started by using the following shell command:

```
java -cp IBMKeyManagementServer.jar com.ibm.keymanager.EKMLaunch
KeyManagerConfig.properties &
```

Example 4-20 shows the EKM configuration file with two keystores.

Example 4-20 EKM configuration file with two keystores

```
TransportListener.ssl.port = 1100
config.keystore.password.obfuscated = 6C09DCCDDFDFA0EDDDDBE0
TransportListener.tcp.port = 1200
TransportListener.tcp.timeout = 0
TransportListener.ssl.keystore.password.obfuscated = 5B09CBCCCECE8FCBCACACF
Admin.ssl.keystore.name = keys/SSLKeys.jck
TransportListener.ssl.clientauthentication = 0
TransportListener.ssl.ciphersuites = JSSE_ALL
Audit.handler.file.size = 1000
drive.acceptUnknownDrives = true
Audit.metadata.file.name = metadata/EKMData.xml
TransportListener.ssl.truststore.name = keys/SSLKeys.jck
Audit.handler.file.directory = audit
TransportListener.ssl.protocols = SSL_TLS
Admin.ssl.keystore.password.obfuscated = 3709A798AAAA6BA9B6A6AB
config.keystore.file = keys/EKMKeys.jck
debug.output = simple_file
TransportListener.ssl.keystore.name = keys/SSLKeys.jck
Audit.eventQueue.max = 0
debug.output.file = log/debug.log
```

```

Audit.handler.file.name = kms_audit.log
drive.default.alias1 = ibmkey01
drive.default.alias2 = ibmkey02
Audit.event.outcome = success,failure
Audit.event.types = all
config.drivetable.file.url = FILE:filedrive.table
Admin.ssl.truststore.name = keys/SSLKeys.jck
Sync.timeinhours = 24
Sync.action = rewrite
Sync.ipaddress = 9.152.31.2
Sync.type = all

```

---

### ***ClientKeyManagerConfig.properties***

The ClientKeyManagerConfig.properties file is in the EKMS folder. The parameters that are listed in Table 4-7 can be modified.

*Table 4-7 EKM configuration parameters*

<b>Parameter</b>	<b>Description</b>
TransportListener.ssl.truststore.name=/home/ekms/EKMSKeys/Keyring01.jks	Defines the transport listener truststore name.
TransportListener.ssl.keystore.type=jceks	The keystore type must be set to JCEKS.
TransportListener.ssl.truststore.type=jceks	The truststore type must be set to JCEKS.

You start the command-line interface (CLI) as shown in the following example:

```

java -cp IBMKeyManagementServer.jar com.ibm.keymanager.KMSAdminCmd
ClientKeyManagerConfig.properties -i

```

Example 4-21 shows a CLI configuration file.

*Example 4-21 CLI configuration file*

---

```

TransportListener.ssl.truststore.name=keys/SSLKeys.jck
debug.output.file=log/debug_cli.log
TransportListener.ssl.ciphersuites=JSSE_ALL
TransportListener.ssl.host=localhost
TransportListener.ssl.keystore.type=jceks
TransportListener.ssl.keystore.password.obfuscated=8F09FFF00202C301FBFE03
TransportListener.ssl.truststore.type=jceks
debug.output=simple_file
TransportListener.ssl.port=1100
TransportListener.ssl.keystore.name=keys/SSLKeys.jck
TransportListener.ssl.protocols=SSL_TLS
TransportListener.tcp.timeout=0

```

---

## Useful EKM shell scripts

This section lists several useful shell scripts for starting, stopping, and showing the status of the EKM under Linux.

### ***ekmstart.sh***

The shell script that is shown in Example 4-22 starts the EKM.

*Example 4-22 ekmstart.sh*

---

```
#!/bin/sh
#
# Start Enterprise Key Manager (EKM)
#
export CLASSPATH=IBMKeyManagementServer.jar
pid=$(ps -ef | grep -i com.ibm.keymanager.EKMServer | grep -v grep | \
    head -1 | awk '{print $2}')
if [ "$pid" != "" ]; then
    echo "EKM already running with pid $pid"
else
    echo $pid
    java -cp IBMKeyManagementServer.jar \
        com.ibm.keymanager.EKMLaunch KeyManagerConfig.properties
fi
```

---

### ***ekmstop.sh***

The shell script that is shown in Example 4-23 stops the EKM.

*Example 4-23 ekmstop.sh*

---

```
#!/bin/sh
#
# Stop Enterprise Key Manager (EKM)
#
pid=$(ps -ef | grep -i com.ibm.keymanager.EKMServer | grep -v grep | head -1 | awk
    '{print $2}')
if [ "$pid" != "" ]; then
    echo -n "Stopping EKM with pid $pid: "
    kill $pid
    while ps -ef | grep -i com.ibm.keymanager.EKMServer | \
        grep -v grep > /dev/null 2>&1; do
        sleep 1
    done
    if [ $? == 0 ]; then
        echo "done"
    else
        echo "failed"
    fi
else
    echo "EKM not running"
fi
```

---

### **ekmstatus.sh**

The shell script that is shown in Example 4-24 displays the current status of the EKM.

*Example 4-24 ekmstatus.sh*

---

```
#!/bin/sh
#
# Show EKM status
#
pid=$(ps -ef | grep -i com.ibm.keymanager.EKMServer | grep -v grep | head -1 | awk '{print $2}')
if [ "$pid" != "" ]; then
    echo "EKM is running with pid $pid".
else
    echo "EKM is not running."
fi
```

---

### **Automatically activate EKM at Linux boot time**

On Red Hat Linux-based distributions, you can edit the `/etc/rc.local` file and add the following statement:

```
su - ekm -c 'cd /home/ekm/EKM && /home/ekm/EKM/ekmstart.sh'
```

On other Linux distributions, this statement might work differently.

### **Considerations for the drive table**

All encryption-capable tape drives that are used by the EKM must be made known to the EKM by using the **addrive** CLI command or with the key manager configuration file.

The **addrive** command features the following syntax:

```
addrive -drivename name -rec1 certname1 -rec2 certname2
```

Consider the following example:

```
addrive -drivename 0000059732346 -rec1 ibmkey01 -rec2 ibmkey02
```

Instead of making all drives explicitly known to the EKM by using the **addrive** command, you can force the EKM to accept all encryption-capable tape drives in the network by using the parameter that is listed in Table 4-8.

*Table 4-8 EKM configuration parameter*

Parameter	Description
Drive.acceptUnknownDrives=true	EKM accepts all encryption-capable tape drives in the network.

### **Considerations for caching keys**

The TS1120 control unit caches used KEKs so that an encrypted tape can be read, even when all EKMs are not yet reachable. After an IML of the control unit, the cache is cleared and the EKM must be available for further encryptions or decryptions.

## **4.4.4 z/VSE considerations**

On z/VSE 6.2, you can define a 3592 tape drive with permanent encryption setting in the z/VSE hardware configuration, by using fast path 241.

You can also use a statement similar to the following statement in the VSE IPLPROC to automatically perform encryption, compression, or both, for 3592 tape drives based on their device addresses:

```
ADD A80:A89,TPA,0B
```

Where

0B Encryption enabled, compression is ON

08 Encryption disabled, compression is ON

In addition, encryption can always be specified by using the // ASSGN and // KEKL job control statements.

For more information about all possible tape drive modes, see *z/VSE Administration*, SC34-2692.

## 4.4.5 Observations

This section describes any observations we made during our tests.

### Drive not capable for encryption

The symptom and reason are:

► Symptom:

The drive cannot be successfully encrypted, although the TS1120 web interface indicates the following message:

```
drive enabled for encryption
```

► Possible reason:

Encryption was specified on the TS1120 hardware panels, but the drive controller was not IMLed after the configuration change. Run the SCSI/FCP Configurator on the drive controller and re-IML the controller.

### Message 0P68I KEYXCHG ER

The symptom and reason are:

► Symptom:

Message 0P68I KEYXCHG ER is displayed on the operator console.

► Possible reason:

Most likely, the tape drive has no network connection to the EKM. For more information, see *z/VSE Administration*, SC34-2692.

For more information about this issue and other EKM error messages, see *IBM Encryption Key Manager component for the Java platform, Introduction, Planning, and User's Guide*, GA76-0418.

If you use z/VM, consider which key is used. You can specify by using the VM command **attach** whether to use the default keys or to use a key specified by an alias.

In the following example, the 680/681 drives are attached to virtual machine TESTSYS with default key support:

```
attach 680-681 testsys key
```

For more information about the **attach** command and the **key** parameter, see *CP Commands and Utilities Reference*, SC24-6081.



## 4.5 Software-based encryption with Encryption Facility for z/VSE V1R1

This section describes the use of Encryption Facility for z/VSE V1R1 and how to exchange encrypted data with Encryption Facility for z/OS V1R1 and V1R2, and workstation-based tools.

This software-based encryption solution is complemented by a hardware-based encryption solution that is described in 4.3, “Hardware-based tape encryption with z/VSE” on page 126.

The following software is used in the example setup:

- ▶ z/VSE V6R2
- ▶ TCP/IP for z/VSE V2.2
- ▶ VSE Connector Server as part of z/VSE V6R2 (job STARTVCS)
- ▶ Java 8 from Oracle
- ▶ Keytool.exe as part of Java 8
- ▶ Keyman/VSE (build date July 2017)
- ▶ Encryption Facility for z/VSE V1R1
- ▶ IBM Java Client as part of IBM Encryption Facility for z/OS Client V1R2

Encryption Facility for z/VSE, program number 5686-CF8-40, is packaged as an optional, priced feature and provides data protection by offering the encryption of data for exchange, archiving, and backup purposes. Depending on the kind of processor and the type of cryptographic hardware that is installed, the Encryption Facility for z/VSE uses hardware accelerated crypto support for encryption and decryption.

Supported file formats include single SAM files, VSAM files, or z/VSE library members, but also complete backups made with any backup tool from IBM or vendors. For single VSAM files or z/VSE library members, the filenames are specified directly in the JCL that is starting the tool.

For full backups, you first back up your data (by using any available backup tool) to a real tape or virtual tape. Then, this backup tape is encrypted to an encrypted data set, which can be written to a second real tape, virtual tape, or DASD.

Encryption Facility for z/VSE makes use of triple-DES (TDES) and AES-128 algorithms for data encryption. On a system with TCP/IP for z/VSE, you can use Encryption Facility for z/VSE to generate TDES and AES keys and encrypt them for protection through RSA public keys. Password-based key generation is also an option.

On systems without TCP/IP for z/VSE, you can use only passwords to generate clear TDES and AES keys.

For more information about Encryption Facility for z/VSE V1.1, see *z/VSE Administration*, SC34-2692.

### Prerequisites

To use Encryption Facility for z/VSE V1.1, the following prerequisites must be met:

- ▶ The activated CPU Assist Facility (CPACF) is available.
- ▶ TCP/IP for z/VSE for public key encryption is used.
- ▶ Partition size of at least 8 MB, because the tool is an IBM Language Environment VSE application.

## Relationship to Encryption Facility for z/OS

Encryption Facility for z/VSE V1.1 is closely related to Encryption Facility for z/OS V1.1. It uses the same encrypted data format (System z format) as the z/OS based tool and can exchange encrypted data with any z/OS system that has Encryption Facility for z/OS installed.

Encryption Facility for z/OS consists of several parts, including two web downloadable tools, the Java Client, and the Decryption Client for z/OS. Both tools can be downloaded at no charge.

The Java Client is intended for exchanging encrypted data with platforms that are not System z platforms. The Decryption Client is intended for decrypting data on z/OS systems where the full Encryption Facility product is not installed.

These downloadable tools can also be used in a z/VSE environment, which allows for data exchange with non z/VSE platforms.

Figure 4-37 shows components of Encryption Facility for z/OS and their relationship to Encryption facility for z/VSE.

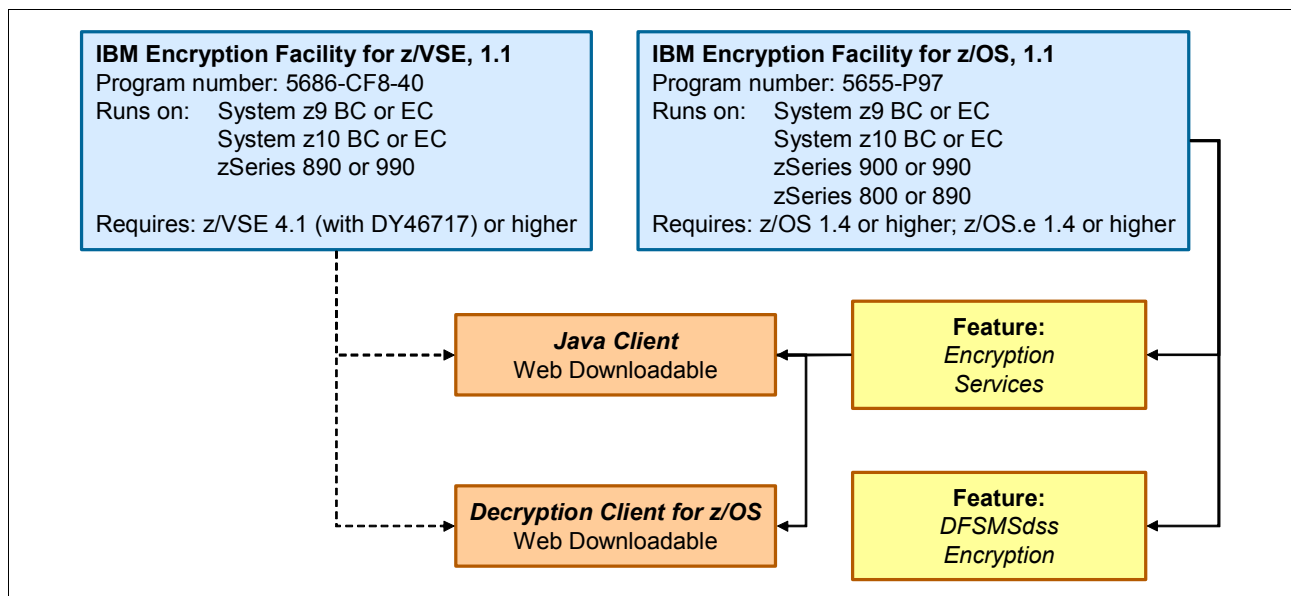


Figure 4-37 Relation between Encryption Facilities for z/VSE and for z/OS

z/VSE clients can use the z/OS Java Client for exchanging encrypted data with platforms that are not System z platforms. The Decryption Client for z/OS can be used for decrypting data that is from z/VSE.

### Obtaining the z/OS Java Client

As described in *z/VSE Administration*, SC34-2692, you can download z/OS Java Client from this web page:

<http://www.ibm.com/systems/z/os/zos/downloads/#efclient>

Extract the downloaded .zip file in a new empty directory. Further installation is not necessary.

The next section describes how Encryption Facility is related to the TS1120 tape drive and later with encryption support.

## Relationship to the TS1120 and later tape drives

The IBM TS1120 and later tape drives with encryption capability are supported by z/VSE V3R1 and later. The support for IBM TS1130 starts with z/VSE V4R2; the IBM TS1140 is supported with z/VSE V5R2.

Encryption is done by the tape drive. Key management is done with the Encryption Key Manager (EKM), which is a Java application that acts as a key server. The tape drive uses public key encryption and the related public keys are maintained by the EKM. For more information about support for the TS1120 and later, see 4.3, “Hardware-based tape encryption with z/VSE” on page 126.

Encryption Facility for z/VSE complements the support for the encrypting tape drives. Although the tape drives are the preferred solution for high-volume backup and archive applications, Encryption Facility for z/VSE is designed to allow secure exchange of encrypted data with other locations within your company, business partners, suppliers, and clients.

Password-based encryption is a simple but secure way for exchanging encrypted data with other locations, but also for creating local backup archives. The ability to write encrypted data to disk for further file transfers also complements the solution with IBM tape drives.

Various encryption scenarios and the preferred use of a tape drive or Encryption Facility are listed in Table 4-9.

*Table 4-9 Positioning of TS1140 to Encryption Facility*

Encryption scenario	Tape drive	Encryption Facility
High volume backup or archiving	Yes	-
Data encryption for rest on z/VSE disks	-	Yes
Data encryption for subsequent file transfer (like FTP)	-	Yes
Data exchange with remote sites having tape drives	Yes	Yes
Local archiving	Yes	-
Use existing tape drive environment	Yes	-
Data exchange with Encryption Facility for z/OS V1.1	-	Yes
Data exchange with non-z platforms (EF Java client)	-	Yes
Password-based encryption	-	Yes
Public key based encryption	Yes	Yes
Offload processor cycles	Yes	-

For more information about performance considerations regarding algorithms, compression, and hardware support, see 4.5.1, “Performance considerations” on page 148.

## 4.5.1 Performance considerations

Overall performance of an encryption or decryption process depends on the following parameters:

- ▶ Compression

Compressing data before encryption usually speeds up the process because less data is encrypted. Compression is performed by using the hardware compression feature that is provided by System z.

- ▶ Encryption algorithm

In terms of speed, significant differences exist between the supported encryption algorithms. AES often performs much faster than TDES. When public key encryption is used, no major differences exist between public key sizes because only the data key is encrypted by using a public key.

- ▶ Physical I/O

When encrypting a KSDS file with many small records, the encrypted data set often has a much bigger record length that uses the maximum possible record length of the underlying ESDS file. Therefore, writing encrypted data to disk requires much less I/O than writing the decrypted data set with its many small records at a later time during the decryption process.

For more information about encrypting data on z/VSE, see 4.5.2, “Password-based encryption”.

## 4.5.2 Password-based encryption

This section describes how to encrypt data on z/VSE by using password-based encryption, transfer the encrypted data set to a workstation, and decrypt the data by using the z/OS Java client.

Password-based encryption (PBE) does not require any keystores. The encryption key is directly derived from the password. Encryption Facility for z/VSE always converts the EBCDIC password that is specified by JCL to ASCII. Because passwords are case-sensitive, ensure that you specify your password correctly both in JCL and on any other related platform.

By default, Encryption Facility uses the following code pages:

- ▶ ASCII code page: IBM-850
- ▶ EBCDIC code page: IBM-1047

You can change the code page with the ASCII\_CODEPAGE and EBCDIC\_CODEPAGE parameters. Specify the code page parameters *before* the PASSWORD parameter so that your code page is active when translating the password.

To manage passwords, you can use any available tool from vendors, freeware, or shareware. For example, the open source tool KeePass<sup>4</sup> is a solution for maintaining passwords.

The use of password-based encryption does not require any special setup; we can directly start encrypting and decrypting files.

---

<sup>4</sup> For more information, see <https://keepass.info/>

## Encrypting on z/VSE

The JCL that is shown in Example 4-25 encrypts a z/VSE library member by using password-based encryption. The encrypted data set is a VSAM ESDS cluster.

*Example 4-25 Job to encrypt z/VSE library member*

---

```
* $$ JOB JNM=ENCRYPT,CLASS=0,DISP=D
// JOB ENCRYPT VSE LIBRARY MEMBER
// LIBDEF *,SEARCH=(PRD2.SCEEBASE,PRD2.PROD,PRD2.DBASE)
// EXEC IJBEFVSE
ENCRYPT
DESC='ENCRYPTION TEST'
CLRTDES
PASSWORD=MYPASSWD
COMPRESSION=NO
ICOUNT=234
CLRFILE=DD:PRD2.CONFIG(IPINIT00.L)
ENCFILE=DD:ENCDATA
/*
/&
* $$ E0J
```

---

**Note:** Do not use compression because decompressing data on a workstation is *not* possible.

Now, download the encrypted data set to your Windows PC, as shown in Example 4-26.

*Example 4-26 Download encrypted data set*

---

```
ftp> get encdata ipinit00.enc
200 Command okay
150-About to open data connection
  File:VSE.EF.ENCDATA
  Type:Binary Recfm:FB Lrecl:  80 Blksize:  80
  CC=ON UNIX=OFF RECLF=OFF TRCC=OFF CRLF=ON NAT=NO CONT=OFF
  MODE=Stream STRU=File
150 File status okay; about to open data connection
226-Bytes sent: 14,496
  Records sent: 1
  Transfer Seconds: .20 ( 71K per second)
  File I/O Seconds: .00 ( 14496-bytes)
226 Closing data connection
ftp: 14496 bytes received in 1.20Seconds 12.05Kbytes/sec.
```

---

The next step is to decrypt the downloaded file by using the z/OS Java Client. Open a command prompt and browse to the directory where the Java Client is installed. The following command string must be entered on a *single* line. Multiple lines are used here for clarity:

```
java -Djava.encryption.facility.debuglevel=1 com.ibm.encryptionfacility.EncryptionFacility
-mode decrypt
-password MYPASSWD
-outputFile ipinit00.l
-inputFile ipinit00.enc
```

**Note:** On z/VSE, the password is always specified in uppercase characters. Therefore, you must use uppercase characters when decrypting the file on a workstation.

Because the decrypted file now contains EBCDIC characters, you must upload the file (for example, to z/VM) to view it correctly. This step is not necessary when encrypting binary data on z/VSE or when the decrypted data are not intended to be human-readable.

## Decrypting on z/VSE

In this section, we show how to encrypt a local file by using the z/OS Java Client, upload it to z/VSE, and decrypt it on z/VSE. In the following example, enter the password in uppercase characters:

```
java -Djava.encryption.facility.debuglevel=1 com.ibm.encryptionfacility.EncryptionFacility
-mode encrypt
-underlyingKey PBEwithSHA1and3DES
-password MYPASSWD
-inputFile mypic.jpg
-outputFile mypic.enc
-iterations 123
```

Then, upload the encrypted file to z/VSE, as shown in Example 4-27.

### Example 4-27 Upload encrypted file

---

```
ftp> put mypic.enc encdata
200 Command okay
150-About to open data connection
  File:VSE.EF.ENCDATA
  Type:Binary Recfm:FB Lrecl:   80 Blksize:   80
  CC=ON UNIX=OFF RECLF=OFF TRCC=OFF CRLF=ON NAT=NO CONT=OFF
  MODE=Stream STRU=File
150 File status okay; about to open data connection
226-Bytes received: 11,576
  Records received: 145
  Transfer Seconds:   .03 ( 377K per second)
  File I/O Seconds:  .01 ( 1130K per second)
226 Closing data connection
ftp: 11576 bytes sent in 0.00Seconds 11576000.00Kbytes/sec.
```

---

The JCL that is shown in Example 4-28 decrypts the encrypted data set and writes the clear data into the clear data set, which is also a VSAM ESDS cluster.

### Example 4-28 Decrypt data set on z/VSE

---

```
* $$ JOB JNM=DECRYPT,DISP=D,CLASS=0
// JOB DECRYPT
// LIBDEF *,SEARCH=(PRD2.SCEEBASE,PRD2.PROD,PRD2.DBASE)
// EXEC IJBEFVSE
DECRYPT
PASSWORD=MYPASSWD
CLRFILE=DD:CLRDATA
ENCFILE=DD:ENCDATA
/*
/&
* $$ EOJ
```

---

Check the job output for any error messages.

### 4.5.3 Public key encryption

Public key encryption (PKE) requires setting up keystores on both platforms. The encrypting site requires an RSA public key; the decrypting site requires a corresponding RSA private key.

The z/OS Java Client requires a Java keystore that contains any keys to be used. Keyman/VSE supports Java keystore (JKS) files; however, an alternative for creating a Java keystore is the use of the `keytool.exe`, which is part of your Java installation and is in the `jre/bin` directory. This section describes how to perform the following tasks:

- ▶ Set up the Java keystore by using the `keytool`, as described in “Encrypting on z/VSE” on page 151.
- ▶ Use Keyman/VSE, as described in “Creating RSA key pair using Keyman/VSE” on page 155.

#### Encrypting on z/VSE

When encrypting on z/VSE, we start on the PC side to create an RSA key pair in a Java keystore. The following tasks must be completed:

- ▶ Export the public key from the Java keystore.
- ▶ Import the public key certificate into Keyman/VSE.
- ▶ Upload the certificate to z/VSE, where it can be accessed by Encryption Facility.

#### Creating RSA key pair by using `keytool.exe`

In a Java keystore, RSA key pairs are always wrapped into a certificate. The following command string creates a private key certificate:

```
keytool -genkey -keyalg "RSA" -alias mykey -keypass mypasswd -keystore efvse.jks -storepass mypasswd
```

An important step is to specify key algorithm *RSA*, because the default is *DSA*, which is not supported on z/VSE. You must specify some personal information to be used when creating the private certificate (see Example 4-29).

*Example 4-29 Input for a private certificate*

---

```
What is your first and last name?  
[Unknown]: Joerg Schmidbauer  
What is the name of your organizational unit?  
[Unknown]: VSE Development  
What is the name of your organization?  
[Unknown]: IBM  
What is the name of your City or Locality?  
[Unknown]: Boeblingen  
What is the name of your State or Province?  
[Unknown]: BW  
What is the two-letter country code for this unit?  
[Unknown]: DE  
Is CN=Joerg Schmidbauer, OU=VSE Development, O=IBM, L=Boeblingen, ST=BW, C=DE correct?  
[no]: y
```

---

The JKS file is now created. A list of the `keytool` parameters is available at this web page:

<https://docs.oracle.com/javase/1.5.0/docs/tooldocs/windows/keytool.html>

### **Exporting public key from Java keystore**

Use the `keytool -list` command to show the contents of the keystore, as shown in the following example:

```
keytool -list -keystore efvse.jks -storepass mypasswd
```

#### *Example 4-30 Keystore output*

---

Keystore type: JKS  
Keystore provider: SUN

Your keystore contains 1 entry

mykey, Oct 29, 2008, PrivateKeyEntry,  
Certificate fingerprint (MD5): 95:20:54:AC:3C:BF:96:42:1D:CF:E6:9E:7E:45:29:D5

---

Now, export the certificate into a binary file, as shown in the following example:

```
keytool -export -alias mykey -file efvse.crt -keystore efvse.jks -storepass mypasswd
```

The following message is displayed:

Certificate stored in file <efvse.crt>

Only the public key was written to the output file. By definition, private keys never leave their original keystore.

### **Importing public key certificate into Keyman/VSE**

If you did not define your z/VSE system in Keyman/VSE, do so now.

To import the public key from the previously created file, select **File** → **Import certificate from file**, as shown in Figure 4-38.

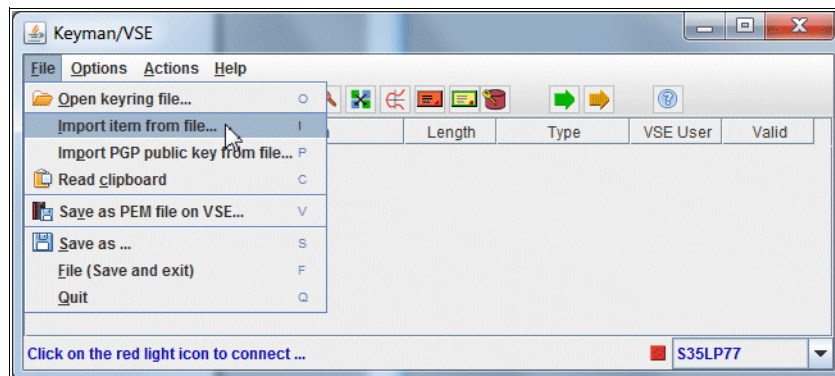


Figure 4-38 Importing certificate from file

Browse to the directory where the `efvse.crt` file is stored and open the file. The certificate is now displayed in the Keyman/VSE main window. It is displayed as a ROOT certificate because it is self-signed.



Right-click the certificate and select **Upload to VSE** (see Figure 4-39).

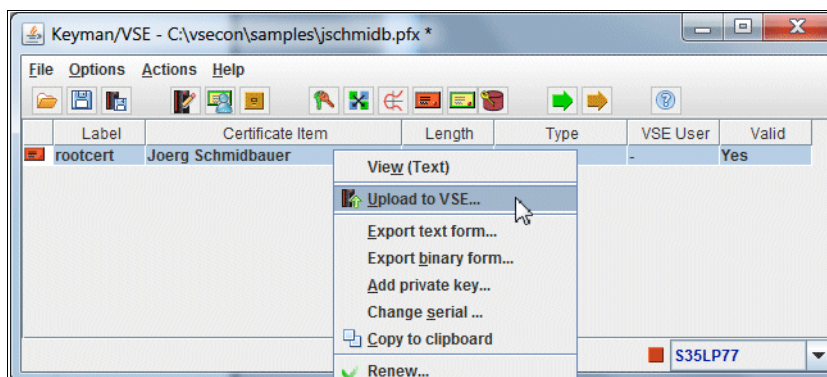


Figure 4-39 Starting upload to z/VSE

### Uploading public key certificate to z/VSE keyring library

Enter a unique member name and select **CERT** as the member type, as shown in Figure 4-40.

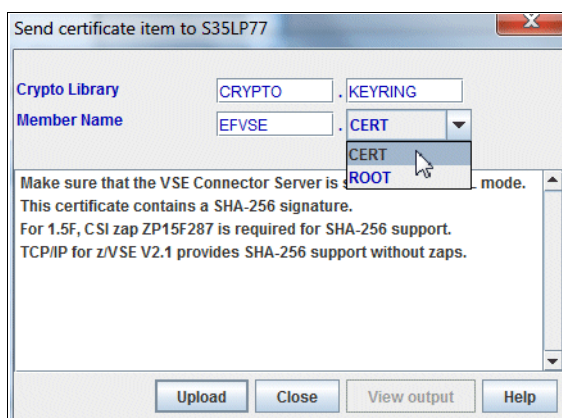


Figure 4-40 Upload public key certificate to z/VSE

Click **Upload**.

The public key is now stored in the crypto library.

Use the LIBR command **LD** to verify that the certificate is stored in sublibrary **CRYPTO.KEYRING**, as shown in Example 4-31.

*Example 4-31 Control result with LIBR command LD*

```
LD EFVSE.*
```

```
DIRECTORY DISPLAY      SUBLIBRARY=CRYPTO.KEYRING      DATE: 2008-10-29
                                                                    TIME: 20:28
```

```
-----
 M E M B E R      CREATION  LAST      BYTES  LIBR CONT SVA  A- R-
 NAME           TYPE      DATE      UPDATE  RECORDS  BLKS  STOR ELIG  MODE
-----
EFVSE     CERT      08-10-29  - -     609 B     1 YES  - - -
L113I RETURN CODE OF LISTDIR IS 0
-----
```

## ***Performing encryption on z/VSE***

The following prerequisites must be met:

- ▶ The public key that is used for encryption is on z/VSE in a CERT member.
- ▶ The corresponding private key is in the Java keystore on the PC side.

In this example, we put a JPG image into the clear data set before running the job that is shown in Example 4-32.

### *Example 4-32 Job to encrypt on z/VSE*

---

```
* $$ JOB JNM=ENCRSA,CLASS=4,DISP=D
// JOB ENCRSA ENCRYPT USING PKE
// LIBDEF *,SEARCH=(PRD2.SCEEBASE,PRD2.PROD,PRD2.DBASE)
// EXEC IJBFEVSE
ENCRYPT
DESC='ENCRYPTION TEST'
CLRAES128
RSA=CRYPTO.KEYRING(EFVSE)
CLRFILE=DD:CLRDATA
ENCFILE=DD:ENCDATA
/*
/&
* $$ EOJ
```

---

Review the job output. Because Encryption Facility first attempts to open a PRVK member with the specific member name, some LIBR errors can occur. Finally, the public key should be taken from the CERT member, as shown in Example 4-33.

### *Example 4-33 Output of the encryption job*

---

```
T037: SSL303E IPDSCRFI failed RC=00000008(LIBROPIF) reason=00000418 00000008
T037: SSL113W IPDSCRFI get for CRYPTO KEYRING EFVSE PRVK failed
T037: SSL303E IPDSCRFI failed RC=000007E7(LIBRCALL) reason=000005D0
INFO: USING RSA PUBLIC KEY FROM CERTIFICATE:
      CRYPTO.KEYRING(EFVSE) (1024 BIT)
```

---

## ***Decrypting locally***

Download the encrypted data set in binary to your workstation, as shown in Example 4-34. Decrypt the file with the z/OS Java Client by using the private key in the Java keystore.

### *Example 4-34 Download the encrypted data set*

---

```
ftp> get file2 mypic.enc
200 Command okay
150-About to open data connection
      File:VSE.EF.ENCDATA
      Type:Binary Recfm:FB Lrecl: 80 Blksize: 80
      CC=ON UNIX=OFF RECLF=OFF TRCC=OFF CRLF=ON NAT=NO CONT=OFF
      MODE=Stream STRU=File
150 File status okay; about to open data connection
226-Bytes sent: 12,432
      Records sent: 1
      Transfer Seconds: .00 ( 12432-bytes)
      File I/O Seconds: .00 ( 12432-bytes)
226 Closing data connection
ftp: 12432 bytes received in 1.00Seconds 12.43Kbytes/sec.
```

---

The following command string performs the decryption:

```
java -Djava.encryption.facility.debuglevel=1 com.ibm.encryptionfacility.EncryptionFacility
-mode decrypt
-keyStoreName efvse.jks
-keyStoreType JKS
-keyStoreCertificateAlias mykey
-password mypasswd
-inputFile mypic.enc
-outputFile mypic2.jpg
```

The password here specifies the keyring file password and must not be confused with the password as specified for password-based encryption.

### ***Encrypting for multiple recipients***

You can specify multiple RSA statements in the same job to allow multiple recipients to decrypt the encrypted data set. This process requires having the public key of each recipient available on z/VSE in a PRVK, or CERT member, as shown in Example 4-35.

*Example 4-35 Job to encrypt with multiple recipients*

---

```
* $$ JOB JNM=ENCMULT,CLASS=4,DISP=D
// JOB ENCMULT ENCRYPT WITH MULTIPLE PUBLIC KEYS
// LIBDEF *,SEARCH=(PRD2.SCEEBASE,PRD2.PROD,PRD2.DBASE)
// EXEC IJBEFVSE
ENCRYPT
DESC='ENCRYPTION TEST'
CLRAES128
RSA=CRYPTO.KEYRING(PUBKEY1)          <- refers to member PUBKEY1.CERT
RSA=CRYPTO.KEYRING(PUBKEY2)
RSA=CRYPTO.KEYRING(PUBKEY3)
RSA=CRYPTO.KEYRING(MYKEY)           <- refers to member MYKEY.PRVK
CLRFILE=DD:PRD2.CONFIG(IPINIT00.L)
ENCFILE=DD:ENCDATA
/*
/&
* $$ E0J
```

---

As shown in Example 4-35, the data key is encrypted with the public keys of three CERT members and one PRVK member. This approach allows decrypting the output data set on three remote systems where the corresponding private RSA keys are present.

The remote systems can be z/OS or z/VSE systems, but also any Java workstation. The encrypted data set can also be decrypted on the same system by using the public key that belongs to the private key that was specified in the last RSA control statement.

The maximum number of RSA statements is 16.

### **Decrypting on z/VSE**

When decrypting on z/VSE, we need a PRVK member that includes a private key. In this case, the setup of the keystore is different because we start on the z/VSE side and export the z/VSE public key to the Java keystore on the workstation.

### ***Creating RSA key pair using Keyman/VSE***

Open the Keyman/VSE tool and create a new RSA key pair. Click the **Generate new RSA** key pair toolbar button, as shown in Figure 4-41 on page 156.

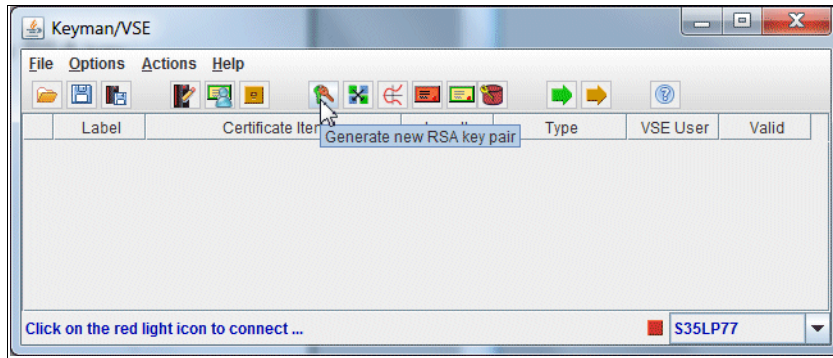


Figure 4-41 Generate a new RSA key pair

The Generate new RSA key window opens, as shown in Figure 4-42. Select the RSA key length, and click **Generate key**.

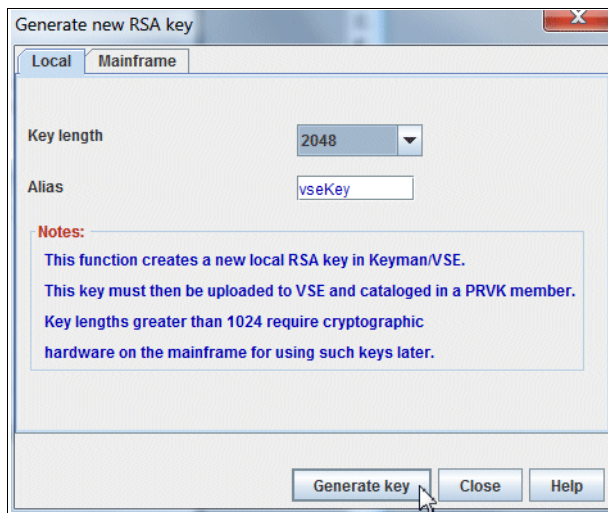


Figure 4-42 Specify key length for new RSA key

### Uploading the private key to z/VSE

Start the z/VSE Connector Server on z/VSE in non-SSL mode and upload the key pair to z/VSE, as shown in Figure 4-43.

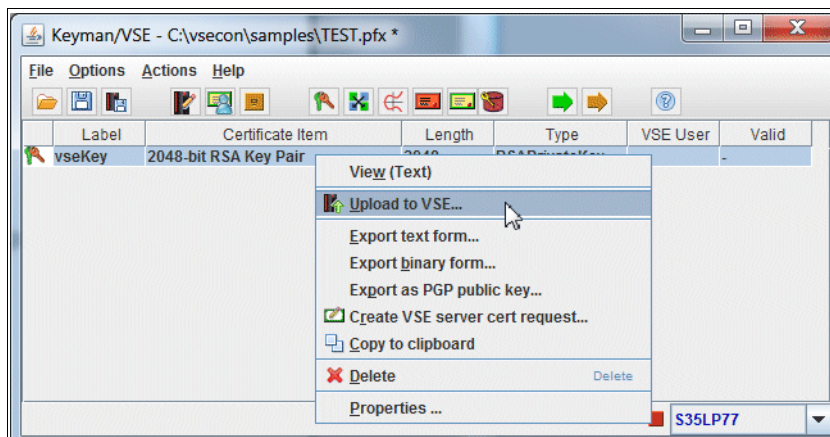


Figure 4-43 Upload key pair to z/VSE using the Keyman/VSE

Because this key is used for decryption with Encryption Facility, we name the library member EFDECR.PRVK, as shown in Figure 4-44.

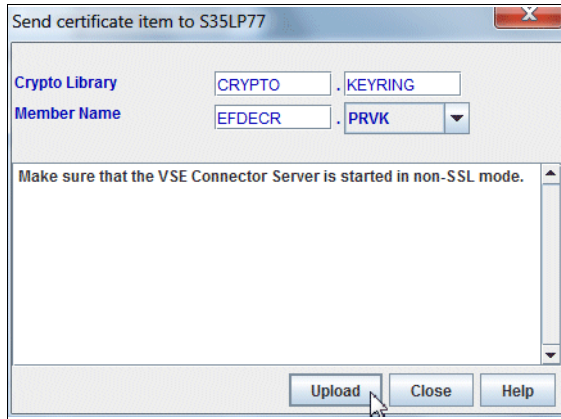


Figure 4-44 Specify a member name

Click **Upload**. The key pair is now stored in library member EFDECR.PRVK in the specified keyring library. The next step is to export the public key into a Java keystore where it can be used by the z/OS Java Client.

### **Exporting the public key into a Java keystore**

Click **Save**, as shown in Figure 4-45.

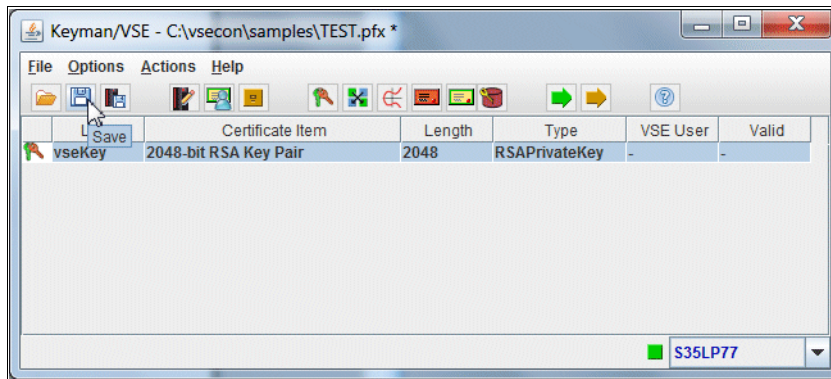


Figure 4-45 Save key pair

In the next window (see Figure 4-46), select tab **JKS options** and enter a JKS file password for the Java keystore.

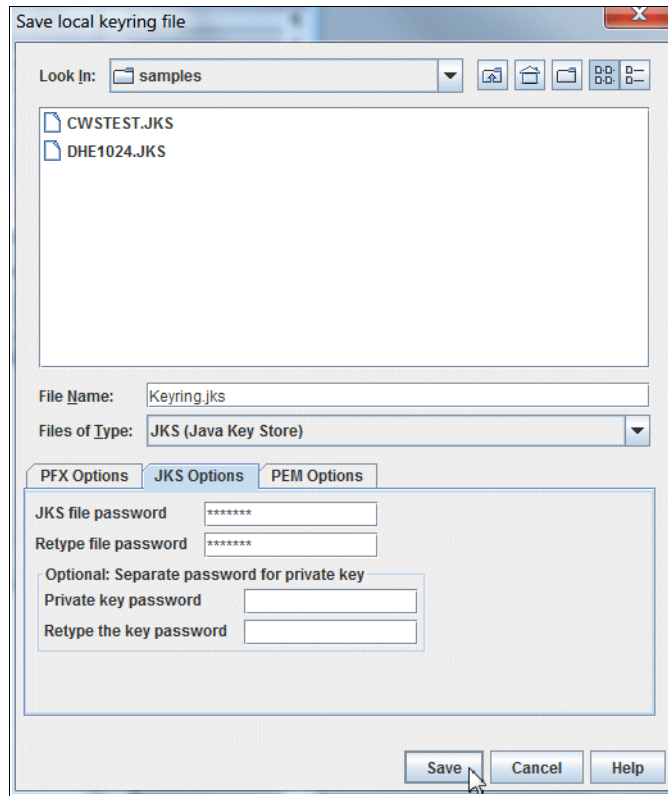


Figure 4-46 Set keyring properties

Click **Save**.

The Enter Personal Information for Private Key window opens, as shown in Figure 4-47. As in a Java keystore, keys are always wrapped into certificates and must enter personal information.

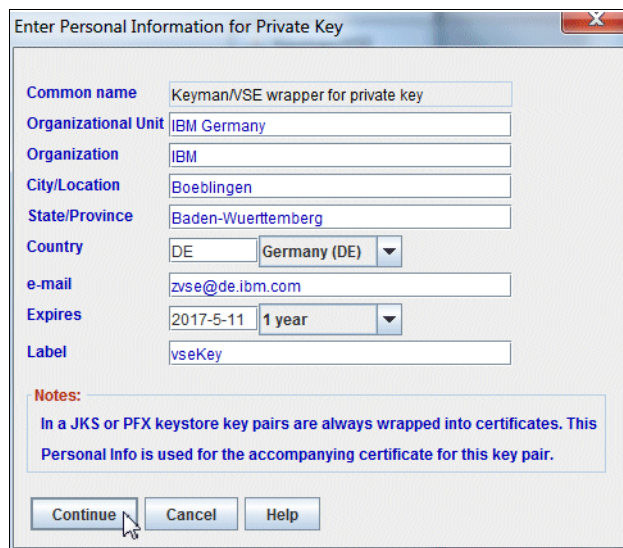


Figure 4-47 Enter the information for the private key

Click **Continue** to return to the main window of Keyman/VSE (see Figure 4-48).

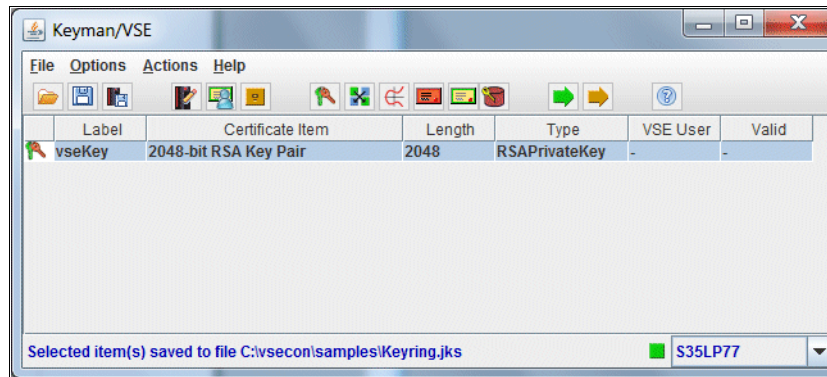


Figure 4-48 Export of public key completed

The Java keystore (Keyring.jks) can now be used directly by the z/OS Java Client.

### **Encrypting locally**

The following command string encrypts a local file with the z/OS Java Client. The alias name in Keyman must match the `keyStoreCertificateAlias` parameter of the Java Client:

```
java -Djava.encryption.facility.debuglevel=1 com.ibm.encryptionfacility.EncryptionFacility
-mode encrypt
-underlyingKey AES16
-keyStoreName keyring.jks
-keyStoreType JKS
-keyStoreCertificateAlias vseKey
-password mypasswd
-inputFile mypic.jpg
-outputFile mypic.enc
```

Upload the encrypted file to z/VSE, as shown in Example 4-36.

#### *Example 4-36 Upload to z/VSE*

---

```
ftp> put mypic.enc encdata
200 Command okay
150-About to open data connection
  File:VSE.EF.ENCDATA
  Type:Binary Recfm:FB Lrecl:   80 Blksize:   80
  CC=ON UNIX=OFF RECLF=OFF TRCC=OFF CRLF=ON NAT=NO CONT=OFF
  MODE=Stream  STRU=File
150 File status okay; about to open data connection
226-Bytes received: 11,584
  Records received: 145
  Transfer Seconds:      .02 ( 566K per second)
  File I/O Seconds:     .01 ( 1131K per second)
226 Closing data connection
ftp: 11584 bytes sent in 0.00Seconds 11584000.00Kbytes/sec.
```

---

Now, you can decrypt the file on z/VSE.



### **Performing decryption on z/VSE**

The last step is the decryption on z/VSE. Example 4-37 shows the job that decrypts a file on z/VSE. In the JCL, you must specify the member name of the PRVK that includes the private key (EFDECR) that was uploaded to z/VSE, as described in “Uploading the private key to z/VSE” on page 156.

*Example 4-37 Job to decrypt a file*

---

```
* $$ JOB JNM=DECRSA,CLASS=4,DISP=D
// JOB DECRSA DECRYPT USING PRVK
// LIBDEF *,SEARCH=(PRD2.SCEEBASE,PRD2.PROD,PRD2.DBASE)
// EXEC IJBEFVSE
DECRYPT
RSA=CRYPTO.KEYRING(EFDECR)
CLRFILE=DD:PRD2.CONFIG(MYPIC.JPG)
ENCFILE=DD:ENCDATA
/*
/&
* $$ E0J
```

---

## **4.6 Software-based encryption with Encryption Facility for z/VSE V1R2**

This section describes the setup of PGP-based encryption in various scenarios, including z/VSE, z/OS, and workstation platforms.

Pretty Good Privacy (PGP) was created by Phil Zimmermann in 1991. PGP became an open standard in 1998 and is described in RFCs 2440 and 4880 (OpenPGP Message Format). PGP is often used for signing and encrypting emails.

In z/OS, support for PGP-based encryption is available with Encryption Facility for z/OS V1R2. In z/VSE, the support is provided with z/VSE V4R2M1 and the second release of Encryption Facility for z/VSE. OpenPGP support now provides the following features:

- ▶ Data format compatibility to Encryption Facility for z/OS V1.2 and Open Source programs such as GNU Privacy Guard (GnuPG).
- ▶ More algorithms that can be used to encrypt data.

For more information about OpenPGP message format (RFC 4880), see this web page:

<http://tools.ietf.org/html/rfc4880>

For more information about the OpenPGP support on z/VSE (including a list of all commands and command options), see *z/VSE Administration*, SC34-2692, which is available at:

<http://www.ibm.com/systems/z/os/zvse/documentation/#vse>

In our test setup, we used the following software:

- ▶ z/VSE V6R2M0
- ▶ TCP/IP for z/VSE v2.2
- ▶ VSE Connector Server as part of z/VSE V6R2M0 (job STARTVCS)
- ▶ Java 8 from Oracle
- ▶ Keyman/VSE from July 2017
- ▶ Encryption Facility for z/VSE V1R2 OpenPGP



- ▶ GNU Privacy Guard 1.4.7
- ▶ GNU Privacy Assistant 0.7.6
- ▶ GPGe extension for Windows Explorer

### 4.6.1 Prerequisites

To use Encryption Facility for z/VSE V1R2, the following prerequisites must be met:

- ▶ An activated CP Assist Facility (CPACF) is used
- ▶ TCP/IP for z/VSE for public key encryption is used
- ▶ A Crypto Express feature for processing 2048 or 4096 bit RSA keys is used
- ▶ At least 8 MB partition size is available because the tool is a Language Environment/VSE application

### 4.6.2 Differences in Encryption Facility between z/VSE V1R1 and V1R2

Encryption Facility for z/VSE V1R1 provides (ships) a single utility IJBEFVSE, which is described in 4.5, “Software-based encryption with Encryption Facility for z/VSE V1R1” on page 145. Another utility (IJBEFPGP) is included with Encryption Facility for z/VSE V1R2.

Figure 4-49 shows the relationship between the two facilities.

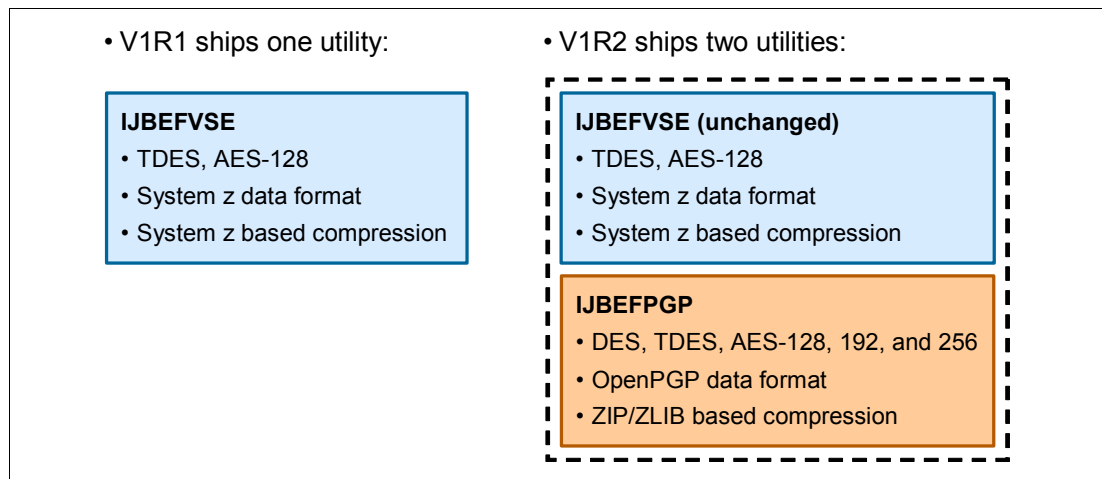


Figure 4-49 Encryption Facility for z/VSE V1R1 versus V1R2

Although IJBEFVSE is included unchanged (except for certain corrective service), the IJBEFPGP utility provides the support for OpenPGP encryption. Encryption Facility for z/VSE V1R1 cannot be ordered now that V1R2 is available.

The basic functionality that is provided by the two utilities is similar; however, the algorithms that are used and the data format are different.

Both utilities provide password-based encryption, but the way that the encryption key is calculated from the password differs in IJBEFVSE and IJBEFPGP. Both utilities provide public key encryption, but the encrypted session key is stored differently in the encrypted data set. However, in both cases, up to 16 recipients can be specified.

When the same encryption algorithm (for example AES-128) is used with both versions of Encryption Facility for z/VSE, the encrypted data is equally safe; however, the encrypted data format differs. The decision of whether to use the IJBEFVSE utility or the IJBEFPGP utility depends on your environment. Consider the following questions:

- ▶ What are the tools and platforms with which you want to exchange data?
- ▶ Do you want to use an encryption algorithm that is not supported by the IJBEFVSE utility (for example, AES-256)?

The two utilities include the following differences:

- ▶ Encrypted data format  
IJBEFVSE supports the System z data format; IJBEFPGP supports the OpenPGP data format.
- ▶ Compatibility  
IJBEFVSE provides compatibility to the IBM provided Java client and the Decryption client for z/OS. IJBEFPGP provides compatibility to PGP implementations.
- ▶ Algorithms  
IJBEFPGP supports more algorithms and provides better System z hardware usage (for example, AES-256 and SHA-512).
- ▶ Compression  
ZIP/ZLIB versus compression from System z differs. Used in IJBEFPGP, ZIP/ZLIB compression is done in software; used in IJBEFVSE, System z compression is hardware-accelerated.

**Note:** ZIP/ZLIB compression is much slower than System z compression. For large amounts of data, we recommend the use of IJBEFVSE with System z compression. Use IJBEFPGP for exchanging data in OpenPGP format with workstation or z/OS platforms.

### 4.6.3 Downloading the prerequisite programs

In addition to Encryption Facility for z/OS V1.2, we used the open source software GNU Privacy Guard (GnuPG) in our tests. GnuPG claims to be the reference implementation for OpenPGP. A free GUI is available for Windows, which is the Gnu PG for Windows (GPG4win), including an extension to the Microsoft Windows Explorer, named GPGee. We downloaded and installed the packages that are described next. Encryption Facility for z/OS is an optional feature that must be ordered.

#### GNU Privacy Guard

Download the GNU Privacy Guard (GnuPG) installation package from this website:

<https://www.gnupg.org>

The package is available as one file (named `gnupg-1.4.7.tar.gz`). Unpack the file contents into a new folder. We used version 1.4.7, but you can download the latest version.

#### GPG4win

GPG4win is a free Windows GUI and includes the Windows Explorer extension GPGee. The documentation is available in German only. The file is available from this website:

<https://www.gpg4win.org>

The installation file is a Windows executable (named `gpg4win-1.1.3.exe`). Double-click the file and follow the installation instructions. We used version 1.1.3, but you can download the latest version.

## Keyman/VSE

Download the Keyman/VSE tool from the z/VSE homepage:

<http://www.ibm.com/systems/z/os/zvse/downloads/>

The Keyman/VSE tool requires the installation of the VSE Connector Client, which can be downloaded from the same link.

### 4.6.4 Usage hints

Encryption Facility for z/VSE can be used to encrypt data for creating the following encrypted components:

- ▶ Archives
- ▶ Data sets on tape or disk for physical delivery or transmission over an insecure network

**Important:** If you plan to delete your original unencrypted data after encryption, we strongly recommend that you perform the following tasks:

- ▶ Verify that your data can be decrypted successfully before destroying any original data.
- ▶ Keep a copy of this particular version of Encryption Facility for z/VSE to ensure that you can perform the decryption again.

### 4.6.5 Flexible support of record and stream data

The OpenPGP standard does not consider record-based data. However, record-based data is heavily used in mainframe environments. Therefore, providing PGP encryption without supporting record-based data (such as VSAM data sets) does not make sense.

The IJBEFPGP utility provides support for record-based data by using the `USE_RECORDINFO` parameter. The implementation uses one feature of the OpenPGP standard, which allows more private or experimental data packets into a PGP encrypted data set. Such data packets are ignored by other PGP implementations. The option to be used only when encrypting and decrypting on z/VSE. The GnuPG and z/OS implementations cannot handle the metadata correctly.

When option `USE_RECORDINFO` is used, a private data packet that contains the `LRECL`, `RECFM`, and `BLKSIZE` of the original clear input data set is added to the encrypted data set. The length of each data record is maintained by providing more bytes to each record containing its actual length. This data is recognized as garbage by other PGP implementations. On z/VSE, this information is used when decrypting to restore the decrypted data with its original record structure.

Our recommendation is to use the `USE_RECORDINFO` option when encrypting and decrypting on z/VSE and the clear input data set includes a record-based structure. This recommendation includes VSAM data sets and SAM/BAM data sets. Input tapes can be considered as data streams where the option is not necessary. Also, encrypting z/VSE library members does not require the use of this option.

## 4.6.6 Considerations on compression

The use of compression usually speeds up the encryption process. However, compression makes the process even slower in certain scenarios. Consider the following points when compression is used:

- ▶ Compression is always applied before encryption.
- ▶ When compression is used, less data often must be encrypted, except when clear data is binary, such as .jpg, where the compression ratio is small (sometimes zero). In rare situations, compressed data can become even bigger than uncompressed data using ZIP file format.
- ▶ Compression adds security by removing any recognizable patterns from original clear data before encryption. However, the use of cipher block chaining also removes any clear text data patterns.
- ▶ Compression is usually slower than decompression because a compression dictionary must be built during compression. Decompression is a table lookup.
- ▶ When encrypting and compressing small files, the process can become slower compared to not using compression because of the compression overhead that is involved in building the dictionary.
- ▶ ZIP/ZLIB compression is entirely done in software; System z compression is done in microcode.

## 4.6.7 Password-based encryption

Password-based encryption does not require any keystores. The encryption key is directly derived from the password. Encryption Facility for z/VSE always converts the EBCDIC password that is specified with JCL to ASCII. Because passwords are case-sensitive, specify your password correctly in JCL and on any other related platform.

Encryption Facility uses the following code pages by default:

- ▶ ASCII code page: IBM-850
- ▶ EBCDIC code page: IBM-1047

You can change the code page by using the ASCII\_CODEPAGE and EBCDIC\_CODEPAGE parameters. You should specify the code page parameters *before* the S2K\_PASSWORD parameter to have your code page active when translating the password.

The following sections describe how to use password-based encryption and exchange the encrypted data with a workstation platform and with z/OS.

### Encrypting on z/VSE, decrypt on workstation

The JCL that is shown in Example 4-38 encrypts a VSAM data set (CLRDATA) by using password-based encryption (PBE).

*Example 4-38 Job for password-based encryption*

---

```
* $$ JOB JNM=EFPGP,CLASS=0,DISP=D
* $$ LST DISP=D,CLASS=A,RBS=100
// JOB EFPGP TEST OPENPGP SUPPORT
// LIBDEF *,SEARCH=(PRD2.SCEEBASE,PRD2.PROD,PRD2.DBASE)
// EXEC IJBEPGP
PB_ENCRYPT
S2K_PASSPHRASE=MYPASSWD
```

```
S2K_CIPHER_NAME=AES_128
CLRFILE=DD:CLRDATA
ENCFILE=DD:ENCDATA
/*
/&
* $$ E0J
```

---

After downloading the encrypted data set in binary to a Windows PC, it can be decrypted by using the GPGe tool (see Figure 4-50). The file extension must be .gpg; otherwise, GPGe does not display its pop-up menu.

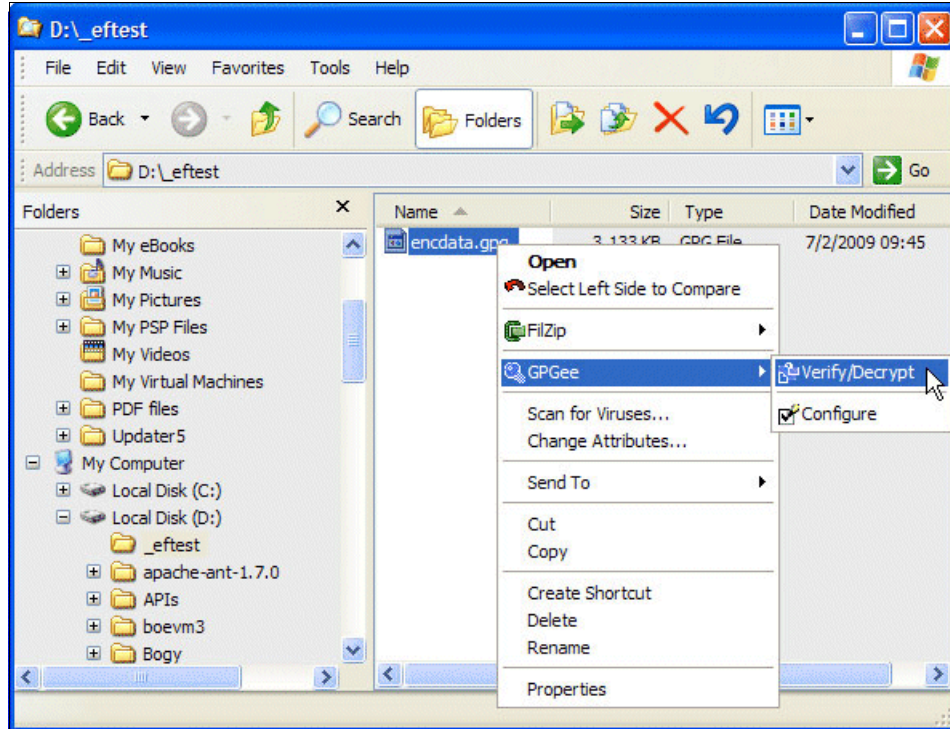


Figure 4-50 Verify or decrypt a file with GPGe

You are prompted for the password, as shown in Figure 4-51.

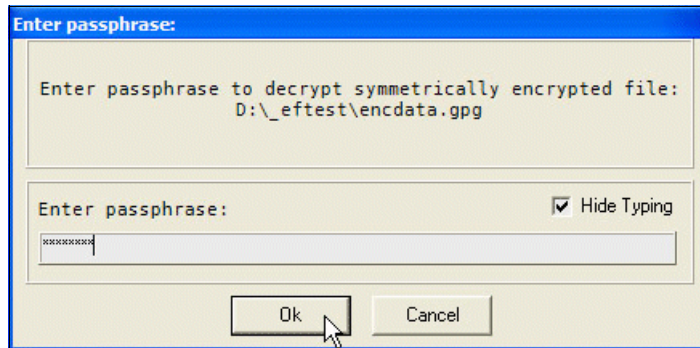


Figure 4-51 Password window of GPGe for decryption

Enter the password and click **OK**.



When the decryption process finishes, a window opens, as shown in Figure 4-52.

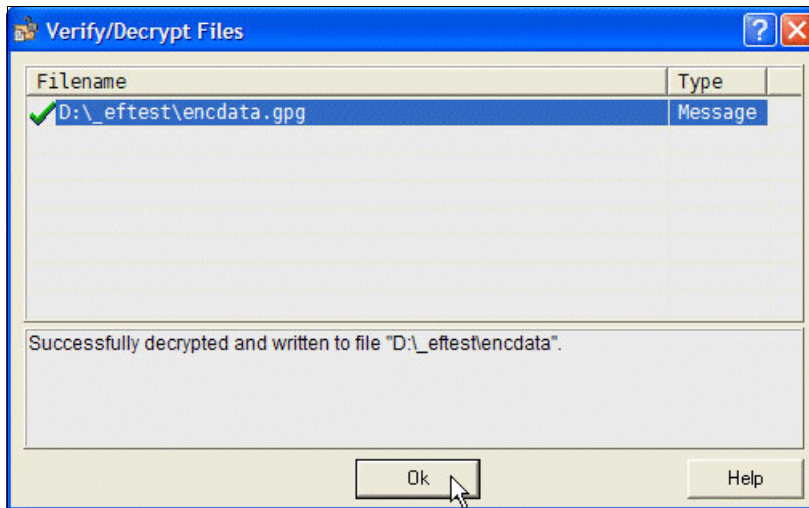


Figure 4-52 Message dialog box after decrypting a file with GnuPG

### Encrypting on workstation and decrypting on z/VSE

In this section, we show how to encrypt a local file by using GPGe, upload it to z/VSE, and decrypt it on z/VSE.

Figure 4-53 shows the pop-up menu of the GPGe tool for encryption.

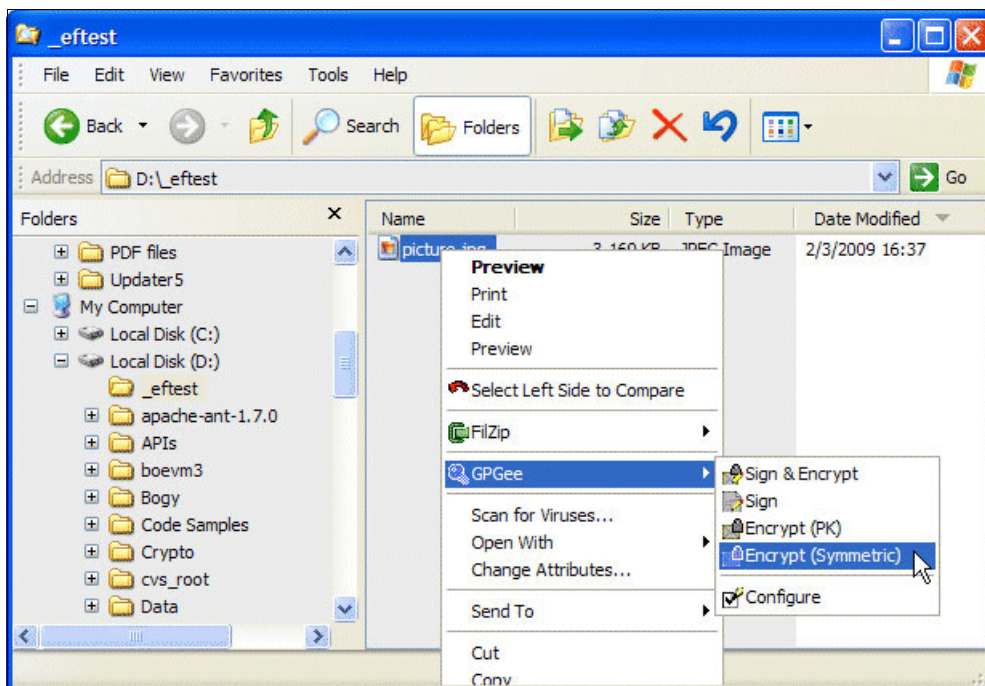


Figure 4-53 Symmetric encryption with GPGe

Enter your password, as shown in Figure 4-54. Use uppercase characters because the z/VSE JCL also specifies the password in uppercase. Then, click **OK**.

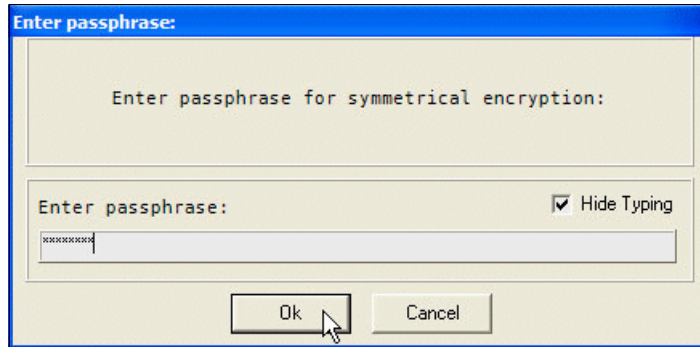


Figure 4-54 Password dialog box of GPGe for encryption

GPG now creates an encrypted output file, `picture.jpg.gpg`, which must be transferred to z/VSE in binary. In this example, we show you how to upload the file into a VSAM ESDS file with file name ENCDATA.

The JCL that is shown in Example 4-39 decrypts the encrypted data set and writes the clear data into another VSAM ESDS data set CLRDATA.

Example 4-39 Symmetric decryption using IJBEFPGP

---

```
* $$ JOB JNM=EFPGP,CLASS=0,DISP=D
* $$ LST DISP=D,CLASS=A,RBS=100
// JOB EFPGP TEST OPENPGP SUPPORT
// LIBDEF *,SEARCH=(PRD2.SCEEBASE,PRD2.PROD,PRD2.DBASE)
// EXEC IJBEFPGP
DECRYPT
S2K_PASSPHRASE=MYPASSWD
CLRFILE=DD:CLRDATA
ENCFILE=DD:ENCDATA
/*
/&
* $$ EOJ
```

---

Check the job output in the VSE/POWER list queue for messages and return codes.

## Encrypting on z/OS and decrypting on z/VSE

The shell script that is shown in Example 4-40 encrypts a file in a UNIX System Services shell on z/OS by using a password.

Example 4-40 Symmetric encryption on z/OS with Encryption Facility for z/OS

---

```
#!/bin/sh
#-----
# Encryption Facility for z/OS V1.2 PBE
#-----
export LIBPATH=$LIBPATH:/usr/lib/java_runtime
java -jar /usr/lpp/encryptionfacility/CSDEncryptionFacility.jar
-o $1.gpg
-cipher-name AES_128
-s2k-cipher-name TRIPLE_DES
-s2k-digest-name SHA256
```

```
-c $1
```

---

Example 4-41 shows the command to run this script and the resulting messages.

*Example 4-41 Output messages of symmetric encryption on z/OS*

---

```
./pbe.sh picture.jpg
CSD1001A Enter passphrase for passphrase-based encryption:
MYPASSWD
CSD0000A Confirm passphrase:
MYPASSWD
CSD0051I Command processing has completed successfully.
```

---

After transferring the encrypted file to z/VSE through binary FTP, it can be decrypted with the same JCL that is shown in Example 4-39 on page 167.

### Encrypting on z/VSE and decrypting on z/OS

To encrypt a data set on z/VSE, you can use the same JCL as described in “Encrypting on z/VSE, decrypt on workstation” on page 164. After transferring the encrypted data set to a z/OS system, it can be decrypted by using Encryption Facility for z/OS.

Use the shell script that is shown in Example 4-42 to perform the decryption on z/OS.

*Example 4-42 Symmetric decryption on z/OS*

---

```
#!/bin/sh
#-----
# Encryption Facility for z/OS V1.2 PBD
#-----
export LIBPATH=$LIBPATH:/usr/lib/java_runtime
java -jar /usr/lpp/encryptionfacility/CSEncryptionFacility.jar
-o $1
-cipher-name AES_128
-s2k-cipher-name TRIPLE_DES
-s2k-digest-name SHA256
-d $1
```

---

Example 4-43 shows the decryption and the related messages.

*Example 4-43 Output messages of symmetric decryption on z/OS*

---

```
./pbd.sh picture.jpg.gpg
CSD1002A Enter passphrase for passphrase-based decryption:
mypasswd
CSD0051I Command processing has completed successfully.
```

---

## 4.6.8 Public key encryption

The section describes how to use public key encryption, including the creation and distribution of RSA keys. Public key encryption requires the setup of keystores on all involved platforms. The encrypting site requires a public key; the decrypting site requires the corresponding private key.



The process generates a random encryption key that is used for data encryption. Then, the encryption key is encrypted with one or more given public keys. This approach allows the owner of a corresponding private key to decrypt the encryption key and in turn, decrypt the data.

Usually, exporting any private keys from a keystore is not possible. Therefore, the process of setting up the keys always starts on the platform where data is decrypted. Public keys can then be exported from a keystore and transferred to the encrypting platform.

**Note:** PGP uses DSA keys by default; z/VSE supports RSA keys only.

### Setting up the keys for encryption with GnuPG

When encrypting on a workstation by using GnuPG, the following process is used (as shown in Figure 4-55):

1. Create an RSA key pair with Keyman/VSE.
2. Upload the private key to z/VSE.
3. Export the PGP public key into a local text file.
4. Import the file into the GnuPG keystore.

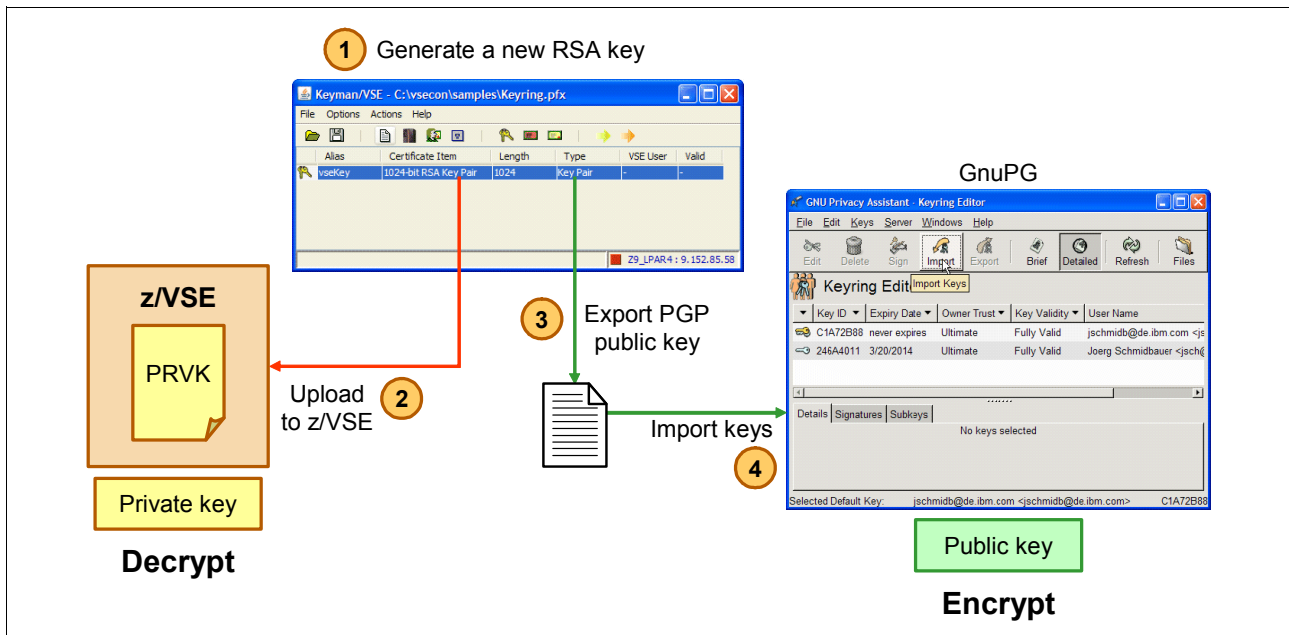


Figure 4-55 Setting up the keys for encryption with GnuPG

Now, the public key is available for encryption in the GnuPG keystore; the private key is stored on z/VSE for decryption.

### Setting up the keys for encryption on z/VSE

When encrypting on z/VSE, the following process is used (as shown in Figure 4-56 on page 170):

1. Generate the key by using GnuPG.
2. Export the public key into a local text file.
3. Keyman/VSE imports this file.
4. Keyman/VSE internally converts the PGP public key into an x.509 certificate, which is then uploaded into a z/VSE library member, (the CERT member).

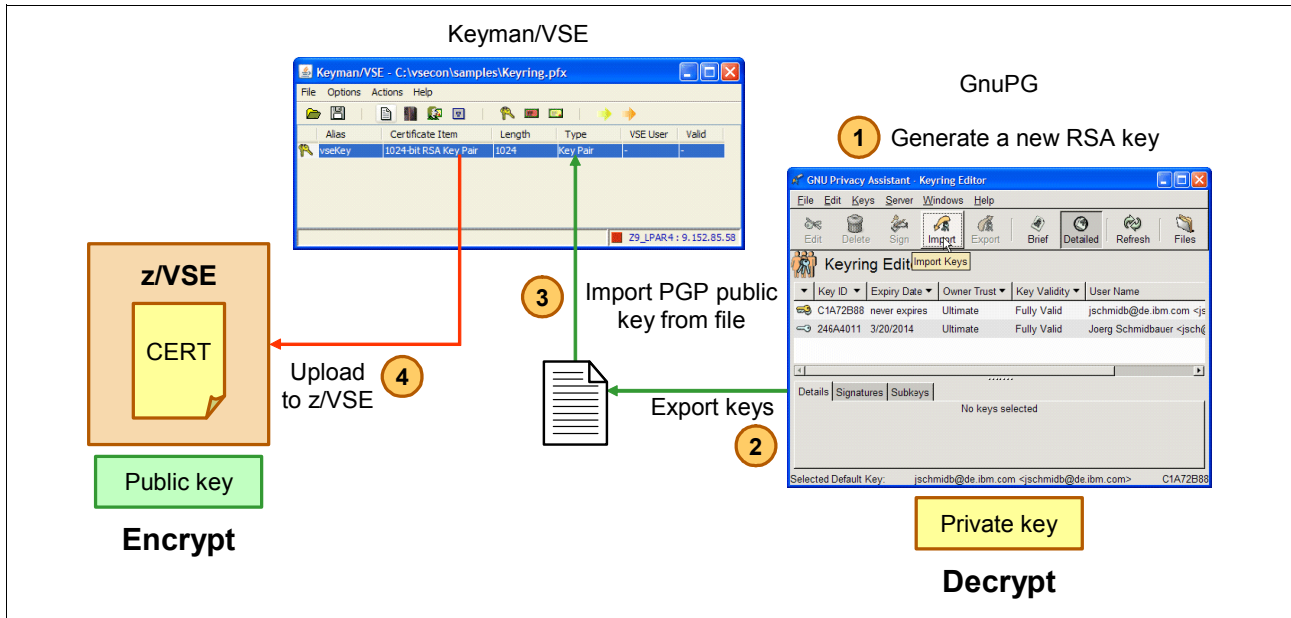


Figure 4-56 Setting up the keys for encryption on z/VSE

### Creating an RSA key pair by using Keyman/VSE

When decrypting on z/VSE, you must have a PRVK member that contains a private key. In this case, the setup of your keystore is different, because you start on the VSE side and export the z/VSE public key to the GnuPG keystore on your Windows PC.

An overview of this process is described in “Setting up the keys for encryption with GnuPG” on page 169.

Open the Keyman/VSE tool and create an RSA key pair, as shown in Figure 4-57.

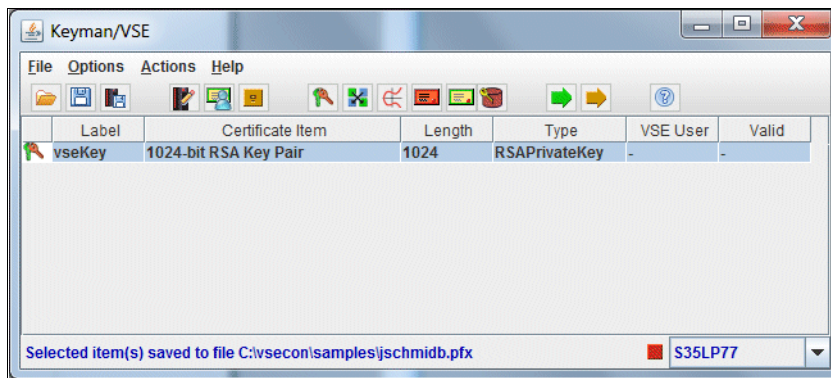


Figure 4-57 Keyman/VSE main window with a created RSA key

Start the z/VSE Connector Server on z/VSE in non-SSL mode and upload the key pair to z/VSE.

Because this key is used for PGP decryption, you name the library member PGPDECR.PRVK, as shown in Figure 4-58 on page 171.

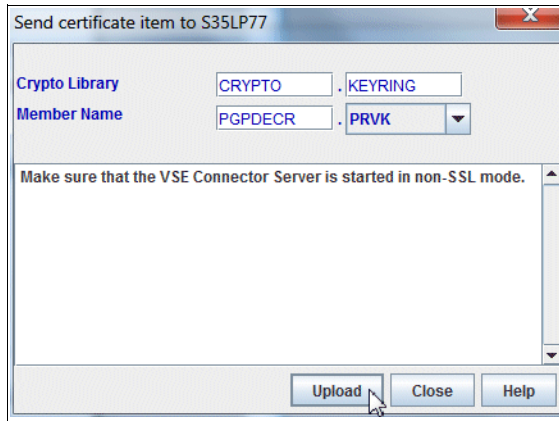


Figure 4-58 Send certificate item to the VSE

### Exporting the public key as PGP public key file

In the Keyman/VSE main window, right-click the RSA key and select **Export PGP public key**, as shown in Figure 4-59.

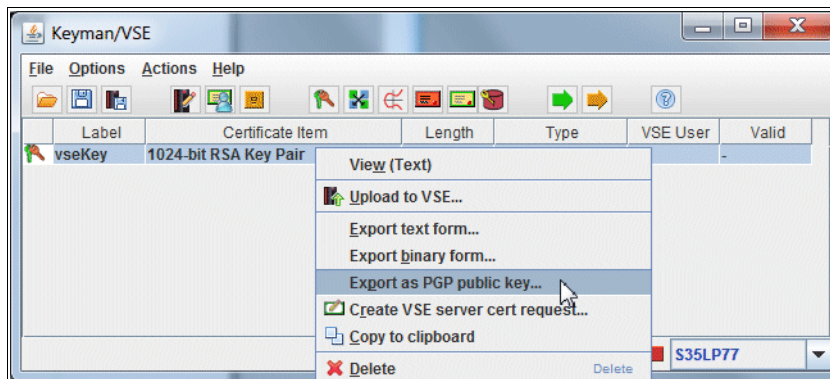


Figure 4-59 Export PGP public key in Keyman/VSE

Specify the name of the output file, as shown in Figure 4-60. Click **Save**.

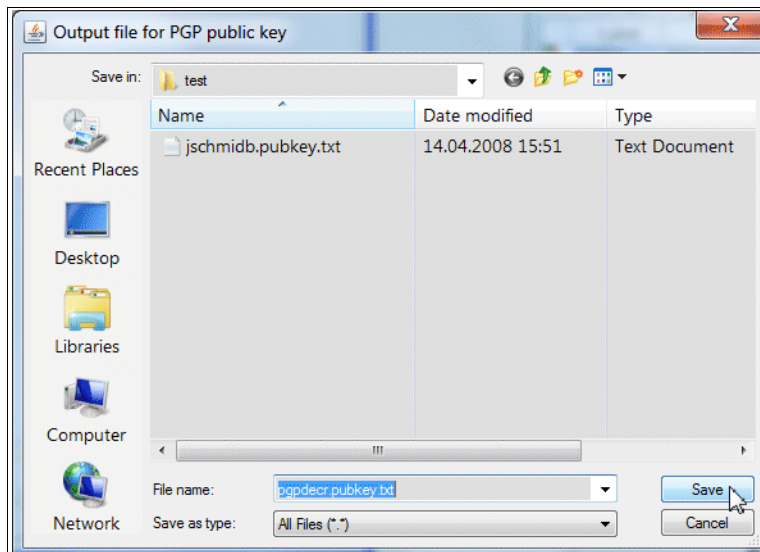


Figure 4-60 Specifying the file name

A PGP public key is always associated with various personal information about the owner of the key (see Figure 4-61). Enter your personal information and click **OK**.

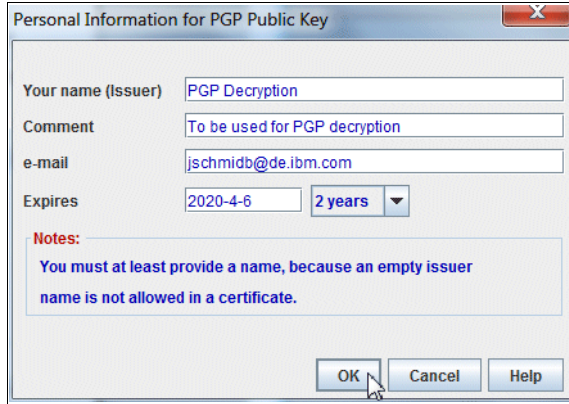


Figure 4-61 Entering your personal information

The output file is similar to the output that is shown in Example 4-44.

*Example 4-44 PGP public key block*

---

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: IBM Encryption Facility for z/VSE V1.2

xo0ESAYBqwEEAIWsS1KT6aM0qdBrBDHat0wiAQ1jbtYw6GWxcpf0/mL4RYA/371xxbV109BkMJzk
W5JNX4MYodUiCZ7B98Wda8kMs90xtyEb6bikVD8W228b1m8K5amg5NRTTztYoH3exwtItq31oI11
QHI2AQRCSHY571KGCTARInqf8/DQPtBpABEBAAHCuwQfAQIAJQUCSAYBqwIeAQIbDwULAgkIBwUV
AgoJCAMWAQICF4AFCQEDt4AACGkQj0AvFgM5ohcYnQP9GMWdqoRa6rKMI9C7wnKKVHaAE1uCY8dA
SWTALHrLufR+5Ua10nBE36YcGGxN/NNZu4C02t551+Lro4Lh3dnU8TtP1kx2w0eMTtoobDZ2n1ivv
8G1T0AqdyW09b8qJ53pa7sZKa1ZVy1fAESWUiXBfUPHEz4bJUMP78cmx/Gx8ssrNJBHUCBEZWNj
cn1wdG1vbiA8anNjaG1pZGJAZGUuWJtLmNvbT7CuwQTAQIAJQUCSAYBrAIeAQIbDwULAgkIBwUV
AgoJCAMWAQICF4AFCQEDt4AACGkQj0AvFgM5ohc7aAP9GMg1gDR3z5YNvHwAi3LXzyi0kae/wh1z
fE60myjzpmPNy2iJ+nVfQXCzuPrWYeA0sWVVLDrseVGJkjqKfaUDCsxoAoEprUHaFc16JsFa2YCB
SIfBzrhMyR0mFJwAygTnSuy7rYmr1Vou065mfkvidV1JBXTVHXIwD9bK1093F0g=
=zugp
-----END PGP PUBLIC KEY BLOCK-----
```

---

The file can now be imported into the GnuPG keystore.



## Importing the public key into the GnuPG keystore

In the Gnu Privacy Assistant GUI, select **Keys** → **Import Keys**, as shown in Figure 4-62.

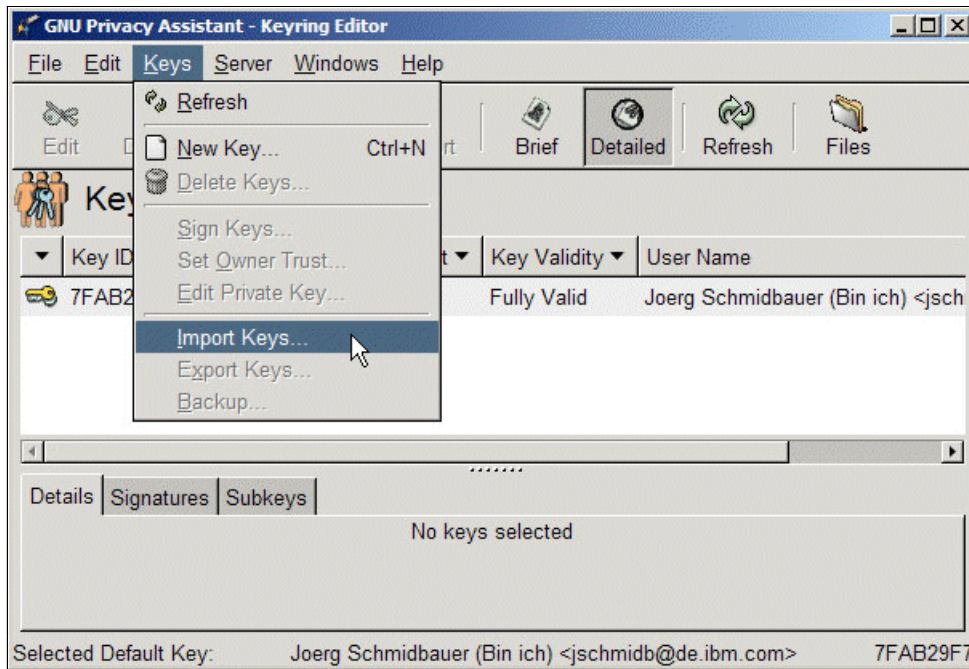


Figure 4-62 Importing public key

Figure 4-63 shows the public key file.

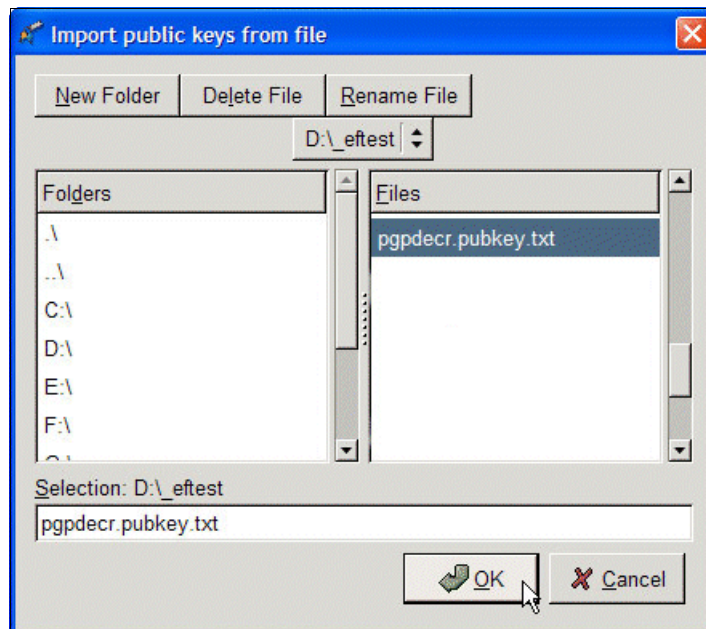


Figure 4-63 Selecting public key file in GNU Privacy Assistant

Click **OK**.

After importing the public key into the GnuPG keystore, window opens, as shown in Figure 4-64.

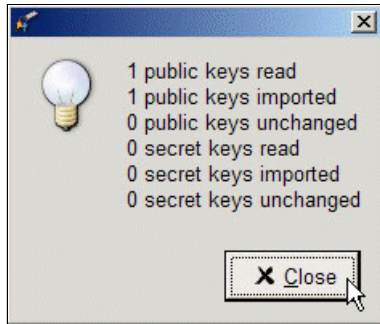


Figure 4-64 GNU Privacy Assistant message

Click **Close**.

Figure 4-65 shows the new key in the GUI as public key, which includes only a public part.

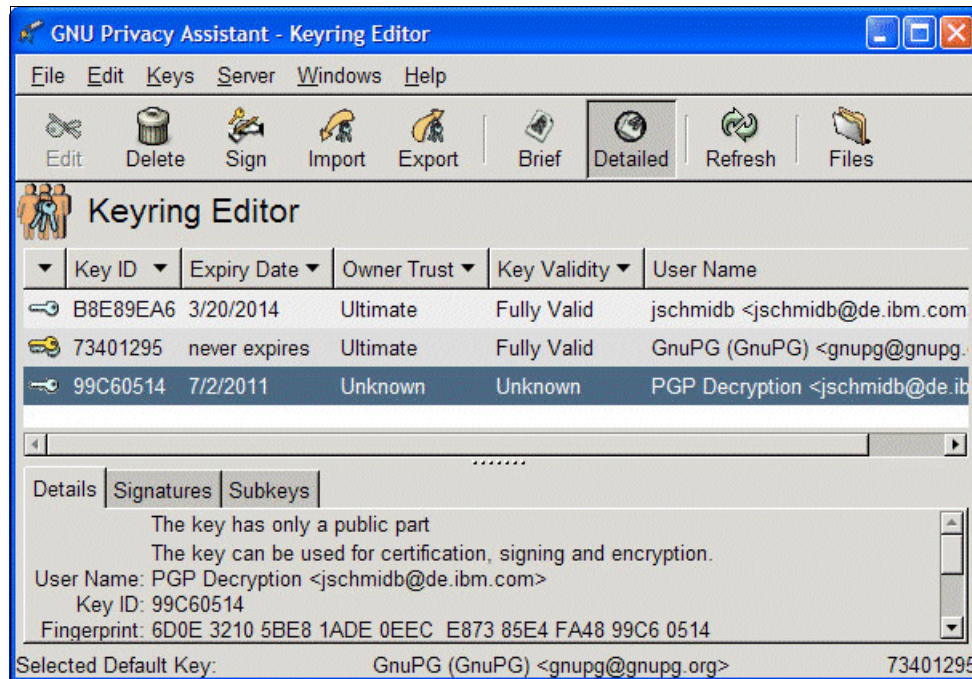


Figure 4-65 GNU Privacy Assistant main window

The process of setting up the keys for encryption with GnuPG is now complete.

To set up the keys for encryption on z/VSE, follow the steps that are described in the following sections:

- ▶ “Creating an RSA key pair by using GnuPG” on page 175
- ▶ “Exporting the public key from the GnuPG keystore” on page 179
- ▶ “Importing the PGP public key into Keyman/VSE” on page 181
- ▶ “Uploading PGP public key to z/VSE” on page 183

## Creating an RSA key pair by using GnuPG

With GnuPG, the following methods can be used to set up the keystore:

- ▶ GnuPG CLI
- ▶ GNU Privacy Assistant GUI

After setting up the keystore by following the steps that are described in this section, you export the public key into a file, as described in “Exporting the public key from the GnuPG keystore” on page 179.

### *Using the GnuPG CLI*

By using this method, you create an RSA key pair by using the GnuPG command-line tool. You enter the `gpg --gen-key` command and follow the prompts, as shown in Example 4-45 on page 176.

**Important:** Be sure to select RSA as the key type.

*Example 4-45 Creating an RSA key by using the GnuPG CLI*

---

```
C:\Program Files\GNU\GnuPG\pub>gpg --gen-key
gpg (GnuPG) 1.4.7; Copyright (C) 2006 Free Software Foundation, Inc.
This program comes with ABSOLUTELY NO WARRANTY.
This is free software, and you are welcome to redistribute it
under certain conditions. See the file COPYING for details.

Please select what kind of key you want:
  (1) DSA and Elgamal (default)
  (2) DSA (sign only)
  (5) RSA (sign only)
Your selection? 5
RSA keys may be between 1024 and 4096 bits long.
What keysize do you want? (2048)
Requested keysize is 2048 bits
Please specify how long the key should be valid.
  0 = key does not expire
  <n> = key expires in n days
  <n>w = key expires in n weeks
  <n>m = key expires in n months
  <n>y = key expires in n years
Key is valid for? (0) 1y
Key expires at 10/22/08 14:05:18
Is this correct? (y/N) y
. . .
Real name: Joerg Schmidbauer
Email address: jschmidb@de.ibm.com
Comment: Blah
You selected this USER-ID:
  "Joerg Schmidbauer (It's me) <jschmidb@de.ibm.com>"

Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit? o
You need a Passphrase to protect your secret key.
. . .
gpg: key E57429F5 marked as ultimately trusted
public and secret key created and signed.

gpg: checking the trustdb
gpg: 3 marginal(s) needed, 1 complete(s) needed, PGP trust model
gpg: depth: 0 valid: 3 signed: 3 trust: 0-, 0q, 0n, 0m, 0f, 3u
gpg: depth: 1 valid: 3 signed: 0 trust: 0-, 1q, 0n, 1m, 1f, 0u
gpg: next trustdb check due at 2008-10-22
pub 2048R/E57429F5 2007-10-23 [expires: 2008-10-22]
   Key fingerprint = 7B53 8429 007F BA9B C064 3227 EADE 6428 E574 29F5
uid                               Joerg Schmidbauer (It's me) <jschmidb@de.ibm.com>

Note that this key cannot be used for encryption. You may want to use
the command "--edit-key" to generate a subkey for this purpose.
```

---



### Using the GNU Privacy Assistant GUI

By using this method of creating the RSA key, you use the GNU Privacy Assistant tool.

In the GNU Privacy Assistant main window, select **Keys** → **New Key**, as shown in Figure 4-66.

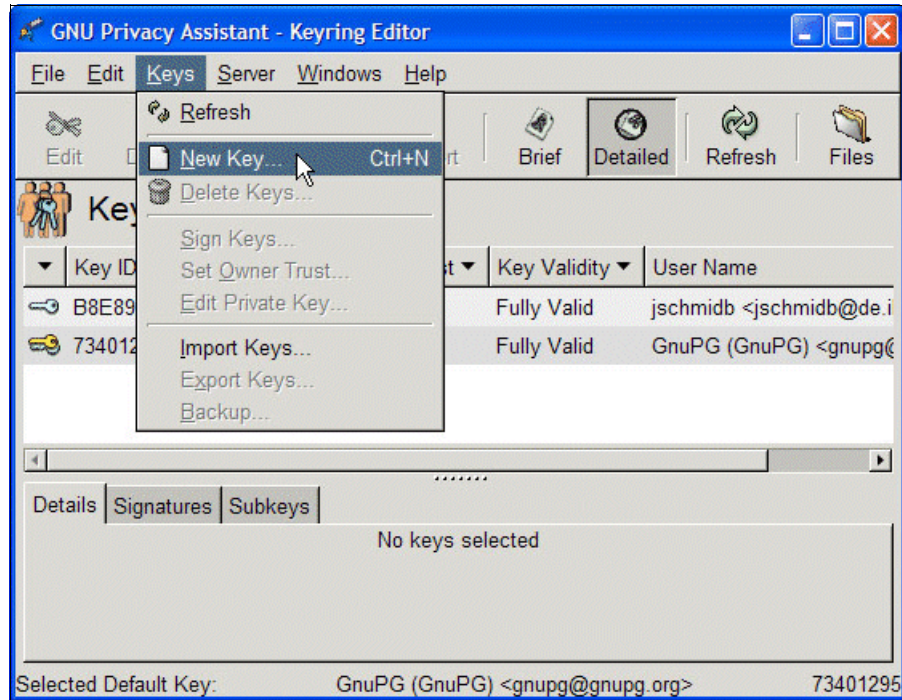


Figure 4-66 Generate RSA key with GNU Privacy Assistant

In the next window, select **RSA** as the key type and enter any personal information, as shown in Figure 4-67.

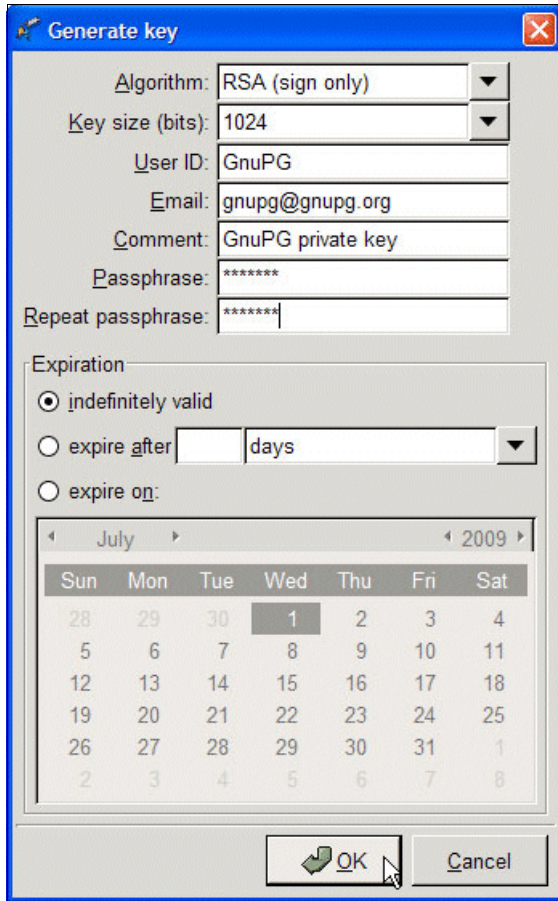


Figure 4-67 Generate a key window

Click **OK** to create the key. When this key is used to encrypt files, you are prompted for the passphrase in this window.

When created, the GNU Privacy Assistant can further show and process the generated RSA key, as shown in Figure 4-68.

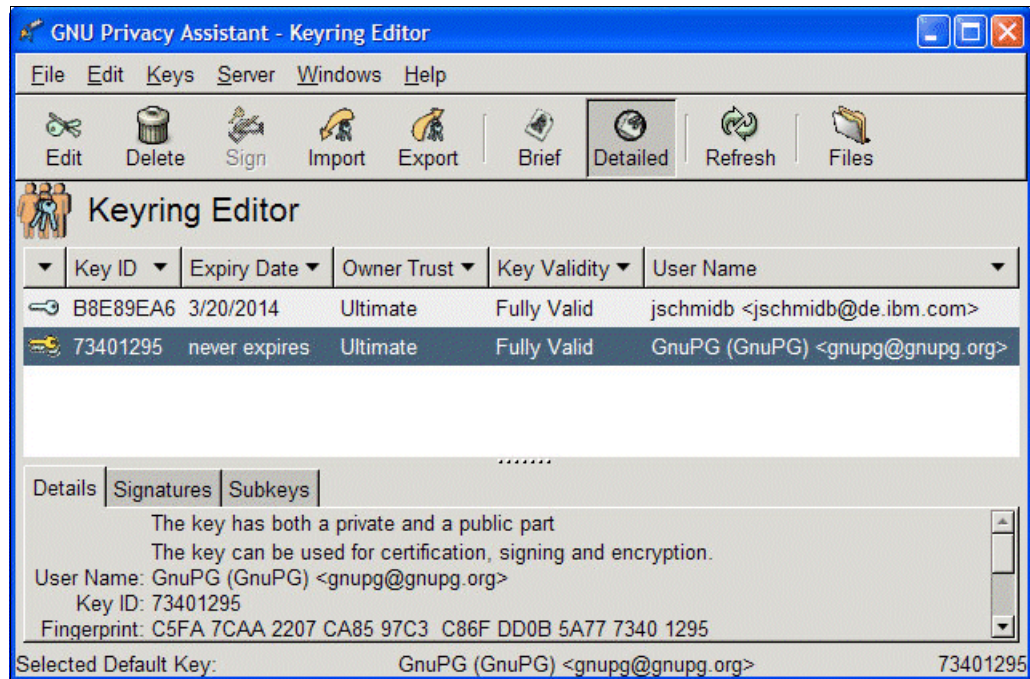


Figure 4-68 GNU Privacy Assistant main window

### Exporting the public key from the GnuPG keystore

Open the GUI and select **Keys** → **Export Keys**, as shown in Figure 4-69.

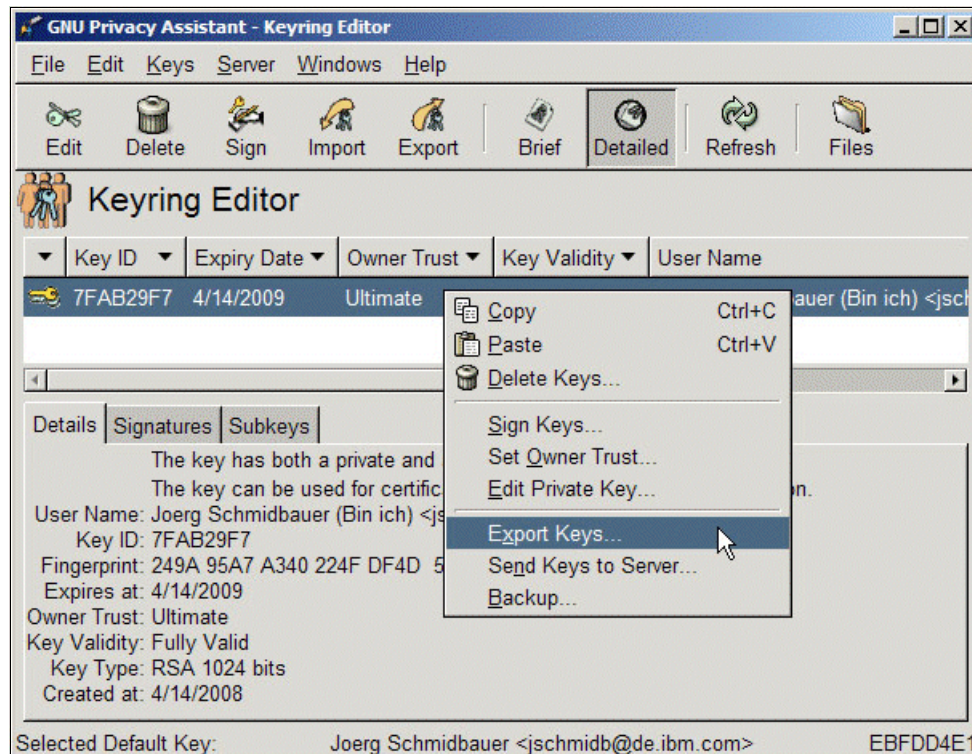


Figure 4-69 Exporting the key with GNU Privacy Assistant



Select the output file name and click **OK**, as shown in Figure 4-70.

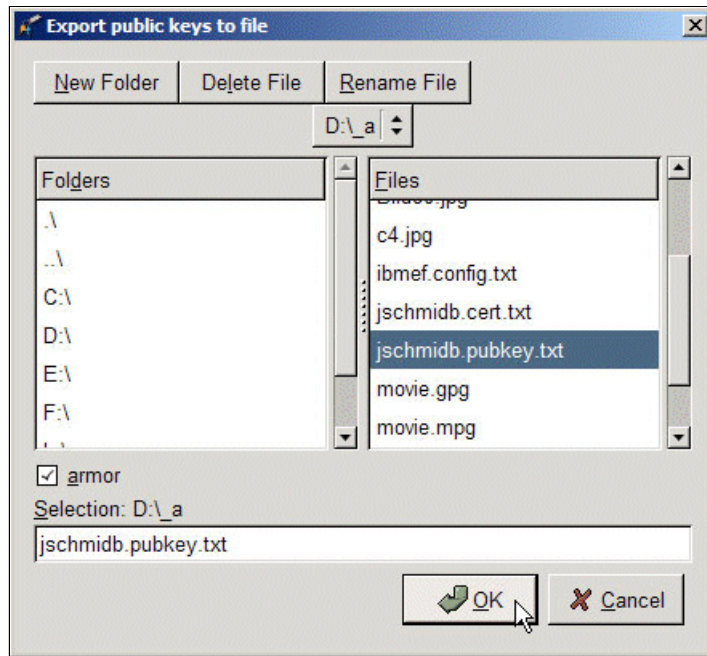


Figure 4-70 Export public keys to file window

The PGP public key is now contained in the `jschmidb.pubkey.txt` file. The content is similar to Example 4-46.

*Example 4-46 PGP public key block*

---

```
-----BEGIN PGP PUBLIC KEY BLOCK-----  
Version: GnuPG v1.4.7 (MingW32)  
  
mIOESANgbAEEAL4PtZXMRI/ncYePt/BXm/8UCgzhuMQyDjbeONvE5XhhYBKiyfe0  
va14c/yYHfs1DPf5UcX1V1KiqaK9xsuq0zCs7VRb8AWtKTDEzEzbhwPmqixvhHLh  
/ImPtZlqBPMI8vkqZtvL9MnSfv9C9tju9gC/LJUXYJqqrJSkMG5RMm9vABEBAAG0  
MUUpvZXJnIFNjaG1pZGJhdWVyICdCaW4gaWNoKSA8anNjaG1pZGJAZGUuaWJtLmNv  
bT6IvAQTAQIAJgUCSANGbAIbAwUJAeEzgAYLCQgHAwIEFQIIAwQWAgMBAh4BAheA  
AAoJEPo+Urr/qyn3NhMD/i84nWz42vqDfXv62ztgr6PTuVZf0duFqyC3vzbtfsJR  
048uhIx25q0igBjiUjhKSsP1Uk6HUr2A75+coJhq+FQDmKmm+2a+9+zx1Br3hZaX  
vmSvwQZfNNejDfimbicvFDbr/dzNzyPFzkHqTetQSwKc3ibK662npdrqX0Dh2u15  
=kmid  
-----END PGP PUBLIC KEY BLOCK-----
```

---

## Importing the PGP public key into Keyman/VSE

Start the Keyman/VSE tool and select **File** → **Import PGP public key from file**, as shown in Figure 4-71.

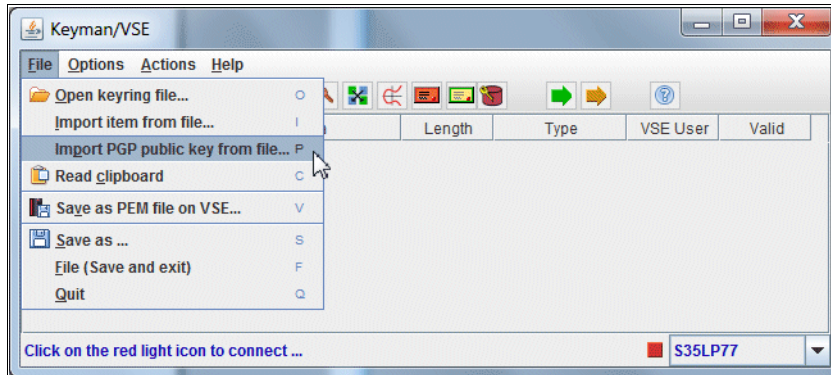


Figure 4-71 Import the PGP public key from the file with Keyman/VSE

Figure 4-72 shows the previously exported file.

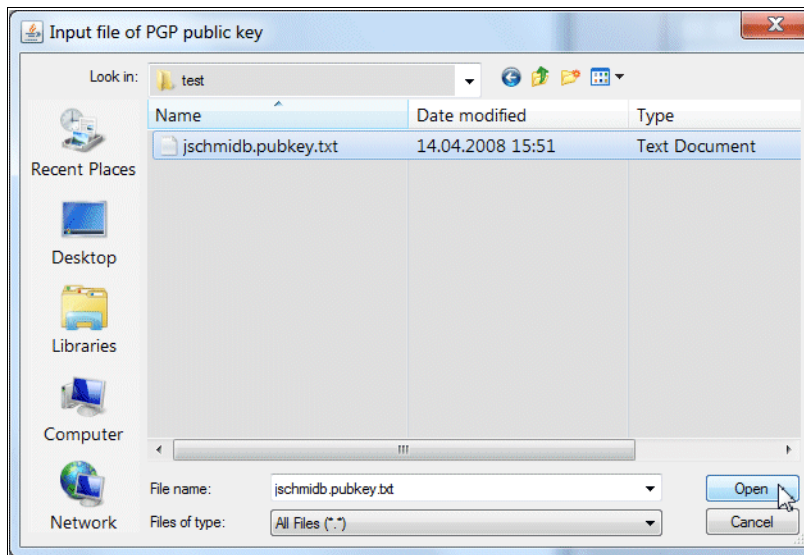


Figure 4-72 Open the file dialog in Keyman/VSE

Select your file and then, click **Open**. The window that is shown in Figure 4-73 opens.



Figure 4-73 Import the PGP public key

Click **Continue** to open the next window, as shown in Figure 4-74.

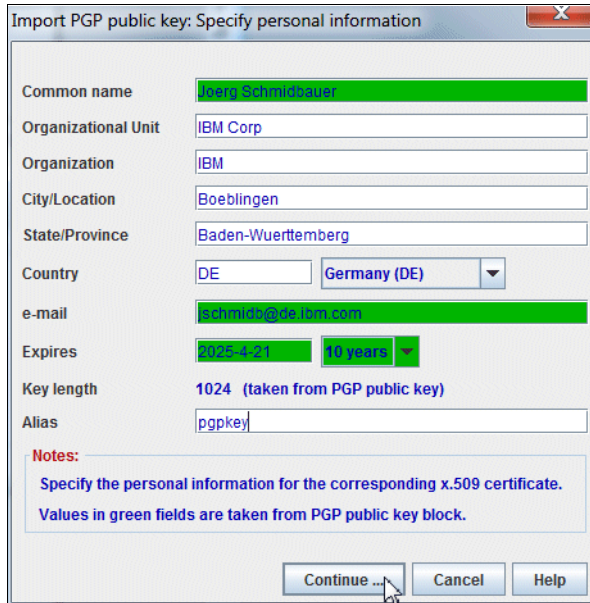


Figure 4-74 Specify the personal information dialog

Enter your personal information and click **Continue**.

The PGP public key is now imported in Keyman/VSE and can be uploaded to z/VSE.

## Uploading PGP public key to z/VSE

Right-click the imported PGP public key and select **Upload to VSE**, as shown in Figure 4-75.

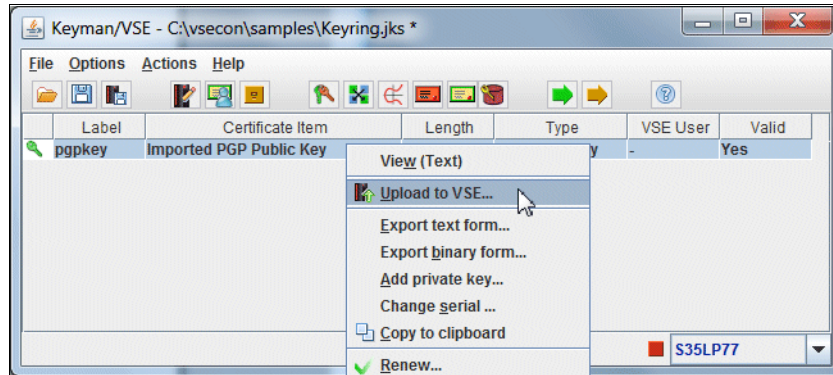


Figure 4-75 Upload the key dialog

Because only a public key is available, select member type **CERT**, and click **Upload**, as shown in Figure 4-76.

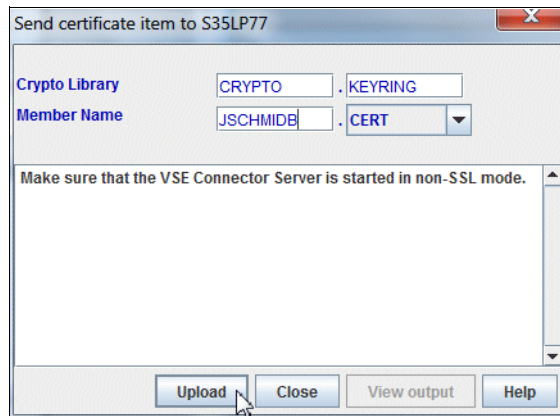


Figure 4-76 Sending the certificate item to the VSE dialog

The PGP public key is now stored in z/VSE library member JSCHMIDB.CERT. We can now encrypt a z/VSE data set by using public key encryption with this public key.

## Encrypting on VSE and decrypting on the workstation

The JCL that is shown in Example 4-47 encrypts a data set with public key encryption by using the previously uploaded public key.

*Example 4-47 Public key encryption on z/VSE*

```
* $$ JOB JNM=EFPGP,CLASS=0,DISP=D
* $$ LST DISP=D,CLASS=A,RBS=100
// JOB EFPGP TEST OPENPGP SUPPORT
// LIBDEF *,SEARCH=(PRD2.SCEEBASE,PRD2.PROD,PRD2.DBASE)
// EXEC IJBEPGP
PK_ENCRYPT
RECIPIENT_ALIAS=CRYPTO.KEYRING(JSCHMIDB)
S2K_CIPHER_NAME=AES_128
CLRFILE=DD:CLRDATA
ENCFILE=DD:ENCDATA
/*
```



/&  
\$\$ E0J

After downloading the encrypted data set in binary to a Microsoft Windows PC, you can decrypt it by using the GPGee tool, as shown in Figure 4-77.

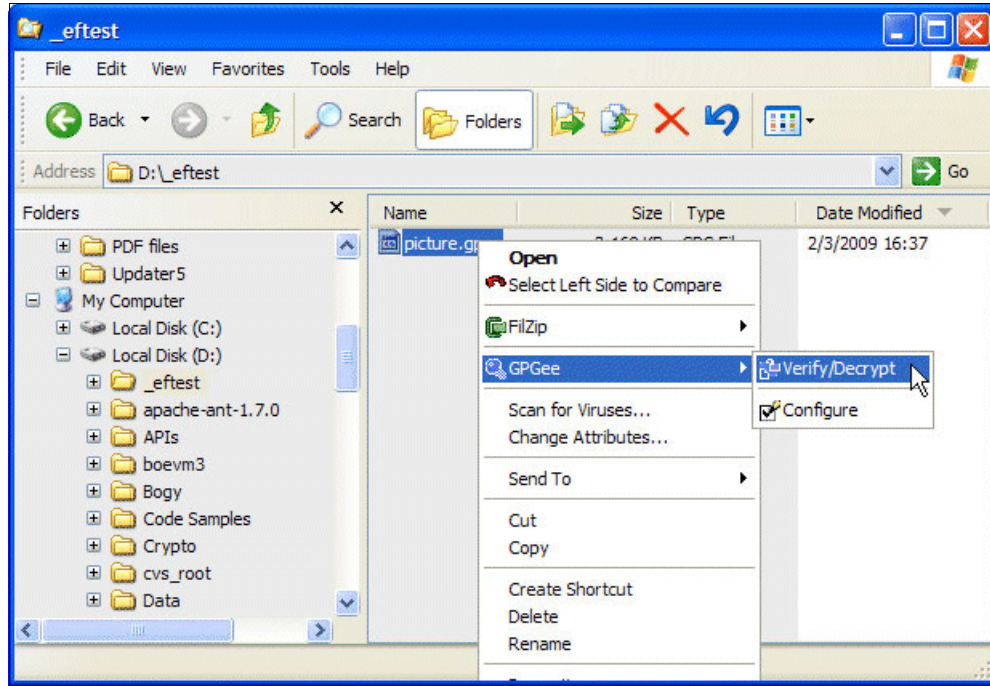


Figure 4-77 Verify or decrypt a file with GPGee

As shown in Figure 4-78, you are prompted for the private key's password (passphrase). You specified this password when the RSA key is created. For more information, see “Creating an RSA key pair by using GnuPG” on page 175.

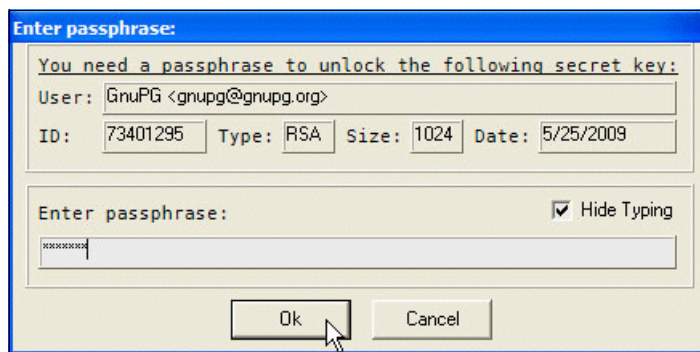


Figure 4-78 Enter passphrase dialog for public key decryption

Enter the password (passphrase) and click **OK**.



Figure 4-79 shows the GPGe window after a file is successfully decrypted.

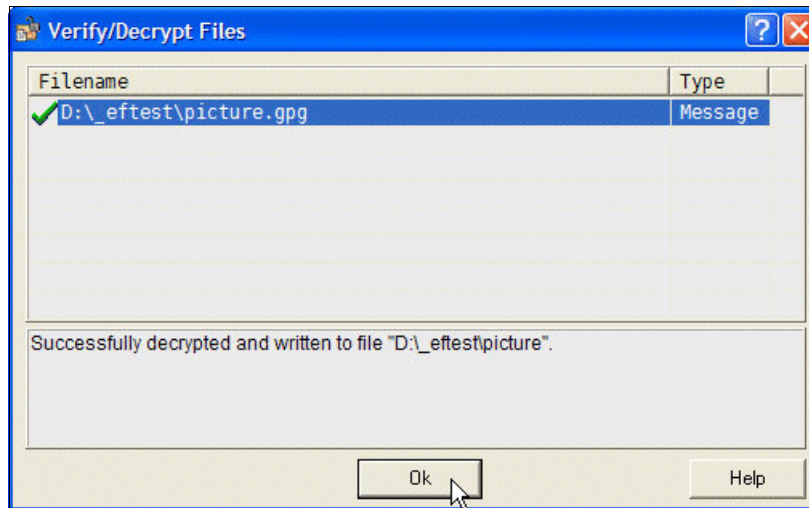


Figure 4-79 Verify/decrypt files window

### Encrypting for multiple recipients

You can specify multiple RECIPIENT\_ALIAS parameters in the same job to allow multiple recipients to decrypt the encrypted data set. This behavior requires having the public key of each recipient available on z/VSE in a PRVK or CERT member. As of this writing, the maximum number of RECIPIENT\_ALIAS parameters is 16.

Example 4-48 shows a job that uses multiple recipients.

Example 4-48 Encryption on z/VSE with multiple recipients

---

```

* $$ JOB JNM=EFPGP,CLASS=S,DISP=D
* $$ LST DISP=D,CLASS=A,RBS=100
// JOB EFPGP ENCRYPT
// LIBDEF *,SEARCH=(PRD2.SCEEBASE,PRD2.PROD,PRD2.DBASE)
// EXEC IJBEPGP
PK_ENCRYPT
RECIPIENT_ALIAS=CRYPTO.KEYRING(BOBSKEY)
RECIPIENT_ALIAS=CRYPTO.KEYRING(JIMSKEY)
RECIPIENT_ALIAS=CRYPTO.KEYRING(RODSKEY)
RECIPIENT_ALIAS=CRYPTO.KEYRING(ALICEKEY)
RECIPIENT_ALIAS=CRYPTO.KEYRING(MYKEY)
S2K_CIPHER_NAME=AES_128
COMPRESSION=1
COMPRESS_NAME=ZIP
DIGEST_NAME=SHA_1
CLRFILE=DD:FILE1
ENCFILE=DD:FILE2
/*
/&
* $$ EOJ

```

---

## Encrypting on a workstation and decrypting on z/VSE

Use the GPGe tool to encrypt a local file. Figure 4-80 shows how to use the GPGe tool for encrypting a file by using public key encryption.

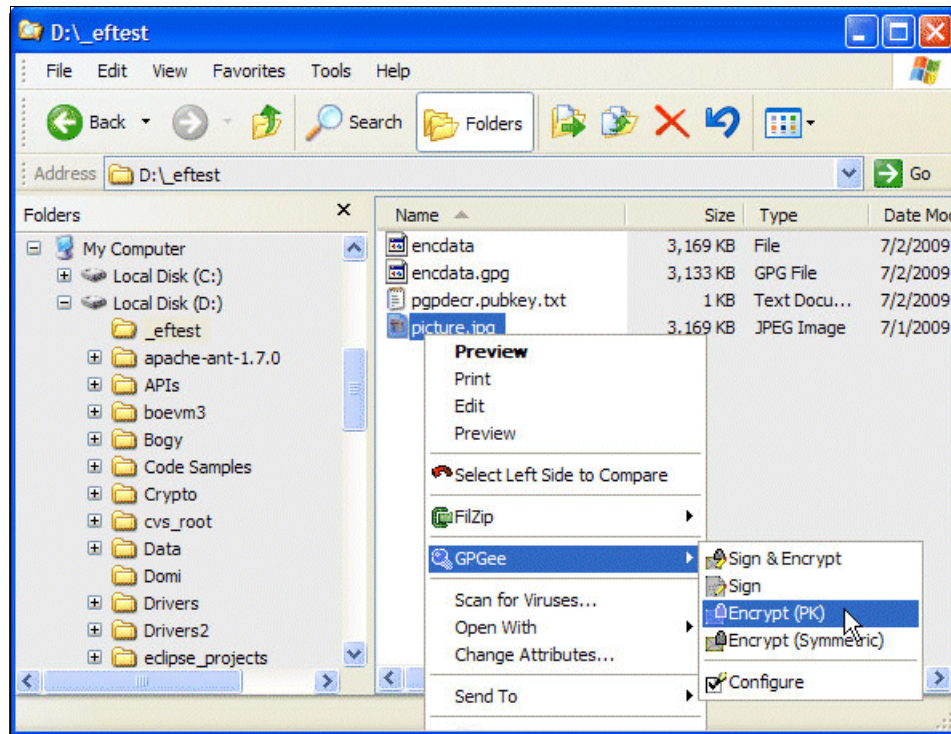


Figure 4-80 GPGe pop-up menu for public key encryption

As shown in Figure 4-81, select the key that is to be used. Click **OK**.

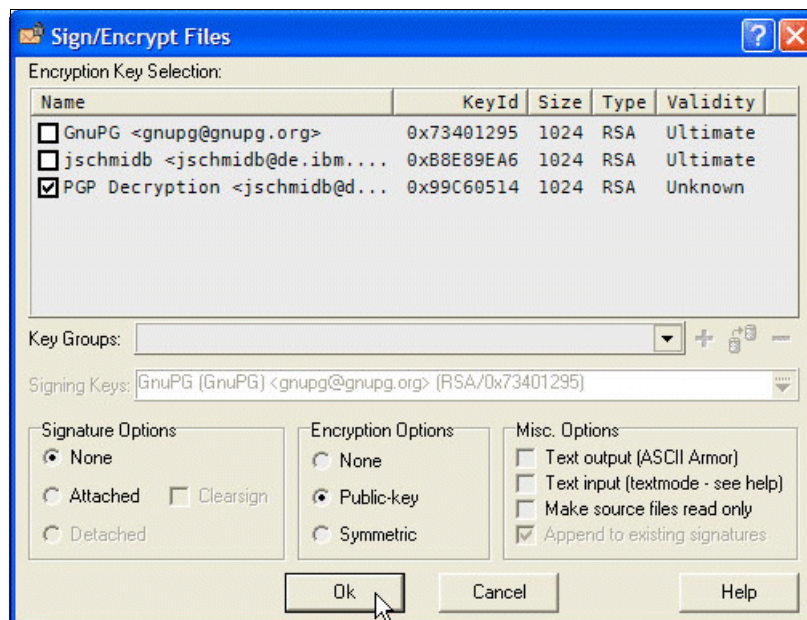


Figure 4-81 Sign/Encrypt Files dialog in GNU Privacy Assistant

After transferring the encrypted file to z/VSE, it can be decrypted on z/VSE with the JCL that is shown in Example 4-49.

Example 4-49 z/VSE JCL for decryption using a private key

```
* $$ JOB JNM=EFPGP,CLASS=0,DISP=D
* $$ LST DISP=D,CLASS=A,RBS=100
// JOB EFPGP TEST OPENPGP SUPPORT
// LIBDEF *,SEARCH=(PRD2.SCEEBASE,PRD2.PROD,PRD2.DBASE)
// EXEC IJBEPGP
DECRYPT
RECIPIENT_ALIAS=CRYPTO.KEYRING(PGPDECR)
CLRFILE=DD:CLRDATA
ENCFILE=DD:ENCDATA
/*
/&
* $$ EOJ
```

### Exchanging RSA keys with z/OS

Because Encryption Facility for z/OS is a Java application, it can work with Java keystores (JKS). You can create a JKS file with Keyman/VSE.

Start the Keyman/VSE tool and create an RSA key pair, as shown in Figure 4-82.

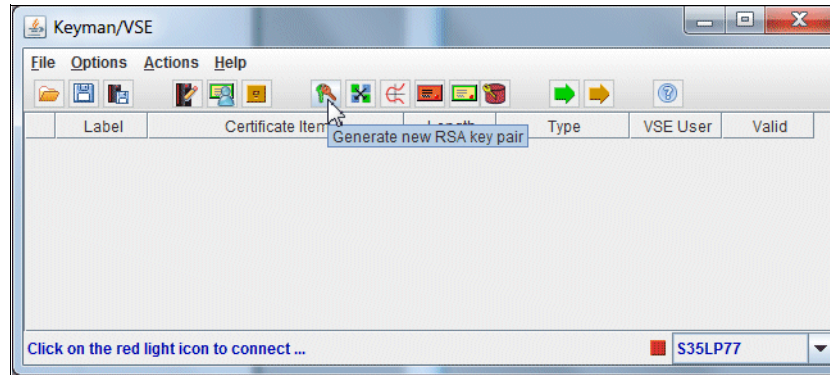


Figure 4-82 Creating an RSA key pair in Keyman/VSE

After the key is created, click the Save button, as shown in Figure 4-83.

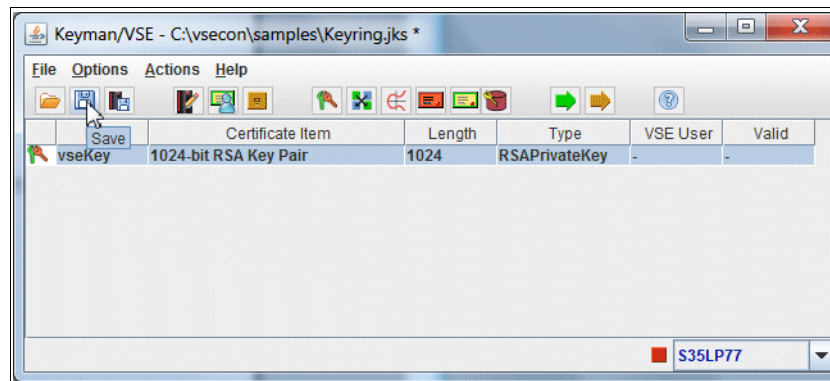


Figure 4-83 Created RSA key in Keyman/VSE

In the Save local keyring file window, change the file type to JKS and specify the keystore password and file name, and then, click **OK**, as shown in Figure 4-84.

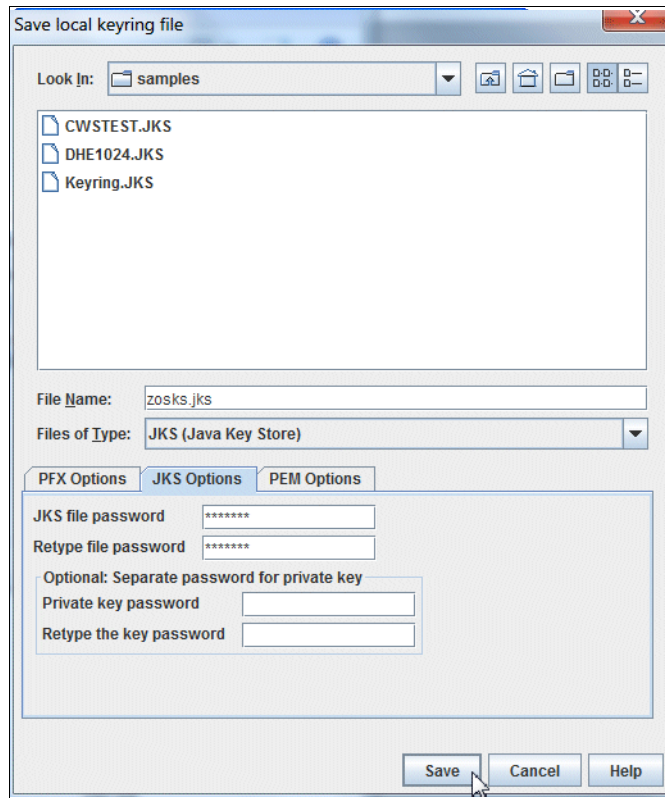


Figure 4-84 Save local keyring file window

After saving the file, close the Keyman/VSE window.

The next step is to transfer the JKS file to the z/OS system by using binary FTP.

## Encrypting on z/OS and decrypting on z/VSE

The shell script that is shown in Example 4-50 encrypts a file on z/OS by using public key encryption. We use the `zosks.jks` file that we created in the previous section. The `-rA` parameter references the key label of the RSA key in this keystore.

*Example 4-50 Command line for encryption on z/OS*

---

```

#!/bin/sh
#-----
# Encryption Facility for z/OS V1.2 PKE
#-----
export LIBPATH=$LIBPATH:/usr/lib/java_runtime
java -jar /usr/lpp/encryptionfacility/CSDEncryptionFacility.jar
  -o $1.gpg
  -rA vseKey
  -keystore zosks.JKS
  -keystore-type JKS
  -keystore-password mypasswd
  -key-password mypasswd
  -cipher-name AES_128
  -e $1

```

---



Running this script produces the messages that are shown in Example 4-51.

*Example 4-51 Script output*

```
RSTU:/SYSE/u/rstu>./pke.sh picture.jpg
CSD0768I Output data can be exchanged with the owner of the key with key ID
254ECD341E2FBE0C.
CSD0051I Command processing has completed successfully.
```

On the z/VSE side, keys are stored in PRVK members (private/public keys) or CERT members (public keys). Key IDs<sup>5</sup> are not maintained on z/VSE.

The output that is shown in Example 4-51 indicates that you can decrypt the encrypted file when your keystore contains a key with the specified key ID. On a workstation with GnuPG, this key ID is displayed for each key. On z/VSE, you must specify the name of the z/VSE library member to decrypt the file.

Before you can decrypt the file on z/VSE, you must transfer the private key to z/VSE. This process is done by using the Keyman/VSE tool by uploading the private key to z/VSE, as shown in Figure 4-85.

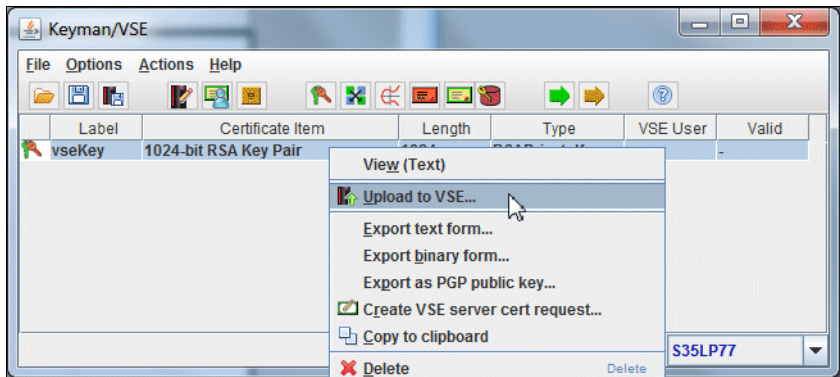


Figure 4-85 Upload RSA key to z/VSE in Keyman/VSE

In the next window (see Figure 4-86), specify a z/VSE library member name for the key. Ensure that the correct keyring library is displayed. Then, click **Upload**.

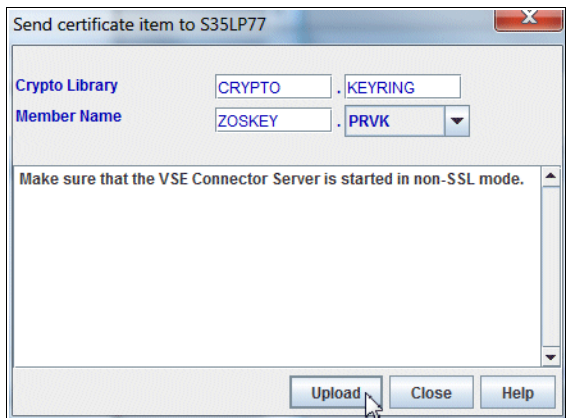


Figure 4-86 Send certificate Item to VSE

<sup>5</sup> Key IDs are 8-byte values that identify a key in a keystore. The key ID is calculated from the binary key data according to an algorithm that is described in RFC 2440/4880. Do not assume that Key IDs are always unique.

After transferring the encrypted file to z/VSE by using binary FTP, the JCL that is shown in Example 4-52 decrypts the file on z/VSE. It is assumed that the encrypted file was uploaded into a VSAM data set with file name ENCDATA.

*Example 4-52 z/VSE JCL for decrypting a file using a private key*

---

```
* $$ JOB JNM=EFPGP,CLASS=0,DISP=D
// JOB EFPGP TEST OPENPGP SUPPORT
// LIBDEF *,SEARCH=(PRD2.SCEEBASE,PRD2.PROD,PRD2.DBASE)
// EXEC IJBEPGP
DECRYPT
RECIPIENT_ALIAS=CRYPTO.KEYRING(ZOSKEY)
CLRFILE=DD:CLRDATA
ENCFILE=DD:ENCDATA
/*
/&
* $$ EOJ
```

---

## 4.6.9 Advanced encryption options

This section describes several variations of GnuPG parameters that we made during our tests. For a complete description of the GnuPG parameters, see the GnuPG documentation. In this section, we show our observations testing several of the parameters.

Certain GnuPG parameters are available only through the GnuPG CLI and not through the Gnu Privacy Assistant. This section primarily shows encryption options that are not available through the GnuPG GUI. Being familiar with the terms that are used in the RFC 2440/4880 also is helpful.

### Encrypt symmetric and with PK

The following command produces an encrypted file containing a PK-encrypted session key packet plus one SK-encrypted session key packet:

```
gpg --cipher-algo aes -e -c text.txt
```

The SK-encrypted session key packet uses the CAST5 algorithm regardless of what was specified in the command line. That is, the session key is encrypted with the key as specified by the S2K values by using the CAST5 algorithm. The PK-encrypted session key packet can be parsed by Encryption Facility for z/VSE, but the SK-encrypted session key packet cannot be read by z/VSE because CAST5 is not supported by z/VSE.

The following cipher algorithms can be used for GnuPG: 3des, aes (=aes128), aes192, aes256, idea, cast5, blowfish, and twofish. For more information about a list of supported ciphers on z/VSE, see Table 4-2 on page 99.

**Note:** Although RFC2440 did accept single-key DES (algorithm ID = 6), it was removed in RFC4880.

On z/VSE, PBE and PKE are mutually exclusive. Only one of the two commands should be specified in the JCL.

## Encrypting without compression

The following commands produce an encrypted file that contains a PK, encrypted session key packet followed by a tag 18 packet (literal data is not compressed):

```
gpg --compress-algo uncompressed --cipher-algo 3des -e text.txt
gpg --encrypt --compress-algo uncompressed --cipher-algo aes text.txt
```

The following compression algorithms can be used for Encryption Facility:

- ▶ Uncompressed
- ▶ Zip
- ▶ Zlib

BZip2 compression is not supported on z/VSE.

The following command specifies the compression level, but not the compression algorithm:

```
gpg --encrypt --cipher-algo aes --compress-level 0 text.txt
```

Valid compression levels are 0 - 9; three of the levels (0, 1, and 9) have special meanings:

- |          |                                   |
|----------|-----------------------------------|
| <b>0</b> | No compression                    |
| <b>1</b> | Best performance                  |
| ...      | (levels 2 - 8)                    |
| <b>9</b> | Best compression, not recommended |

The default value is 6, which is a compromise between speed and compression.

All commands produce the same encrypted file structure: a PK encrypted session key packet is followed by an SK encrypted integrity protected data packet (tag 18), which contains a literal data packet.

## No encryption and no compression

This approach is not possible with GnuPG nor with Encryption Facility for z/OS. You cannot specify *plaintext* as the cipher algorithm, although it is specified in RFC4880. Also, Encryption Facility for z/VSE does not allow the plaintext cipher.

## Encrypting without MDC

The following command produces an encrypted file that contains a PK-encrypted session key packet, followed by an SK-encrypted session key packet, followed by a Symmetrically Encrypted Data Packet (tag 9), which does not contain an MDC:

```
gpg --cipher-algo aes --disable-mdc -e -c text.txt
```

z/VSE always uses MDC for better data integrity, but accepts encrypted data sets without an MDC in a received PGP message.

## Encrypting without using a salt value

GNUPG and Encryption Facility for z/OS provide the `-s2k-mode` parameter to specify how the session key is created on the basis of a password. The following values are valid for `s2k-mode`:

- |          |  |
|----------|--|
| <b>0</b> | Plain PBE (not recommended)                                  |
| <b>1</b> | Salt-based PBE   |
| <b>3</b> | Salt-based PBE that is iterated (default on z/OS and z/VSE). |

For more information about the `-s2k-mode` parameter, see RFCs 2440/4880.

z/VSE always uses iterated and salted PBE (`s2k-mode = 3`), but accepts any valid the `-s2k-mode` parameter in a received PGP message.

GnuPG does not allow PBE without the use of a salt value to generate the session key, as shown in the following example:

```
gpg --cipher-algo aes --s2k-mode 0 -e -c text.txt
gpg: NOTE: simple S2K mode (0) is strongly discouraged
gpg: you cannot use --symmetric --encrypt with --s2k-mode 0
```

However, the following example is possible with PKE:

```
gpg --cipher-algo aes --s2k-mode 0 -e text.txt
gpg: NOTE: simple S2K mode (0) is strongly discouraged
You did not specify a user ID. (you may use "-r")
```

Encryption Facility for z/VSE issues a warning message when decrypting a session key packet without a salt value. In this case, the session key is probably insecure.

## Encrypting confidential data

The following command encrypts the “for your eyes only” data:

```
gpg --for-your-eyes-only --cipher-algo aes -e -c text.txt
```

This data means that the literal data packet contains the reserved string `_CONSOLE` as the file name. The OpenPGP standard recommends to not save such types of data to disk. Instead, data is written to the console only. On z/OS and z/VSE, this is the `CONFIDENTIAL` parameter.

The GnuPG GUI does not show any hint about confidential data. However, when the command mode is used, you can verify the following information:

```
gpg --verbose -d text.gpg
gpg: AES encrypted data
gpg: encrypted with 1 passphrase
gpg: NOTE: sender requested "for-your-eyes-only"
This is some sensitive clear text.
```

When decrypting confidential data on z/VSE, data is *not* displayed on the z/VSE console for obvious reasons. z/VSE does not differentiate the decrypting of confidential and non-confidential data. However, specifying the parameter `CONFIDENTIAL` when encrypting data does cause the Encryption Facility for z/VSE to use the filename “`_CONSOLE`” in the literal data packet, as described in RFCs 2440/4880.

## S2K count code

OpenPGP specifies an S2K count code, which is a one-byte number expanding to values up to 65536 according to a formula in RFCs 2440 and 4880. The expanded value is the size of the hash context that is used to generate the session key out of the password and salt value. The default value is 96 (hex 0x60), which expands to 65536. This information means that password and salt are concatenated multiple times up to a length of 65536 bytes. Then, the specified hash function (parameter `-s2k-digest-name`) is applied to create the first *n* bytes of the key where *n* is the number of output bytes of the hash function. When *n* is less than the wanted key length, the process is repeated as described in RFCs 2440/4880.

The S2K count code can be specified in GnuPG by using the `-s2k-count` parameter. z/VSE always uses the default value of 96 when PBE is used, but accepts any count code in an encrypted PGP message. Encryption Facility for z/OS also does not provide this parameter.



## 4.6.10 Observation

This section describes an observation we made during our tests.

### Failed to open output file

The symptom and reason are:

► Symptom:

When a z/VSE library member is used with a one-character member type, the following error occurs:

```
ERROR: FAILED TO OPEN OUTPUT FILE.  
      ERRNO=57 RC=10 FDBK=EB  
      MSG=EDC5057I The open mode string was invalid.
```

► Reason:

When a z/VSE library member is used as the output file for encryption, you must use a member type that consists of more than one character. Member types with only one character cannot be used for ENCFILE.

## 4.7 z/VSE Navigator GUI for Encryption Facility

The z/VSE Navigator provides a graphical interface for Encryption Facility for z/VSE.

The z/VSE Navigator function provides a limited functionality of Encryption Facility for z/VSE. The following file types are supported by the z/VSE Navigator interface:

- z/VSE library members
- VSAM files (ESDS, KSDS, RRDS)

It does not allow encrypting tapes and SAM data sets on disk or tape.

The interface requires Encryption Facility for z/VSE to be installed on z/VSE. The availability of phases IJBEFVSE and IJBEFPGP is selected when the window is opened.

To encrypt a z/VSE library member, right-click the member and select **Encrypt** in the z/VSE Navigator main window, as shown in Figure 4-87.

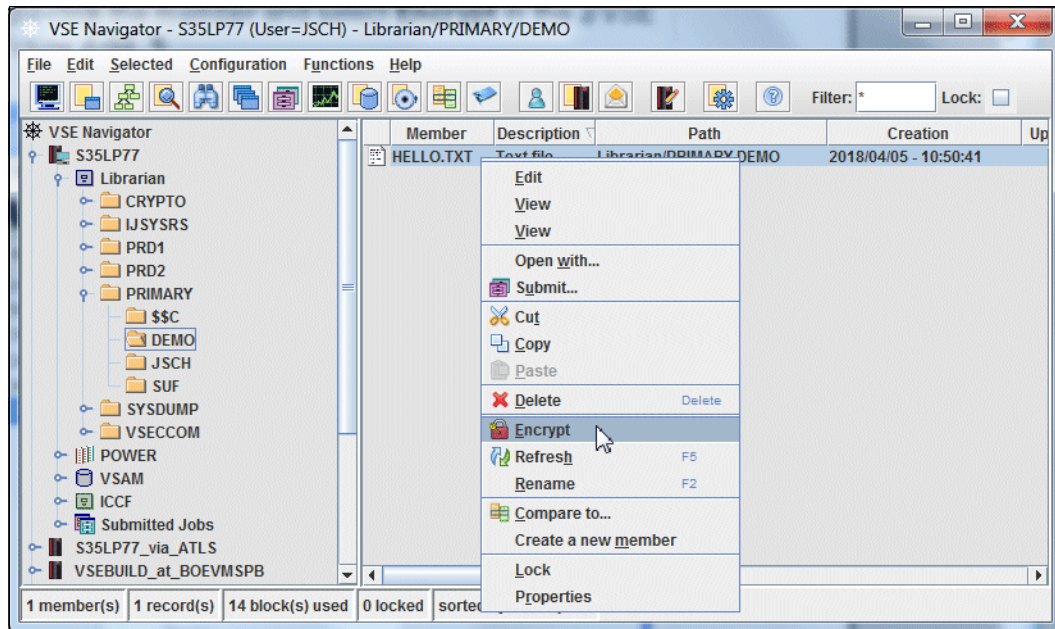


Figure 4-87 z/VSE Navigator main window

You must be familiar with the following rules for encrypting and decrypting:

- ▶ When encrypting a z/VSE library member, a new member that contains the encrypted data is created.
- ▶ Encrypted z/VSE library members get a member type of GPG after they are encrypted by IJBEFPGP utility. This file extension is also used by the GNU Privacy Guard Open Source tool, to which EF for z/VSE is compatible in terms of the encrypted data format.
- ▶ A member type of EFZ (Encryption Facility System z format) is used when encrypting with utility IJBEFVSE.
- ▶ Only members with these two member types show the menu choice Decrypt in their pop-up menu.
- ▶ Decrypted members receive a member type DECR. The GnuPG tool writes decrypted files to disk without a file extension.

- ▶ When encrypting VSAM files, the target cluster must exist and can be selected by clicking **Browse**, (see Figure 4-88). The same rule applies for decrypting an encrypted VSAM file. For more information about considerations when encrypting VSAM files, see *z/VSE Administration*, SC34-2692.

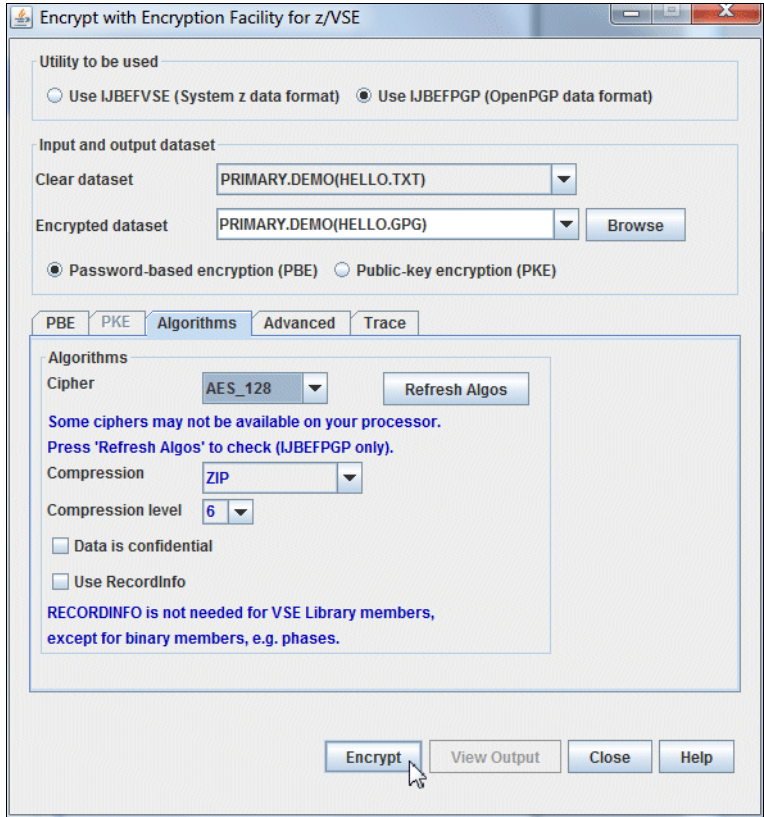


Figure 4-88 z/VSE Navigator encryption window

For more information about how to use this function, click **Help**.





## Secure Sockets Layer with z/VSE

In this chapter, we describe the setup of Secure Sockets Layer (SSL) in various scenarios in which z/VSE acts as server.

The SSL protocol provides security and data integrity for communications between servers and clients over TCP/IP networks. It uses the public key infrastructure (PKI) with RSA key pairs and digital certificates on the server and on the client side.

The examples in this chapter are based on the following software:

- ▶ z/VSE V4R2
- ▶ TCP/IP for VSE/ESA 1.5E as part of z/VSE 4.1
- ▶ VSE Connector Server as part of z/VSE 4.1 (job STARTVCS)
- ▶ Java 1.6.0\_05 from Sun Microsystems
- ▶ Keyman/VSE, update from 08/2007
- ▶ Mozilla Firefox version 2.0.0.6
- ▶ Microsoft Internet Explorer 6.0

This chapter includes the following topics:

- ▶ 5.1, “Generating the server key and certificates” on page 198
- ▶ 5.2, “SSL setup for Java-based connector” on page 205
- ▶ 5.3, “SSL setup for web browsers” on page 214
- ▶ 5.4, “Debugging SSL/TLS connections” on page 218

## 5.1 Generating the server key and certificates

For simplification, we do not purchase certificates from official certificate authorities (CAs), but create our own set of self-signed certificates. *Self-signed certificates* are not signed by an official CA; therefore, they work in a closed test environment only.

The easiest way to generate all necessary keys and certificates for the z/VSE server side is by using the Keyman/VSE utility. This tool is provided by IBM without warranty for no charge and can be downloaded from this web page:

<http://www.ibm.com/systems/z/os/zvse/downloads/>

Keyman/VSE is a Java application, which is typically installed on a personal computer (PC). It has the following prerequisites:

- ▶ Java 1.4 or higher on the workstation side
- ▶ VSE Connector Client on the workstation side
- ▶ TCP/IP for VSE/ESA 1.5E on the z/VSE side
- ▶ z/VSE Connector Server up and running in non-SSL mode on the z/VSE side

**Note:** For more information about the software that is used in our examples, see page 197.

Although Keyman/VSE provides many functions for manually creating keys and certificates, signing certificate requests, and so on, the easiest way to create the necessary files on z/VSE is by using the Wizard dialog for creating a self-signed keyring. For more information about the individual Keyman/VSE functions, see the HTML-based help of the Keyman tool.

The first step is to start Keyman/VSE and enter the properties of your z/VSE system. This information is required later for sending created keys and certificates to z/VSE.

### 5.1.1 Defining the properties of the z/VSE system

In the main window, click the **VSE Host properties** button, as shown in Figure 5-1.

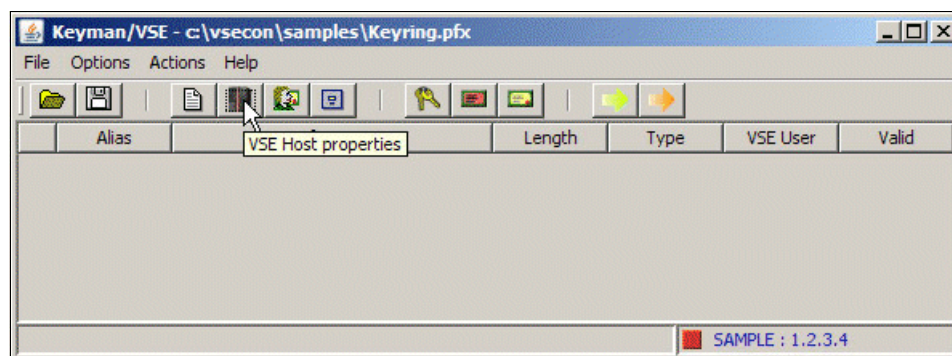


Figure 5-1 Select VSE Host properties

In the VSE Host - Properties window, enter the required information for your z/VSE system. First, click **New** to create a z/VSE host definition (see Figure 5-2).

Name	SAMPLE	New
IP Address		Add
Port	2893	Delete
VSE User	SYSA	Change
VSE Job Class	A	
VSE Password		
VSE Crypto Library	CRYPTO	KEYRING
Cert. Member Name	TEST01	PRVK / CERT / ROOT
Cert. Mapping Member	BSSDCUID	MAPPING
TCP/IP Library	PRD1	BASE
TCP/IP System ID	00	

OK Cancel Help

Figure 5-2 VSE Host-Properties dialog

Enter a unique name for your z/VSE system, its IP address, the port number of the z/VSE Connector Server, a z/VSE user ID with its password, and so on.

Then, as shown in Figure 5-3, click **Add** to add the definition. The Keyman/VSE is now ready to create the z/VSE server key and the necessary certificates.

Name	z9_LPAR4	New
IP Address	9.152.85.58	Add
Port	2893	Delete
VSE User	JSCH	Change
VSE Job Class	A	
VSE Password	*****	*****
VSE Crypto Library	CRYPTO	KEYRING
Cert. Member Name	SSL01	PRVK / CERT / ROOT
Cert. Mapping Member	BSSDCUID	MAPPING
TCP/IP Library	PRD1	BASE
TCP/IP System ID	00	

OK Cancel Help

Figure 5-3 VSE Host-Properties dialog with the new data



## 5.1.2 Creating the z/VSE key and certificates

Click the **Create self-signed keyring** button, as shown in Figure 5-4.

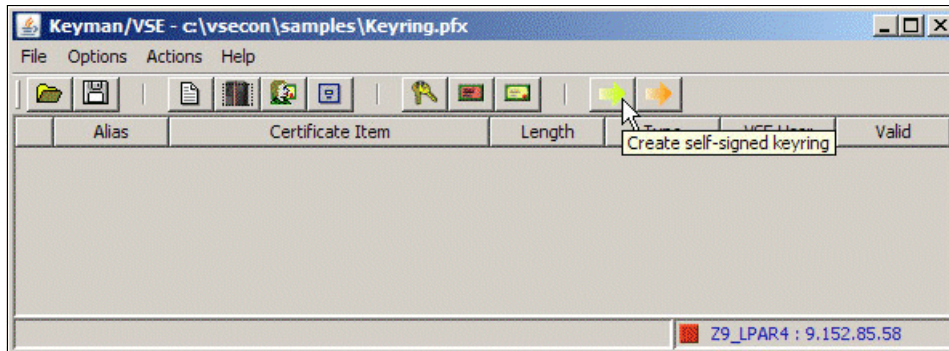


Figure 5-4 Select Create self-signed keyring

Complete the host properties window, as shown in Figure 5-5. Then, click **Next**.

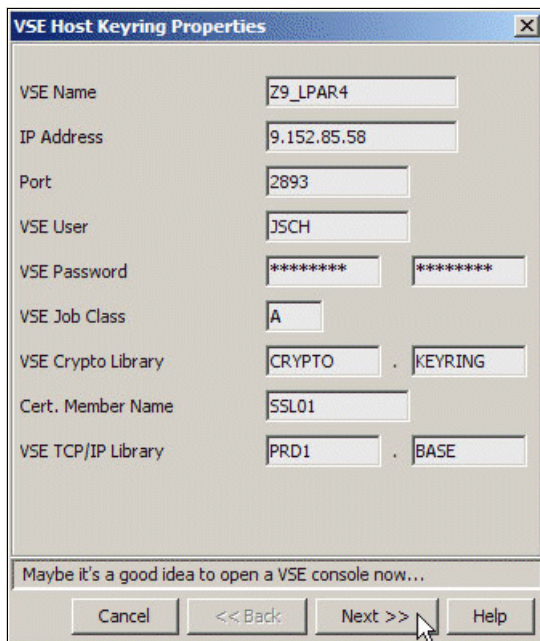


Figure 5-5 Specify keyring properties for z/VSE host



Figure 5-6 shows the window. Specify a password, which is used for protecting the local keyring file. Keep the **No encryption** settings for the encryption of public and private items. Otherwise, depending on your Java runtime, problems might occur when reading the file later.

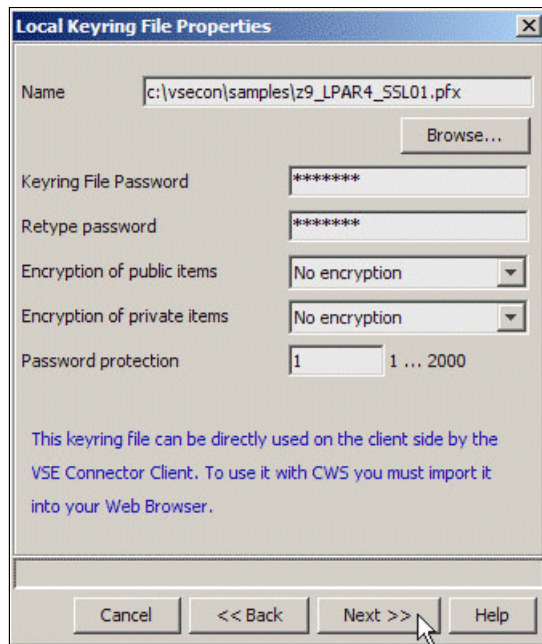


Figure 5-6 File properties of the local keyring

The local keyring file is stored in the PFX format, which is a standard format to store keys and certificates. PFX files can be imported into web browsers. Another format is Java KeyStore (JKS), which provides the same security, but is intended for use by Java applications.

After you specify the password, click **Next**.

Figure 5-7 on page 202 shows the next window in which you specify the key length of your server key and a unique alias string to identify the key. Listed are the available cipher suites with the selected RSA key length. This association was removed with TCP/IP fix ZP15E250 (APAR PK33472). For more information, see Table on page 212.

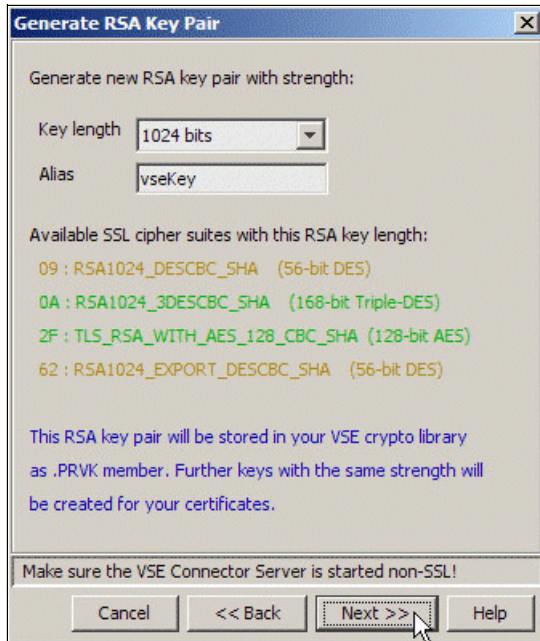


Figure 5-7 Generate the RSA key pair

Enter key length and alias. Click **Next**.

In the next window (see Figure 5-8), enter the personal information for the z/VSE root certificate. Click **Next**.

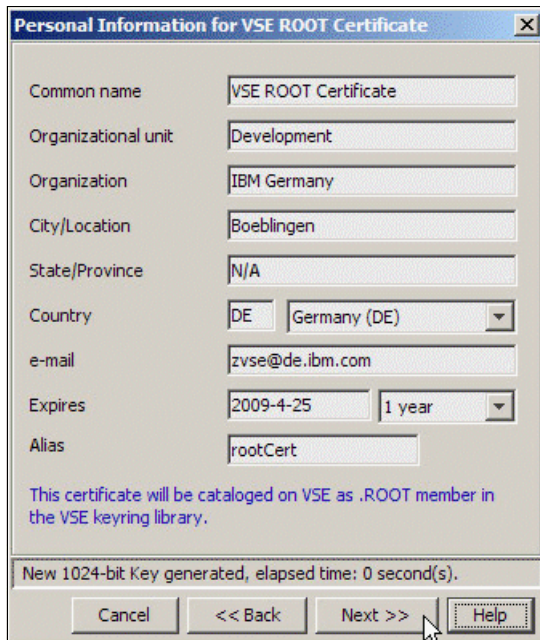


Figure 5-8 Personal information for the root certificate

In the next window, specify the personal information for the z/VSE server certificate, as shown in Figure 5-9. Then, click **Next**.

Personal Information for VSE Server Certificate

Common name: VSE Server Certificate

Organizational unit: Development

Organization: Your organization

City/Location: Your city/location

State/Province: Your state/province

Country: DE Germany (DE)

e-mail: info@your.company.com

Expires: 2009-4-25 1 year

Alias: vseCert

This certificate will be cataloged on VSE as .CERT member in the VSE keyring library.

New 1024-bit ROOT certificate generated.

Cancel << Back Next >> Help

Figure 5-9 Personal information for server certificate

In the next window (see Figure 5-10), enter the personal information for a client certificate. A client certificate is required only for client authentication.

Click **Next**.

Personal Information for VSE Client Certificate

Common name: VSE Client Certificate

Organizational unit: Your company

Organization: Your organization

City/Location: Your location

State/Province: Your state/province

Country: DE Germany (DE)

e-mail: vseclient@your.company.com

Expires: 2009-4-25 1 year

Map to VSE User: JSCH (Optional)

Alias: clientCert

New 1024-bit server certificate generated.

Cancel << Back Next >> Help

Figure 5-10 Personal information for client certificate

A summary of your input is shown in Figure 5-11.

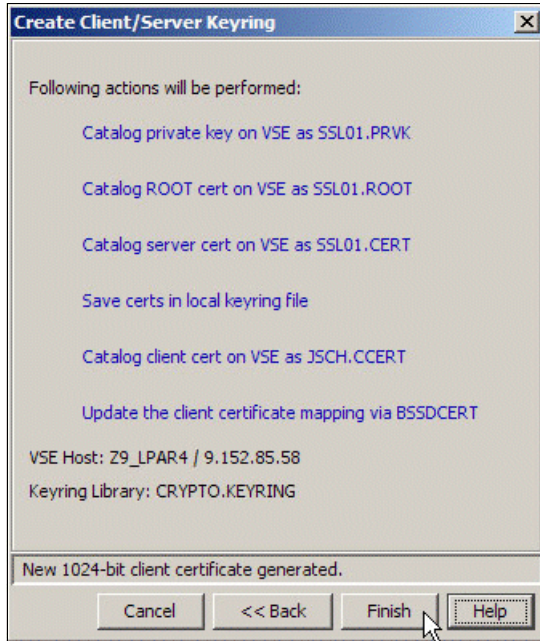


Figure 5-11 Summary of the actions to create client/server keyring

Review the information. Ensure that the z/VSE Connector Server is started in non-SSL mode on the z/VSE side. Then, click **Finish**.

The actions that were performed are shown in Figure 5-12.

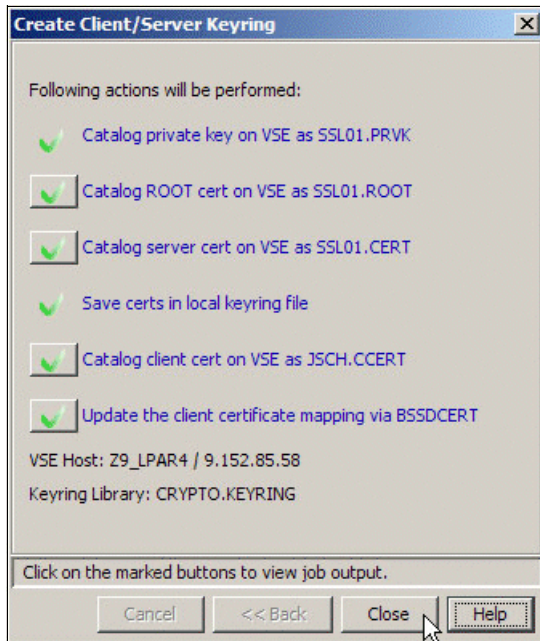


Figure 5-12 Actions successfully performed

You can close the Keyman/VSE tool. Because we do not require the server key on the client side, the key was not saved to the local file.



On the z/VSE side, you can control the contents of your keyring by using the LIBR command LD, as shown in Example 5-1.

*Example 5-1 List directory of CRYPTO.KEYRING*

```
LD SSL01.*

      DIRECTORY DISPLAY      SUBLIBRARY=CRYPTO.KEYRING      DATE: 2008-04-25
                                         TIME: 10:17
-----
 M E M B E R      CREATION   LAST      BYTES   LIBR CONT SVA  A- R-
 NAME            TYPE        DATE      UPDATE  RECORDS  BLKS STOR ELIG MODE
-----
SSL01      CERT      08-04-25  - -      724 B     1 YES  - - -
SSL01      PRVK      08-04-25  - -     2048 B     3 YES  - - -
SSL01      ROOT      08-04-25  - -      686 B     1 YES  - - -
L113I RETURN CODE OF LISTDIR IS 0
L001A ENTER COMMAND OR END
```

Three z/VSE library members are cataloged into CRYPTO.KEYRING. The CERT member contains the z/VSE server certificate, the PRVK member contains the RSA key pair, and the ROOT member contains the self-signed z/VSE ROOT certificate.

## 5.2 SSL setup for Java-based connector

The Java-based connector supports SSL connections from any Java program that is running outside of z/VSE to the z/VSE Connector Server that is running on z/VSE. For more information about setting up SSL for use with the Java-based connector, see *z/VSE e-business Connectors, User's Guide, SC34-2693*.

Java programs are available in one of the following formats:

- ▶ Java applications, such as the z/VSE Navigator program, which is provided by IBM and can be downloaded for no charge from the following z/VSE web page:  
<http://www.ibm.com/systems/z/os/zvse/downloads/>
- ▶ Java applets or servlets that are running in a Web Application Server environment, such as IBM WebSphere. For more information about how to set up SSL from a Java servlet that is running in WebSphere to the z/VSE Connector Server, see *WebSphere V5 for Linux on zSeries Connectivity Handbook, SG24-7042*.

The basic SSL setup for the Java-based connector with server authentication is described next.

**Note:** For more information about the software that is used in our examples, see page 197.

### 5.2.1 Setting up z/VSE Connector Server for SSL

On the z/VSE side, you must have the following ICCF members, which are provided in ICCF library 59, for setting up SSL for the z/VSE Connector Server:

- ▶ SKVCSCFG
- ▶ SKVCSSSL
- ▶ SKVCSCAT

## Setting up SKVCSCFG

SKVCSCFG is the template for the main configuration member of the z/VSE Connector Server. Here, you specify whether SSL is used.

**Note:** The z/VSE Connector Server can be started in SSL mode or non-SSL mode. If you want to have SSL and non-SSL clients connecting to z/VSE at the same time, start two instances of the z/VSE Connector Server in two different partitions, one providing SSL, the other providing non-SSL connections.

Change the SSLENABLE parameter to YES, as shown in Example 5-2.

*Example 5-2 SSL definition in main configuration member (SKVCSCFG)*

---

```
; *****  
; TCP/IP - SERVER SPECIFIC CONFIGURATIONS  
; - SERVERPORT : THE TCP PORT WHERE THE SERVER IS LISTENING  
; - MAXCLIENTS : THE MAXIMUM NUMBER OF CONCURRENT CLIENTS  
; - SSLENABLE : YES/NO - USE SECURE SOCKETS LAYER  
; *****  
  SERVERPORT = 2893  
  MAXCLIENTS = 256  
  SSLENABLE = YES
```

---

## Setting up SKVCSSSL

SKVCSSSL is the template for defining all SSL-related parameters. The most important parameter, CERTNAME, which changes with each setup, is the name of the z/VSE keyring, which consists of three z/VSE library members with file type PRVK (the private key file), ROOT (the root certificate), and CERT (the server certificate).

Change the CERTNAME parameter to the name you used when uploading the members to z/VSE, as shown in Example 5-3.

*Example 5-3 SSL settings in SSL configuration member SKVCSSSL*

---

```
SSLVERSION = SSL30  
KEYRING = CRYPTO.KEYRING  
CERTNAME = SSL01  
SESSIONTIMEOUT = 86440  
AUTHENTICATION = SERVER
```

---

In the first step, we allow all available cipher suites to be used (see Example 5-4).

*Example 5-4 Cipher suites setting in SSL configuration member SKVCSSSL*

---

```
; *****  
; CIPHERSUITES SPECIFIES A LIST OF CIPHER SUITES THAT ARE ALLOWED  
; *****  
CIPHERSUITES = ; COMMA SEPARATED LIST OF NUMERIC VALUES  
              01, ; RSA512_NULL_MD5  
              02, ; RSA512_NULL_SHA  
              08, ; RSA512_DES40CBC_SHA  
              09, ; RSA1024_DESCBC_SHA  
              0A, ; RSA1024_3DESCBC_SHA  
              62, ; RSA1024_EXPORT_DESCBC_SHA  
              2F, ; TLS_RSA_WITH_AES_128_CBC_SHA
```

Now, you must catalog your modified configuration members in the z/VSE library, which are read by the VSE Connector Server at startup. Use the job template SKVCSCAT that is shown in Example 5-5.

*Example 5-5 Job to catalog configuration members*

---

```
* $$ JOB JNM=VCSCAT,DISP=D,CLASS=0
// JOB VCSCAT CATALOG VCS CONFIGURATION MEMBERS
// EXEC LIBR,PARM='MSHP'
ACCESS S=PRD2.CONFIG
CATALOG IESVCSRV.Z REPLACE=Y
* $$ SLI ICCF=(SKVCSCFG),LIB=(12)
/+
CATALOG IESSSLCF.Z REPLACE=Y
* $$ SLI ICCF=(SKVCSSL),LIB=(12)
/+
/*
/&
* $$ EOJ
```

---

Because you changed only the main configuration member and the SSL configuration member, you must catalog only these two members.

After submitting the job that is shown in Example 5-5, restart the z/VSE Connector Server (see Example 5-6).

*Example 5-6 Start of z/VSE Connector Server*

---

```
R1 0045 // JOB STARTVCS START UP VSE CONNECTOR SERVER
      DATE 04/25/2008, CLOCK 10/22/07
R1 0045 IESC1001I BEGINNING STARTUP OF VSE CONNECTOR SERVER
R1 0045 IESC1011I USING CONFIG FILE:          DD:PRD2.CONFIG(IESVCSRV.Z)
R1 0045 IESC1012I USING LIBRARIAN CONFIG FILE: DD:PRD2.CONFIG(IESLIBDF.Z)
R1 0045 IESC1013I USING USERS CONFIG FILE:    DD:PRD2.CONFIG(IESUSERS.Z)
R1 0045 IESC1014I USING PLUGIN CONFIG FILE:   DD:PRD2.CONFIG(IESPLGIN.Z)
R1 0045 IESC1060I USING SSL CONFIG FILE:      DD:PRD2.CONFIG(IESSSLCF.Z)
R1 0045 T045: SSL100I IPCRYPTO 01.05 E 10/10/06 11.26 006DC000 806E0000 00620E
R1 0045 IESC1018I LOADING PLUGIN:  IESSAPLG
R1 0045 IESC1018I LOADING PLUGIN:  IESHTOHP
R1 0045 IESC1018I LOADING PLUGIN:  IESCOMPH
R1 0045 IESC1018I LOADING PLUGIN:  IESVSAPL
R1 0045 IESC1018I LOADING PLUGIN:  IESDLIPL
R1 0045 BSD100I IPNRBSDC 01.05 E 10/10/06 07.12 0071D800 05843C00 0583CB80
R1 0045 IESC1002I FINISHED STARTUP OF VSE CONNECTOR SERVER
R1 0045 IESC1003I WAITING FOR CONNECTIONS OF CLIENTS...
```

---

The SSL100I message that is shown in Example 5-6 indicates that the server is now in SSL mode. To verify this mode, you can also use the server's **status** command, as shown in Example 5-7.

*Example 5-7 Status of the z/VSE Connector Server*

---

```
msg r1,data=status
AR 0015 1I40I  READY
```

```

R1 0045 IESC1029I STATUS COMMAND
R1 0045     SERVER CONFIG FILE      = DD:PRD2.CONFIG(IESVCSR.V.Z)
R1 0045     CONFIGURATION INFORMATION:
R1 0045     MAX NUM. OF CLIENTS    = 256
R1 0045     TCP/IP SERVER PORT     = 2893
R1 0045     SSL ENABLED            = YES
R1 0045     SECURITY               = FULL
...

```

---

The server side is now ready. SSL can now be configured at the client.

## 5.2.2 Setting up z/VSE Navigator for SSL

This section describes how we use the z/VSE Navigator to support the z/VSE Connector client. In the example, we use the z/VSE Navigator as the SSL client.

The z/VSE Navigator includes the following prerequisites:

- ▶ Java 1.4 or higher on the workstation side
- ▶ TCP/IP for VSE/ESA 1.5E on the z/VSE side
- ▶ z/VSE Connector Client installed on the client side
- ▶ z/VSE Connector Server up and running in SSL mode on the z/VSE side

**Note:** For more information about the software that was used in our examples, see page 197.

In the z/VSE Navigator main window, right-click the VSE host icon and select **Configure Host**, as shown in Figure 5-13.

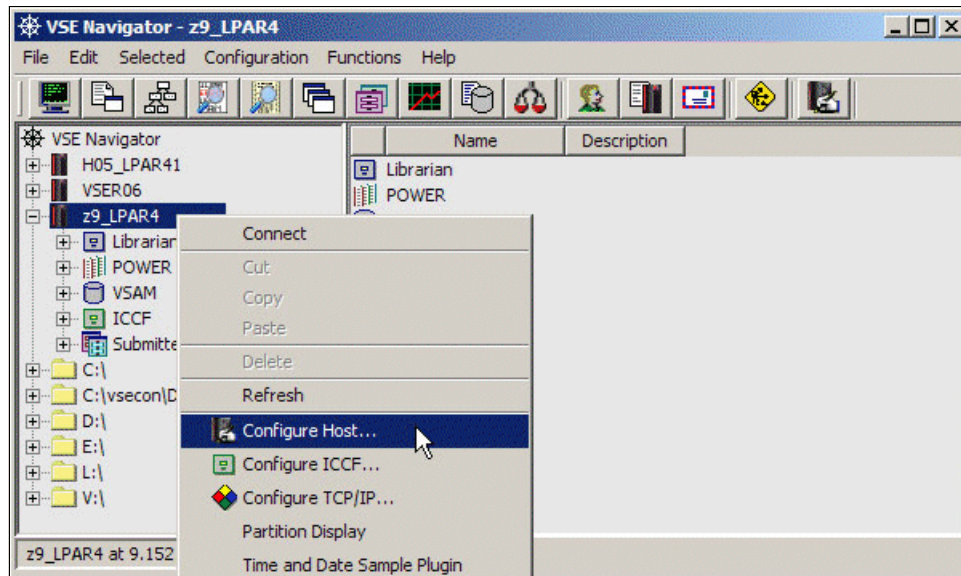


Figure 5-13 Select Configure Host



In the Configure Hosts window, select the **Use SSL connection** option, specify an SSL properties file to set up, and click **Edit**, as shown in Figure 5-14.

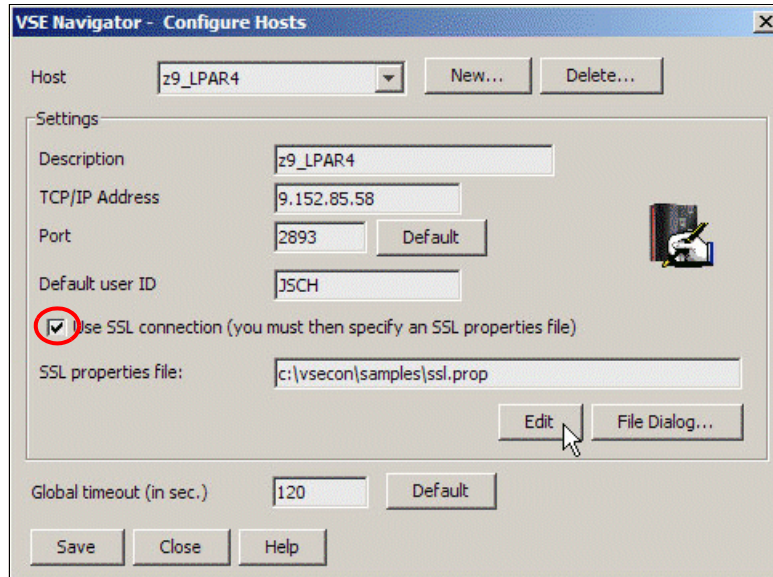


Figure 5-14 Configure host to use SSL

Verify the settings in your SSL properties file. The SSL properties file is a plain text file, which contains various SSL settings, including the name and location of your local keyring file, its password, the level of SSL protocol, and whether you want to use server or client authentication.

**Note:** You might want to have a different SSL properties file for each of your z/VSE systems. For example, for naming PFX files, use the following naming convention for SSL properties files:

machine\_lpar\_keyringname.prop

For example: z9\_LPAR4\_SSL01.prop

In Example 5-8, z/VSE Navigator displays the SSL properties file with the editor program that is defined as default viewer in the navigator configuration.

*Example 5-8 SSL properties file*

```
#-----
# SSL properties file for VSE Connector Client
# Keyring file:
#-----
KEYRINGFILE=c:\\vsecon\\samples\\z9_LPAR4_SSL01.pfx
#-----
# Keyring password:
#-----
KEYRINGPWD=ssltest
#-----
# SSL Version:   SSL (SSL 2.0/3.0) or
#                TLS (TLS 1.0)
#-----
SSLVERSION=SSL
```

```

#-----
# Available cipher suites with VSE:
#-----
CIPHERSUITES=TLS_RSA_WITH_AES_128_CBC_SHA,SSL_RSA_WITH_3DES_EDE_CBC_SHA,SSL_RSA_WI
TH_DES_CBC_SHA,SSL_RSA_WITH_NULL_SHA,SSL_RSA_WITH_NULL_MD5,SSL_RSA_EXPORT_WITH_DES
40_CBC_SHA
#-----
# Logon with the client's certificate: YES
#                               NO
#-----
LOGONWITHCERT=NO

```

---

In this setup, you include all available SSL cipher suites. These suites must be written in one line in the file and with no line breaks. As you first set up for SSL server authentication, you leave the following parameter setting:

```
LOGONWITHCERT=NO
```

### 5.2.3 Connecting to z/VSE by using SSL server authentication

You are now ready to connect to z/VSE.

Right-click the VSE host icon and select **Connect** in the z/VSE Navigator main window, as shown in Figure 5-15.

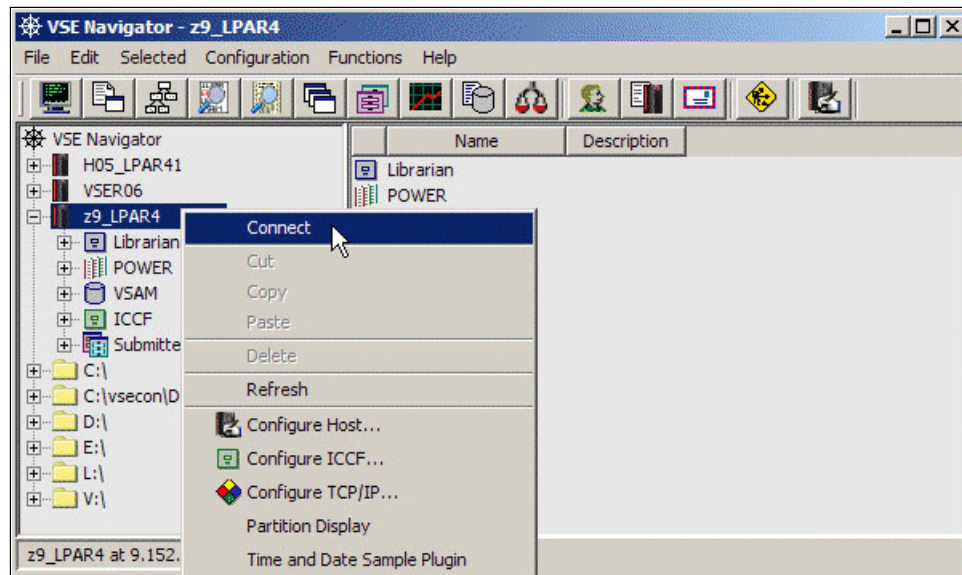


Figure 5-15 Select Connect

Figure 5-16 shows the logon window in which you enter your VSE user ID and password.

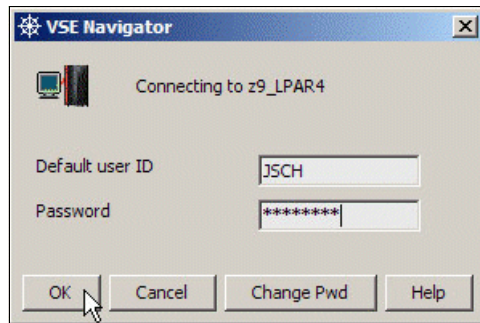


Figure 5-16 Logon window

During the connection process, you can view more information about the current SSL session by selecting **Display Server Certificate**, as shown in Figure 5-17.

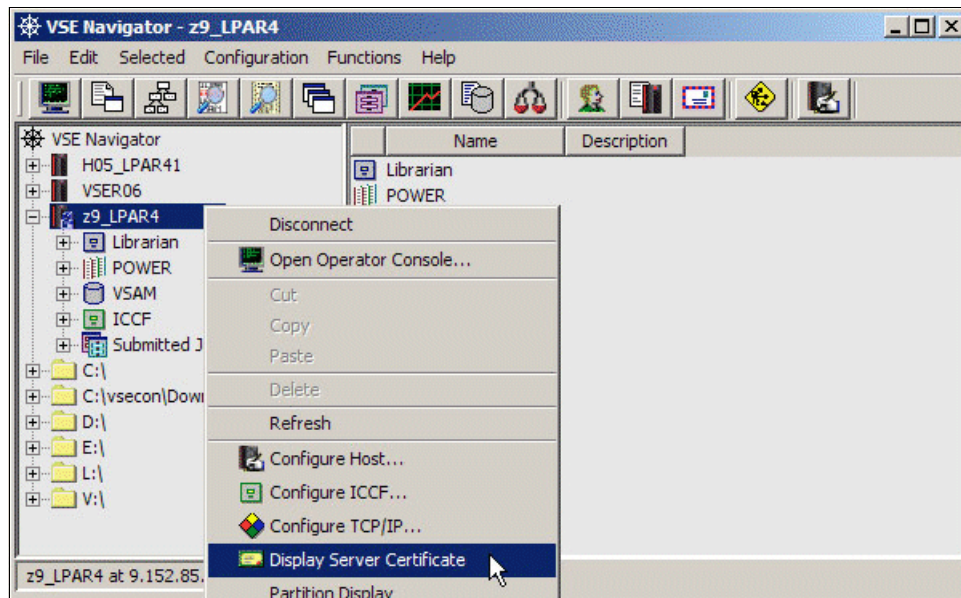


Figure 5-17 Select Display Server Certificate

The z/VSE server certificate box that is shown in Figure 5-18 on page 212 shows that you are connected by using a 1024-bit RSA key length. The used cipher suite indicates that you are using AES-128 for encrypting data that is sent over the line.

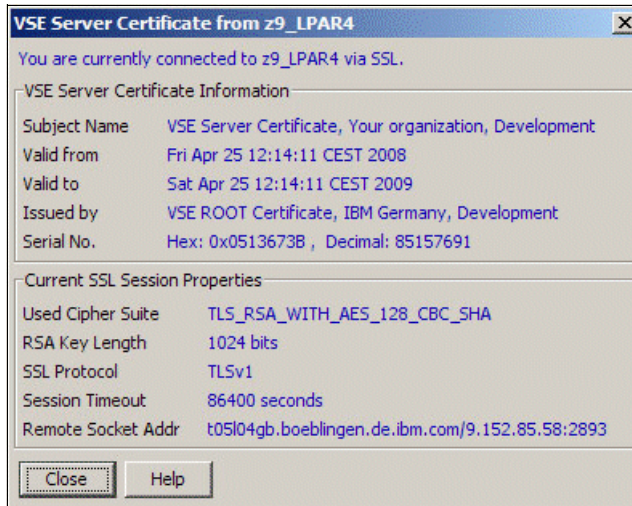


Figure 5-18 z/VSE server certificate

For a list of supported cipher suits and their meaning, see Table 4-2 on page 99 in chapter 4.

**Notes:** Consider the following points:

- ▶ When 2048-bit keys are used, you must have a Crypto Express2 or PCI-X Cryptographic Coprocessor card.
- ▶ AES support was introduced with TCP/IP fix ZP15E214 (APAR PK33472).
- ▶ AES-128 is available as hardware function on IBM System z9<sup>®</sup> processors and is much faster than the software implementations that are provided by TCP/IP. It is used transparently by TCP/IP when available.
- ▶ AES-256 is available as hardware function on IBM System z10<sup>®</sup> processors.

## 5.2.4 Considerations with client authentication

Because of restrictions that are imposed by Java runtime environments, client authentication is limited with the Java-based connector. In fact, when we initially shipped the z/VSE connectors based on Java 1.3, everything worked fine because we used a separate Java class library from IBM, which provided SSL functions.

With Java 1.4, SSL was included in Java, but since then, Java encountered a problem handling PFX files. Getting client authentication to work was possible in the test environment only with Java 1.4.2 from IBM. A solution is to use Java Keystores (JKS) instead of PFX files.

Although JKS was developed by Sun Microsystems and is therefore fully supported by Java, PFX is usually the format that can be imported in web browsers. Therefore, both are required, depending on the SSL client.

Because Keyman/VSE and the z/VSE Connector Client do not currently support JKS keystores, we did not perform client authentication with the Java-based connector.

## 5.2.5 Using encryption with AES-256

By default, your Java installation does not support AES with key sizes that are greater than 128 bits. However, you have unlimited strength cryptography to use AES-256.



Because of import-control restrictions that are imposed by some countries, the jurisdiction policy files that are shipped with Java permit only strong cryptography to be used. An unlimited strength version of these files (that is, with no restrictions on cryptographic strength) is available for download from the following web page:

<http://www.oracle.com/technetwork/java/javase/tech/index-jsp-136007.html>

Complete the following steps to activate unlimited strength cryptography in Java:

1. Replace the files `local_policy.jar` and `US_export_policy.jar` in the following directory of your Java installation, as shown in Figure 5-19:

...\lib\security

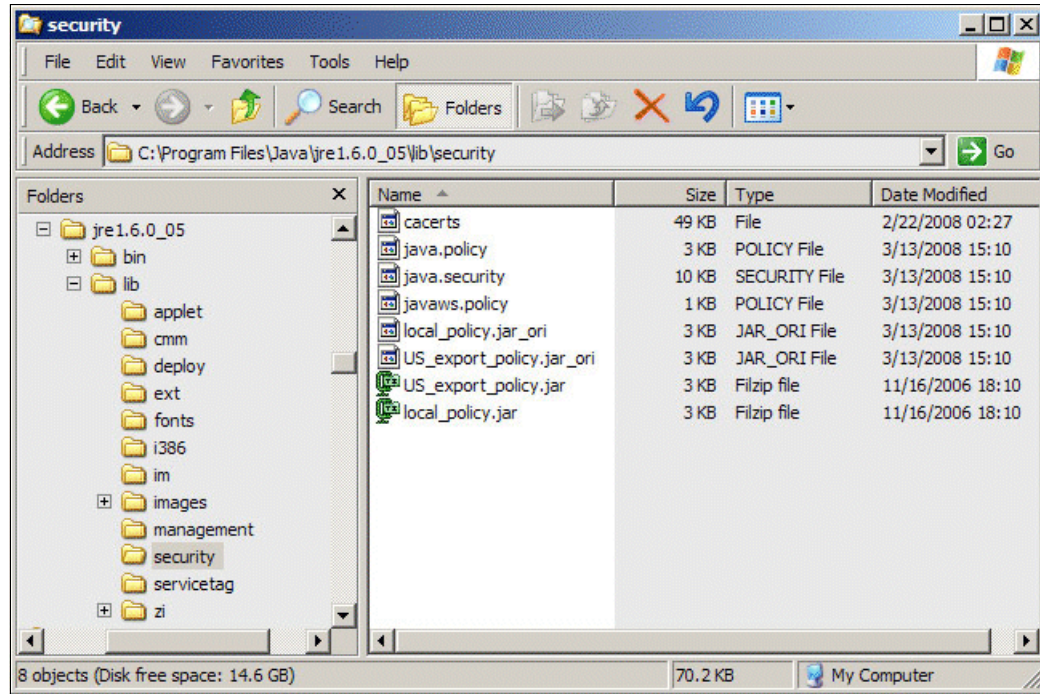


Figure 5-19 Policy files

2. Restart your Java application.

The same files can also be used to activate unlimited strength cryptography for an IBM Java.

When the z/VSE Navigator is used, specify the following cipher suite in the SSL properties file:

```
CIPHERSUITES=TLS_RSA_WITH_AES_256_CBC_SHA
```

On the host side, be sure that this cipher suite is contained in the list as specified in the SKVCSSSL as shown in Example 5-9.

*Example 5-9 Update SKVCSSSL to use TLS\_RSA\_WITH\_AES\_256\_CBC\_SHA*

---

```
; *****
; CIPHERSUITES SPECIFIES A LIST OF CIPHER SUITES THAT ARE ALLOWED
; *****
CIPHERSUITES = 35 ; TLS_RSA_WITH_AES_256_CBC_SHA
```

---

**Notes:** Consider the following points:

- ▶ AES-256 requires TCP/IP for VSE/ESA 1.5E with fix number ZP15E214 (APAR PK33472).
- ▶ When running on a z9 or earlier, AES-256 is performed in software.
- ▶ When running on a z10, AES-256 is provided with the CPU Assist feature (CPACF), which is used by TCP/IP transparently.

Restart the z/VSE Connector Server. The connection is now encrypted by using AES-256, as shown in Figure 5-20.

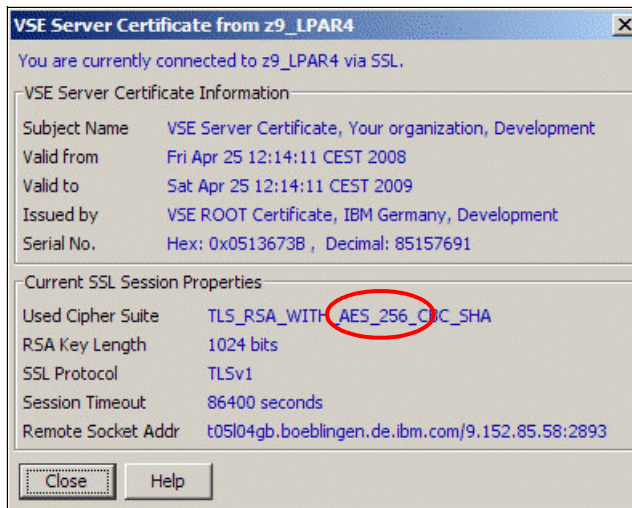


Figure 5-20 z/VSE server certificate

## 5.3 SSL setup for web browsers

This section describes the SSL setup for a web server environment that uses the HTTP and TLS daemons. For more information about the software that was used in our examples, see page 197.

Setting up SSL for the HTTP protocol is based on the SSL daemon that is provided by TCP/IP for VSE/ESA. The following modes are available:

- ▶ **SSL in native mode**  
SSL traffic goes directly to the SSL enabled daemon on z/VSE. Native mode is supported by the HTTP daemon only. In addition to the HTTP daemon, you must define a TLS daemon on the same port. The HTTP daemon then receives all SSL-related settings from the related TLS daemon.
- ▶ **SSL in pass-through mode**  
This mode must be used for secure Telnet, but can also be used for HTTP. Also, here we must define a TLS daemon. The difference in native mode is that we use the `PASSPORT` parameter to route SSL traffic from an unsecured daemon to the SSL daemon.

The preferred method is native mode. How to set it up is described next.

### 5.3.1 Setting up SSL native mode with HTTPD

In this section, we show an example for setting up native mode SSL with an HTTP daemon. The setup uses the z/VSE keyring members as described in 5.1.2, “Creating the z/VSE key and certificates” on page 200.

Add the definitions to your TCP/IP IPINIT member, as shown in Example 5-10.

*Example 5-10 Define TLS daemon*

---

DEFINE TLSD, ID=TLSD1,	ID of this SSL/TLS daemon
PORT=443,	Default HTTPS port
PASSPORT=443,	Native support
CIPHER=2F350A096208,	Allowed cipher suites
CERTLIB=CRYPTO,	Library name
CERTSUB=KEYRING,	Sublibrary name
CERTMEM=SSL01,	Member name
MINVERS=0300,	Minimum version required
TYPE=1,	SSL server authentication
DRIVER=SSLD	Driver phase name

---

In the definition, the PORT and the PASSPORT parameters are identical, which indicates native SSL mode. Also, the HTTP daemon must specify the same port number, as shown in Example 5-11.

*Example 5-11 Define HTTP daemon*

---

DEFINE HTTPD, ID=HTTPS,	ID of this HTTP daemon
PORT=443,	Default HTTPS port
COUNT=3,	Start 3 sessions
ROOT=PRIMARY.HTTPS,	Location of index.html
CONFINE=YES,	Confine to a specific lib
DRIVER=HTTPD	Driver phase name

---

**Note:** Defining multiple HTTP daemons to have multiple parallel sessions available is useful. In Example 5-11, we set the COUNT parameter to 3, which starts three internal HTTP daemons.

To display a simple welcome page, add an index.html file to the z/VSE sublibrary PRIMARY.HTTPS, Example 5-12. This sublibrary is the document root of the HTTP server.

*Example 5-12 Contents of file index.html*

---

```
<HTML>
<head>
<TITLE>Greetings from VSE</TITLE>
</head>
<body>

<h2>Hello world</h2>

<p>
</body>
</html>
```

---

The HTML statements are stored as `index.html` in `PRIMARY.HTTPS`. You can verify by using the LIBR command LD, as shown in Example 5-13.

*Example 5-13 List directory of sublibrary PRIMARY.HTTPS*

```
LD *.*

DIRECTORY DISPLAY      SUBLIBRARY=PRIMARY.HTTPS      DATE: 2008-04-25
                                                                    TIME: 11:24
-----
 M E M B E R          CREATION  LAST      BYTES    LIBR CONT SVA  A- R-
NAME      TYPE        DATE      UPDATE   RECORDS  BLKS STOR ELIG MODE
-----
INDEX     HTML      07-08-17  - -      4 R      1 YES  - - -
L113I RETURN CODE OF LISTDIR IS 0
LO01A ENTER COMMAND OR END
-----
```

Before trying to connect, see 5.3.2, “Considerations on \$WEB user” on page 216.

### 5.3.2 Considerations on \$WEB user

TCP/IP 1.5E requires the special user ID \$WEB to allow a web browser to connect to the HTTP daemon. This user can be defined in the IPINIT member, as shown in the following example:

```
DEFINE USER, ID=$WEB, PASSWORD=$WEB, WEB=YES
```

The user also can be defined as a z/VSE user ID so that the \$WEB user is known to the BSM. This is described in APAR PQ87041, which is the IBM APAR for TCP/IP service pack 1.5E. If the \$WEB user is not known, the following error occurs from the BSM:

```
F7 0100 BSST20I INVALID USER ID $WEB      IP ADDRESS = 9.152.216.58
F7 0098 IPN755I Fail File OpenRead $WEB 9.152.216.58 1288 PRIMARY
```

The APAR problem summary is available from the following Service and Support web page:

<http://www.ibm.com/systems/z/os/zvse/support/>

It states that the following special users must be defined in TCP/IP for the various IP protocols:

```
DEFINE USER, ID=$WEB, PASSWORD=$WEB, WEB=YES
DEFINE USER, ID=$LPR, PASSWORD=$LPR, LPR=YES
DEFINE USER, ID=$EVENT, PASSWORD=$EVENT, LPR=YES
DEFINE USER, ID=$LPD, PASSWORD=$LPD, LPD=YES
```



**Notes:** To avoid security exposures, consider the following points:

- ▶ As an example, when \$WEB is defined as an administrator, everything works, but the system is now open for anyone who knows about the \$WEB user and its publicly known password. Therefore, you must define \$WEB as normal user and restrict the access rights of \$WEB to a minimum with the BSM batch security. This configuration implies that you IPL your system with SYS SEC=YES to use the batch security. A good practice is to use this option.
- ▶ When \$WEB is defined in the IPINIT member only and not known to the BSM, you must turn off IP security so that it works. Otherwise, BSM rejects the file access when contacted by the HTTP daemon to access the index.html file because the user is unknown. However, turning off IP security because you want to run an HTTPD is likely not what you want.

### 5.3.3 Connecting to HTTPD by using a web browser

Because we are using the default port 443 for secure HTTPS connections, we can omit the port number in the URL.

When defining the z/VSE server certificate as described in 5.1.2, “Creating the z/VSE key and certificates” on page 200, we specified VSE Server Certificate as the Common Name. Web browsers usually expect the common name in the received server certificate to be the same as the IP address or symbolic name of this server. When a mismatch exists, the security message that is shown in Figure 5-21 appears.

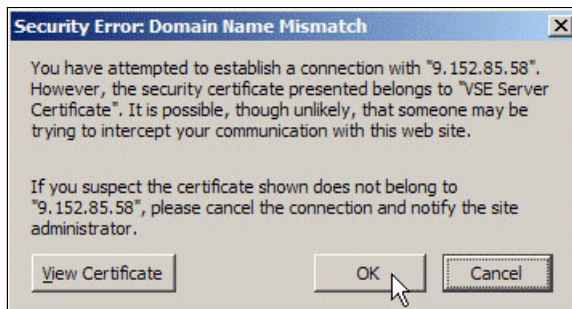


Figure 5-21 Domain name mismatch

Click **OK**.

The browser window with the sample HTML opens, as shown in Figure 5-22.

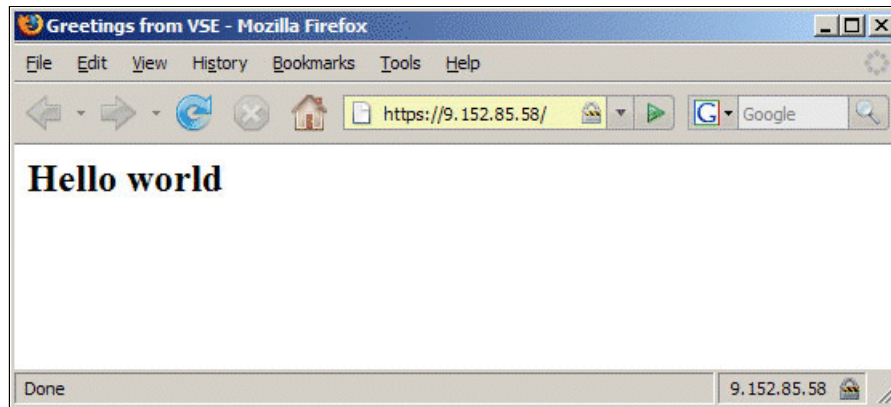


Figure 5-22 Browser with example output

For more information about checking the available SSL cipher suites in Microsoft Internet Explorer, see 5.3.4, “Configuring ciphers in Internet Explorer” on page 218. If you do not receive any errors, or you do not want to force any specific cipher suite for your SSL connection, you can skip that section.

### 5.3.4 Configuring ciphers in Internet Explorer

SSL cipher suites are not visible through any Internet Explorer dialogs. Instead, they are defined in the Windows registry; therefore, they affect your entire computer. For more information, search the Microsoft Support knowledge database by using keyword “SSL cipher suites”:

<http://support.microsoft.com>

## 5.4 Debugging SSL/TLS connections

This section shows how you can activate tracing of SSL/TLS connections with TCP/IP for VSE/ESA on z/VSE and on the workstation.

### 5.4.1 Tracing on z/VSE

To activate tracing on z/VSE, catalog the phase \$SOCKDBG in your TCP/IP sublibrary on z/VSE, as shown in Example 5-14.

*Example 5-14 Catalog job for \$SOCKDBG*

---

```
* $$ JOB JNM=SOCKDBG,CLASS=A,DISP=D
// JOB $SOCKDBG
// OPTION CATAL
// LIBDEF *,CATALOG=lib.sublib
// EXEC ASMA90,SIZE=ASMA90
      PUNCH    ' PHASE $SOCKDBG,* '
$SOCKDBG CSECT
      SOCKDBG CSECT,          GENERATE A PHASE                X
              FLO1=$DBGWLST, +DBGWLOG, MESSAGES TO SYSLST AND SYSLOG X
              FLO2=$DBGISON,  DEBUG IS ON                    X
```

```

        FLO3=$DBGNONE,    NONE                                X
        MSGT=$DBGALL,    ISSUE ALL DIAGNOSTIC MESSAGES      X
        DUMP=$DBGNONE,   NO  DIAGNOSTIC SDUMPS FOR IPNRBDC   X
        SSLD=$DBGSDMP,   YES DIAGNOSTIC SDUMPS FOR IPCRYPTO X
        CIAL=$DBGSDMP,   YES DIAGNOSTIC SDUMPS FOR IPDSCIAL  X
        CECZ=$DBGNONE    NO  DIAGNOSTIC SDUMPS FOR CIALCECZ
    END  $SOCKDBG

/*
// EXEC LNKEDT,SIZE=512K
/*
/&
* $$ E0J

```

---

Ensure that the SSL server on z/VSE (for example the z/VSE Connector Server) uses option NOSYSDMP so that the trace output, which consists of many small SDUMPs, is written to SYSLST.

You also should specify the following statement in your job:

```
// UPSI 1
```

## 5.4.2 Tracing in Java

To activate SSL/TLS tracing in a Java application, such as z/VSE Navigator, use the Java DEBUG option, as shown in the following example:

```
java -Djavax.net.debug=all com.ibm.vse.navigator.VSENavigator %*
```

In this example, you modify the run.bat file of the z/VSE Navigator installation.





## CICS Web Support security

In this chapter, we describe the SSL setup for CICS Web Support (CWS) in scenarios with z/VSE acting as server.

The CICS Transaction Server is the central component of z/VSE. By using CWS, you can communicate with CICS TS from a web browser to start transactions. The transactions might have different security requirements when they are used from CWS.

Therefore, we described the following levels of CWS security:

- ▶ CWS without security
- ▶ Secured CWS with server authentication
- ▶ Secured CWS with client authentication

After the CICS setup, we describe the customization steps of different web browsers.

In the last section of this chapter, we provide more service hints.

This chapter includes the following topics:

- ▶ 6.1, “Introduction” on page 222
- ▶ 6.2, “Setting up CWS” on page 222
- ▶ 6.3, “Setting up secure CWS” on page 224
- ▶ 6.4, “Client setup with Mozilla Firefox” on page 228
- ▶ 6.5, “Client setup with Microsoft Internet Explorer” on page 237
- ▶ 6.6, “Setting up for client authentication” on page 243
- ▶ 6.7, “Observations” on page 248

## 6.1 Introduction

Secure CWS requires RSA key pairs and digital certificates on the server and on the client side. For more information about getting keys and certificates, see 5.1, “Generating the server key and certificates” on page 198.

In the examples, we use the following software:

- ▶ z/VSE V4R2
- ▶ TCP/IP for VSE 1.5F
- ▶ z/VSE Connector Server as part of z/VSE V4R2
- ▶ CICS TS 1.1 as part of z/VSE V4R2
- ▶ Microsoft Windows XP Professional, SP2
- ▶ Java 1.6.0\_03 from Sun Microsystems
- ▶ Keyman/VSE, update from 08/2007
- ▶ Microsoft Internet Explorer 6.0
- ▶ Mozilla Firefox 2.0.0.11

Up to CICS TS V2.1, SSL/TLS support for CWS was based on TCP/IP for VSE.

CICS TS V2.2 (z/VSE 6.2) can choose between the SSL/TLS implementation from CSI and OpenSSL. OpenSSL provides more advanced security features, such as:

- ▶ Elliptic Curve Cryptography (ECC)
- ▶ Diffie-Hellman based key exchange
- ▶ More cipher suits

The implementation is based on the IBM provided BPX callable services. To select the wanted SSL/TLS functionality, you can use the “// SETPARM BPX\$GSK” variable.

## 6.2 Setting up CWS

For more information about how to configure CICS Web Support, see *CICS Transaction Server for VSE/ESA Enhancements Guide*, SC34-2685.

Setting up CWS includes the following overall steps:

1. Modify the DFHSITSP skeleton.

To activate CWS, you must modify a copy of the job skeleton DFHSITSP in ICCF library 59. Change the parameter TCPIP=NO to TCPIP=YES and submit this job.

2. Create a conversion table.

A conversion table phase is required. It is used by CWS to convert incoming requests to EBCDIC and outgoing responses back to ASCII. You can use the job skeleton of the default table DFHCNV from ICCF library 59 to create such a phase.

3. Apply changes in TCP/IP.

As the last point in the list of changes, TCP/IP requires certain modifications. A DNS server should be provided to allow the mapping from IP addresses to host names and vice versa. In addition, TCP/IP must know its own host name and IP address. In the example, you add the following statement to the TCP/IP IPINIT member:

```
DEFINE NAME,NAME=VSER06,IPADDR=9.152.84.115
```

4. Define a TCP/IP service.
5. Recycle CICS.

For more information about the definition of the TCP/IP service, see 6.2.1, “Defining the TCP/IP service” on page 223.

## 6.2.1 Defining the TCP/IP service

Use the CEDA transaction to define a TCP/IP service. SSL is in use, as shown in Figure 6-1.

```

OVERTYPE TO MODIFY                                CICS RELEASE = 0411
CEDA DEFine TCpipservice( NOSSL      )
  TCpipservice   : NOSSL
  Group          : EXTRACT
  Description    ==>
  Urm           ==>
  Portnumber    ==> 01082                1-65535
  Certificate    ==>
  SStatus       ==> Open                 Open ! Closed
  SSL           ==> No                   Yes ! No ! Clientauth
  Attachsec     ==> Verify                Local ! Verify
  TTransaction  ==> CWXN
  Backlog       ==> 00001                0-32767
  TSqprefix     ==>
  Ippaddress    ==>
  Socketclose   ==> No                   No ! 0-240000

```

Figure 6-1 Define TCP/IP service in CICS

Activate your changes, as shown in the following example:

```

CEMT SET TCPIPS(NOSSL) CLOSED
CEDA INSTALL TCPIPS(NOSSL) GROUP(EXTRACT)

```

An important point is that TCP/IP must be started before CICS is started; otherwise, CICS cannot open the TCP/IP service. This issue might be a problem during IPL. After defining the TCP/IP service, you can recycle CICS and then the new TCP/IP service should be open when CICS is up again.

After recycling CICS, you can use the CEMT transaction to check whether the TCP/IP service is open, as shown in Example 6-1.

Example 6-1 Get the status of TCP/IP service NOSSL

---

```

171 cemt i tcpips(nossl)
F2-0171
F2 0173
      Tcpipservice(NOSSL)
      Backlog( 00010 )
      Connections(0000)
      Port(01082)
      Ssltype(Nossl)
      Openstatus( Open )
      Transid(CWXN)
      Urm( DFHWBADX )
      Ippaddress(9.152.84.115)

```

---

To ensure that TCP/IP is initialized before CICS, you can modify the USERBG procedure (see skeleton SKUSERBG in ICCF library 59) to have CICS waiting until TCP/IP is up and running. In the following example, IESWAIT is started with parameter 07, which indicates the TCP/IP partition (F7), as shown in the following example:

```
// PWR PRELEASE RDR,TCPIP00 TCP/IP  
// EXEC IESWAIT,PARM='07'
```

## 6.2.2 Connecting to CWS

To determine whether the connection to CWS is working, you can start the transaction CEMT, as shown in Figure 6-2.

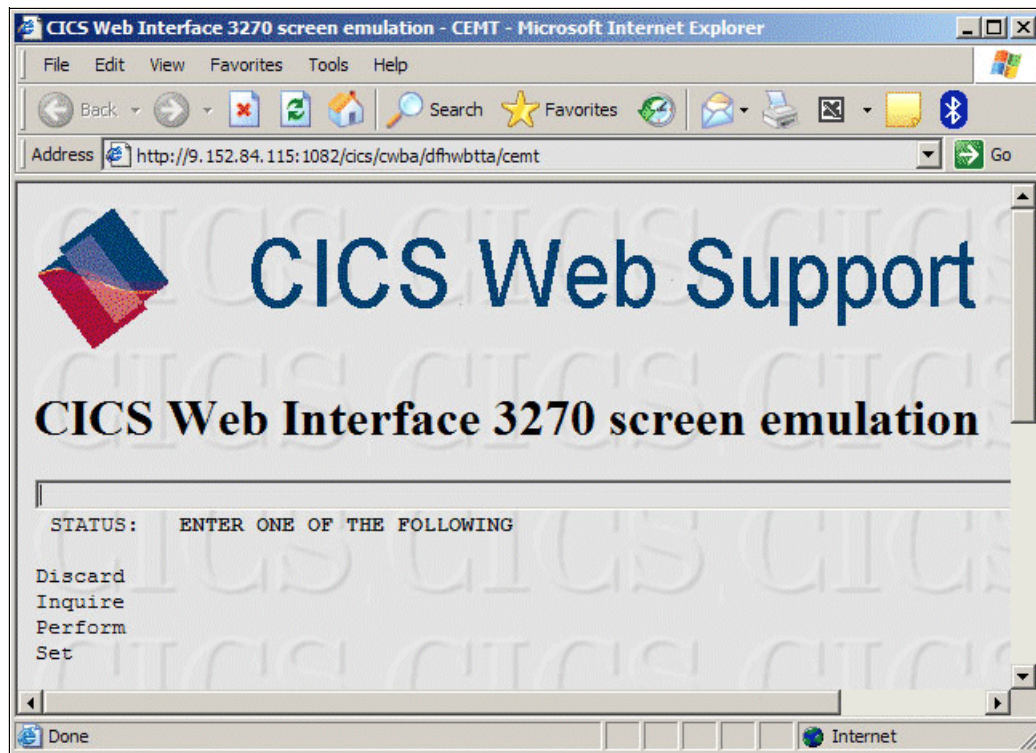


Figure 6-2 CWS output of the transaction CEMT

The next section describes how the previously defined CWS is SSL-enabled.

## 6.3 Setting up secure CWS

A CICS TCP/IP service must be defined for CWS with SSL. For simplification, we call it SSL. When defining the TCP/IP service SSL, you can duplicate your previously defined NOSSL service. Enter the following CICS transaction:

```
CEDA V TCPIPS (*) GR(*)
```

To create the SSL service, use the **copy** command, as shown in Example 6-2 on page 225.



*Example 6-2 Create service SSL as copy of service NOSSL*

```
ENTER COMMANDS
NAME      TYPE          GROUP
NOSSL    TCPIPService EXTRACT  copy as(ssl)
```

Example 6-3 shows that the newly defined TCP/IP service appears in the list when the services are displayed again.

*Example 6-3 View TCP/IP services*

```
ENTER COMMANDS
NAME      TYPE          GROUP
NOSSL    TCPIPService EXTRACT
SSL      TCPIPService EXTRACT
```

### 6.3.1 Configuring the TCP/IP service for SSL

The following SSL-related parameters that must be configured for the TCP/IP service:

► SSL

Specify Yes for SSL server authentication or Clientauth for SSL client authentication. For more information, see 6.6, “Setting up for client authentication” on page 243.

► Certificate

This parameter is the member name of the VSE keyring members that consists of a PRVK, CERT, and ROOT member type.

► Portnumber

This parameter is the number of the secure CWS port.

The TCP/IP service SSL contains the definition from the NOSSL service. Use the **CEDA ALTER** command to modify the TCP/IP service SSL. Change the SSL parameter to Yes, use a currently free TCP/IP port number, and specify the name of the VSE keyring members<sup>1</sup> in the Certificate field, as shown in Figure 6-3.

```
OVERTYPE TO MODIFY                                     CICS RELEASE = 0411
CEDA Alter TCpipservice( SSL                          )
TCpipservice   : SSL
Group          : EXTRACT
Description    ==>
Urm            ==> DFHWBADX
Portnumber     ==> 01083                               1-65535
Certificate    ==> CWS01
Status         ==> Open                               Open ! Closed
SSL           ==> Yes                                 Yes ! No ! Clientauth
Attachsec     ==> Verify                             Local ! Verify
TRansaction   ==> CWXN
Backlog       ==> 00010                               0-32767
TSqpprefix    ==>
Ippaddress    ==>
SOketclose   ==> No                                  No ! 0-240000
```

Figure 6-3 Alter TCP/IP service SSL

After configuring the TCP/IP service, more changes must be made in the CICS initialization.

<sup>1</sup> For more information about creating a self-signed keyring, see 5.1, “Generating the server key and certificates” on page 198.

## 6.3.2 Configuring the CICS system initialization parameters

The following CICS initialization parameters must be configured for SSL:

- ▶ Encryption strength
- ▶ Key file
- ▶ SSL delay

These parameters are configured in the DFHSITSP job. You can use the skeleton DFHSITSP in ICCF library 59 and change them, as shown in Example 6-4.

*Example 6-4 Update DFHSITSP for SSL*

---

```

* $$ JOB JNM=DFHSITSP,CLASS=A,DISP=D,NTFY=YES
* $$ LST CLASS=Q,DISP=H
// JOB DFHSITSP ASSEMBLE
// LIBDEF *,CATALOG=PRD2.CONFIG
// LIBDEF SOURCE,SEARCH=(PRD1.BASE,PRD1.MACLIB)
// OPTION CATAL,LIST
// EXEC ASMA90,SIZE=(ASMA90,64K),PARM='EXIT(LIBEXIT(EDECKXIT)),SIZE(MAXC
-200K,ABOVE)'
*****
*
* 5686-CF7 (C) COPYRIGHT IBM CORP. 1984, 2004
*
*****
          TITLE 'DFHSITSP FOR CICS TS APPLID DBDCCICS'
          PUNCH ' CATALOG DFHSITSP.OBJ REP=YES'
          DFHSIT TYPE=CSECT,
          ...
          ENCRYPTION=STRONG,      SSL ENCRYPTION
          ...
          KEYFILE=CRYPTO.KEYRING,  KEY RESIDENCE FOR SSL
          ...
          SSLDELAY=600,            DELAY FOR SSL CONNECTION
          ...
          DUMMY=DUMMY              TO END MACRO
          END DFHSITBA
/*
// IF $MRC GT 4 THEN
// GOTO NOLINK
// EXEC LNKEDT,PARM='MSHP'
/. NOLINK
/*
/&

```

---

### Encryption strength

Unlike configuring SSL for HTTP, Telnet, or FTP, where client and server specify the SSL cipher suites explicitly, CICS Web Support knows three levels of encryption strength: weak, normal, and strong. These levels are mapped to the list of SSL cipher suites, as listed in Table 6-1.

*Table 6-1 Available cipher suites for CWS*

CICS parameter	Hex code	Cipher suite	Encryption strength
WEAK	01	SSL_RSA_WITH_NULL_MD5	No encryption
	02	SSL_RSA_WITH_NULL_SHA	No encryption

CICS parameter	Hex code	Cipher suite	Encryption strength
NORMAL	08	SSL_RSA_EXPORT_WITH_DES40_CBC_SHA	40 bits
	09	SSL_RSA_WITH_DES_CBC_SHA	56 bits
STRONG	0A	SSL_RSA_WITH_3DES_EDE_CBC_SHA	112 <sup>a</sup> bits
	62	RSA1024_EXPORT_DES_CBC_SHA	56 bits

a. The key length of triple-DES is often stated as 168 bits, but it effectively results in an encryption strength of 112 bits. For more information, see Table 4-2 on page 99.

**Note:** Consider the following points:

- ▶ The TCP/IP fix number ZP15E250 (APAR PK33472) removes the restriction that some cipher suites can be used with a specific RSA key length only. Therefore, you can use each cipher suite with each possible RSA key length. As of this writing, z/VSE supports RSA key lengths of 512, 1024, and 2048 bits.
- ▶ CWS does not support the AES encryption algorithm, which is the reason that the AES-related cipher suites 2F and 35 are not listed.
- ▶ If you did not install TCP/IP fix ZP15E250 (APAR PK33472), the cipher suites NULL\_MD5 (X'01'), NULL\_SHA (X'02'), and DES40\_CBC\_SHA (X'08') must have a 512-bit key on the VSE side. They cannot be used with a 1024-bit key.
- ▶ The CICS Transaction Server supports SSL 3.0 handshaking only.

## Key file

The key file parameter specifies the library and sublibrary name of the z/VSE keyring library. In this case, it is CRYPTO.KEYRING. For more information, see 5.1, “Generating the server key and certificates” on page 198, where we cataloged the keyring members into this sublibrary.

## SSL delay

The SSL delay parameter specifies the length of time in seconds for which CICS retains session IDs for secure socket connections. Session IDs are tokens that represent a secure connection between a client and an SSL server. While the session ID is retained by CICS within the SSLDELAY period, CICS can continue to communicate with the client without the significant overhead of an SSL handshake. The number is a value of seconds 0 - 86400 (the default is 600).

After cataloging the changes in DFHSITSP and recycling CICS, the SSL setup is complete. We now must configure the client side, which means importing the certificates into the web browsers.

### 6.3.3 Configuring OpenSSL

With CICS TS V2.2 (z/VSE 6.2), the secure CWS implementation can use the OpenSSL functions. The implementation is based on the IBM provided BPX callable services. This feature enables the use of vendor-independent SSL/TLS functionality. To select the wanted SSL/TLS functionality, you can use the “// SETPARM BPX\$GSK” variable. Table 6-2 shows how to enable usage of OpenSSL.

Table 6-2 Enabling OpenSSL usage

Scope	Configuration member	Variable specification
Use OpenSSL system-wide	System startup (USERBG)	// SETPARM SYSTEM BPX\$GSK =
Specific job	CICS startup job (SKCICS)	// SETPARM BPX\$GSK =

The z/VSE default system setup is to use the vendor provided SSL functions from CSI. To use the OpenSSL functionality, request it by way of BPX\$GSK variable, the BPX\$GSK variable values are listed in Table 6-3.

Table 6-3 Select SSL /TLS implementation

Variable value	Description
Variable not set	Default to CSI's SSL implementation
CSI	Select CSI's implementation explicit
IJBGSKOS	Use OpenSSL in Metal-C

OpenSSL requires keys and certificates in OpenSSL PEM format.

## 6.4 Client setup with Mozilla Firefox

The self-signed root certificate must be imported into the Mozilla Firefox certificate store. To prevent losing the contained private key, import the entire PFX file into Firefox.

**Note:** Mozilla Firefox does not accept *normal* or *weak* encryption as specified in the DFHSITSP. You must specify *strong* encryption. We are unable to find any information about how and where cipher suites can be configured for Firefox.

## 6.4.1 Importing the z/VSE certificates during session establishment

One possible way to import the z/VSE certificates into the Firefox certificate store is to connect. You are prompted to accept or reject the received server certificate, as shown in Figure 6-4.

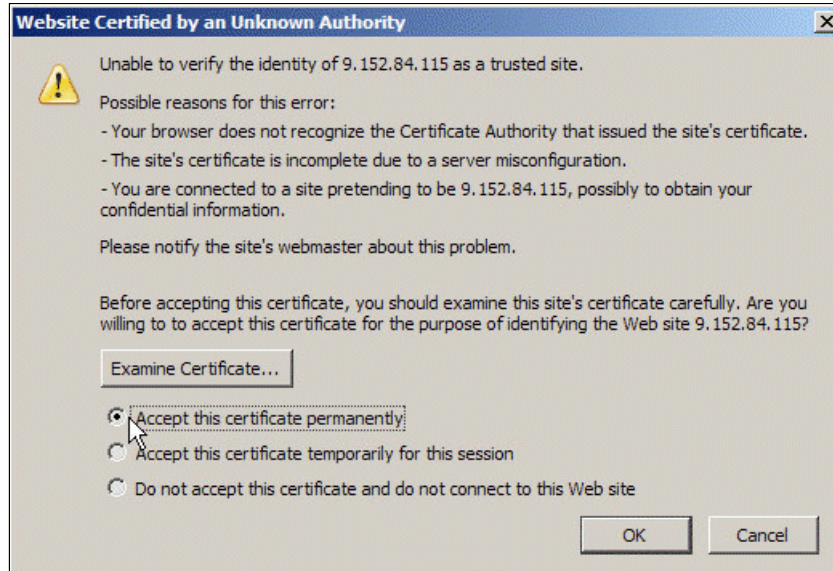


Figure 6-4 Accept certificate for Firefox

You can accept the z/VSE server certificate permanently and click **OK**. When you open the Firefox certificate store, you see the imported certificate under the Web Sites tab, as shown in Figure 6-5.

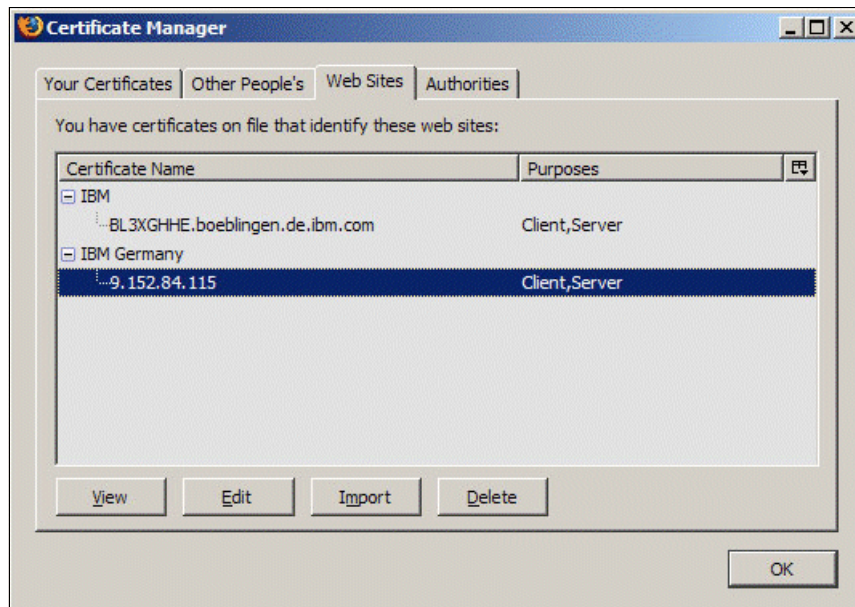


Figure 6-5 View certificates on Firefox

## 6.4.2 Manually importing the z/VSE certificates into Firefox

Another possibility is to import the certificate manually. In the Firefox main window, click **Tools** → **Options**. In the Options window, select **Advanced** → **Encryption**, as shown in Figure 6-6.

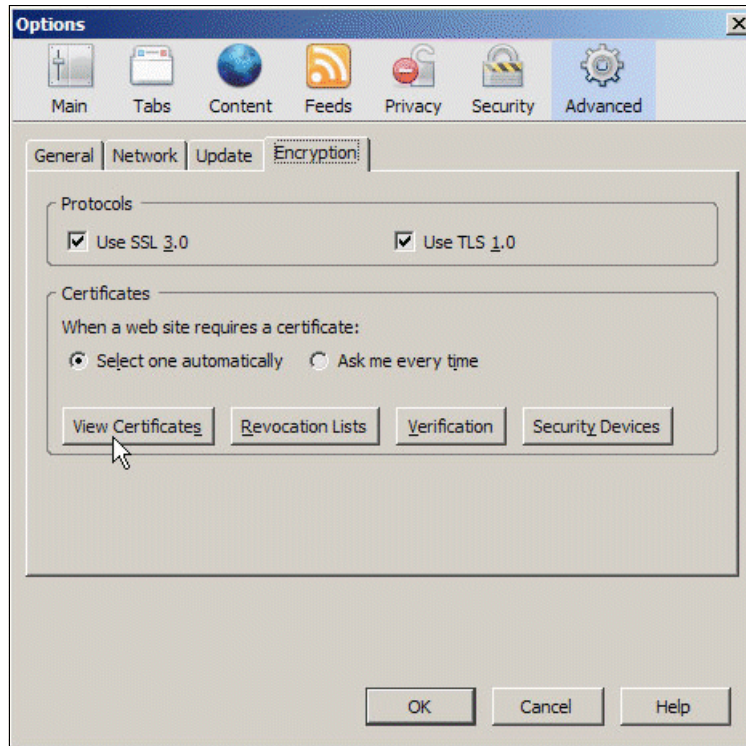


Figure 6-6 Import certificate into Firefox manually

Click **View Certificates**. The following choices are available:

- ▶ Importing the complete z/VSE keyring file
- ▶ Importing the selected z/VSE server certificate



## Importing the complete z/VSE keyring file

Select tab **Your Certificates** and click **Import**, as shown in Figure 6-7.

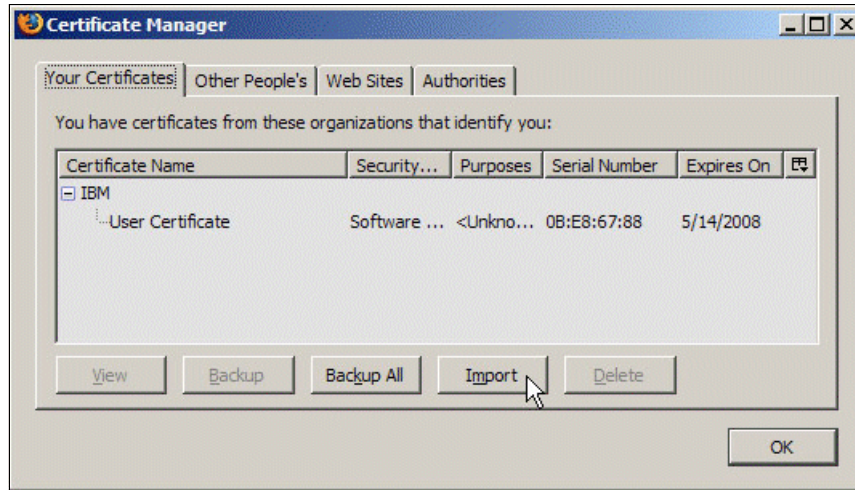


Figure 6-7 Import your certificate

In the File name to restore window, select the z/VSE keyring file and click **Open**, as shown in Figure 6-8.

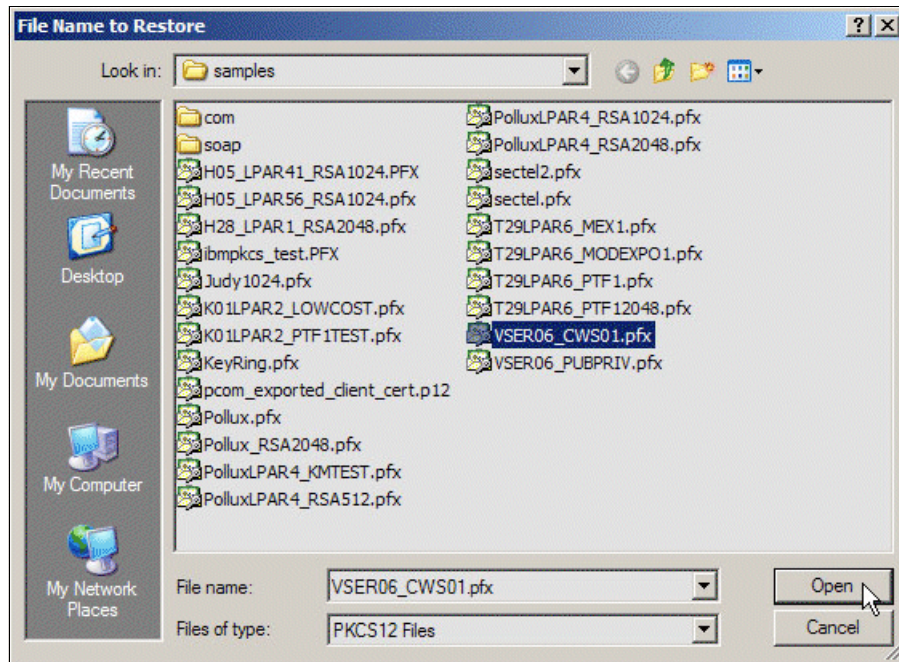


Figure 6-8 Open the z/VSE keyring file

As shown in Figure 6-9, enter the password of the z/VSE keyring file that you specified when the file with Keyman/VSE was created, as described at 5.1, “Generating the server key and certificates” on page 198.



Figure 6-9 Enter the password of the keyring file

Click **OK**. A confirmation message is displayed, as shown in Figure 6-10.

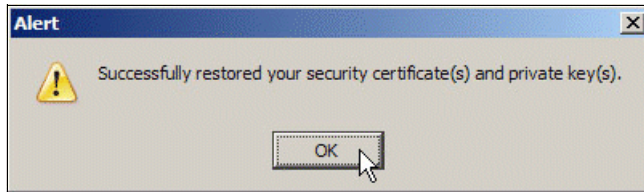


Figure 6-10 Keyring successfully stored

Click **OK** to return to the Certificate Manager window, as shown in Figure 6-11.

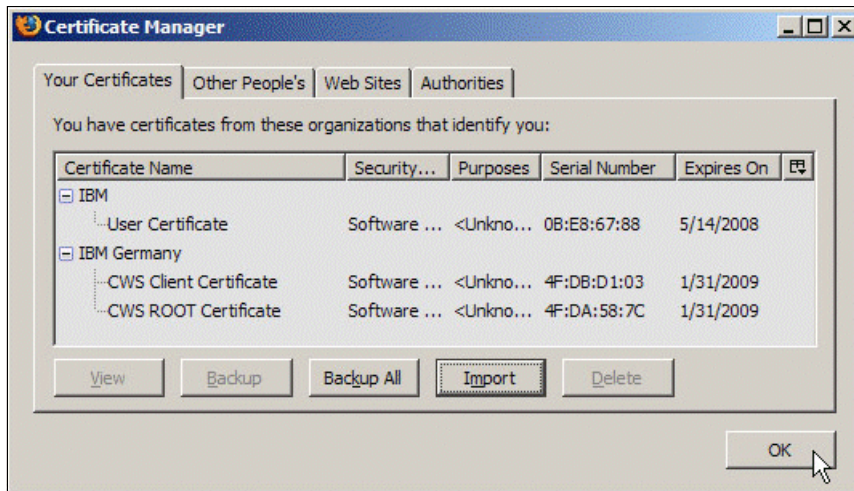


Figure 6-11 Certificate Manager

The following certificates are imported:

- ▶ CWS root certificate can now be used to verify the z/VSE server certificate when establishing an SSL connection.
- ▶ The client certificate is not used.

### Importing the z/VSE certificate selectively

When importing the complete z/VSE keyring file, the required root certificate was imported. Another method that can be used is to import only the z/VSE server certificate into the certificates under tab Web Sites. Then, trust must be established manually. The z/VSE root certificate is not used in this case.



The import window under Web Sites does not accept complete keyring files (PFX), but prompts the user for a binary certificate file that contains only one certificate. You can use Keyman/VSE to export the z/VSE server certificate into a binary file.

Start Keyman/VSE again and open the z/VSE keyring file. Right-click the z/VSE certificate and select **Export binary form**, as shown in Figure 6-12.

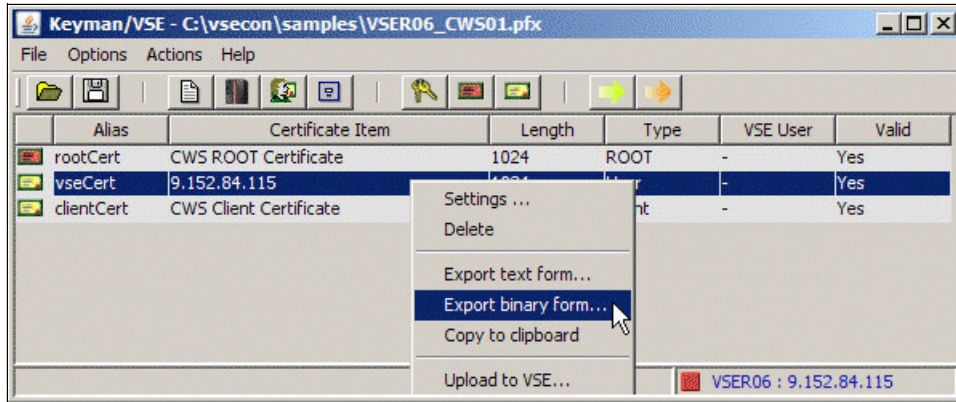


Figure 6-12 Export z/VSE server certificate

Specify a file name with file extension .cer and save the file. You can now import this binary certificate file into Firefox. Ensure that tab Web Sites is selected and click **Import**, as shown in Figure 6-13.

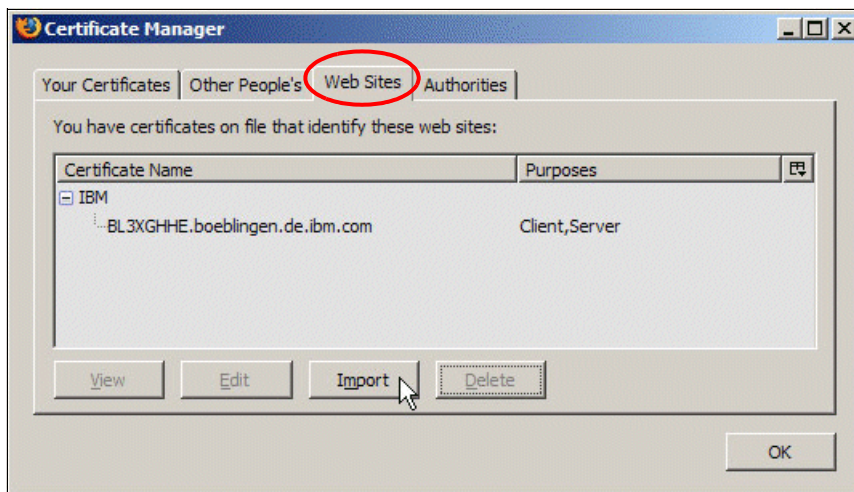


Figure 6-13 Select tab Web Sites to import certificate

Select the binary certificate and click **Open**, as shown in Figure 6-14.

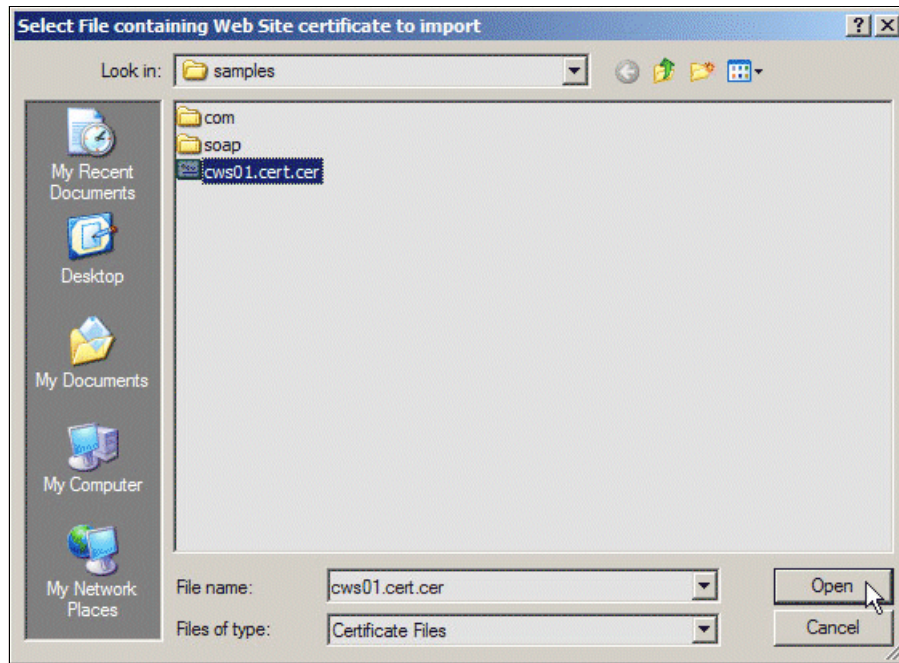


Figure 6-14 Open the certificate

The z/VSE certificate is now imported, but the SSL connection does not work until you specify to trust this certificate. When importing the complete z/VSE keyring file, the trust was established during the import step, because the process also read the z/VSE root certificate.

Figure 6-15 shows <Unknown> in column Purposes. This purpose means that the certificate is not trusted. The connection does not work in this case. To establish trust, select the certificate and click **Edit**.

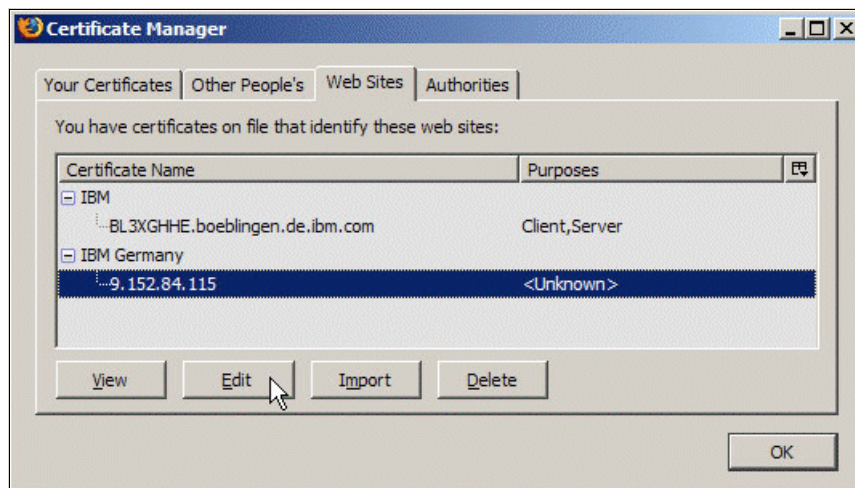


Figure 6-15 Show the status of the certificate

Select **Trust the authenticity of this certificate**, then click **OK**, as shown in Figure 6-16.

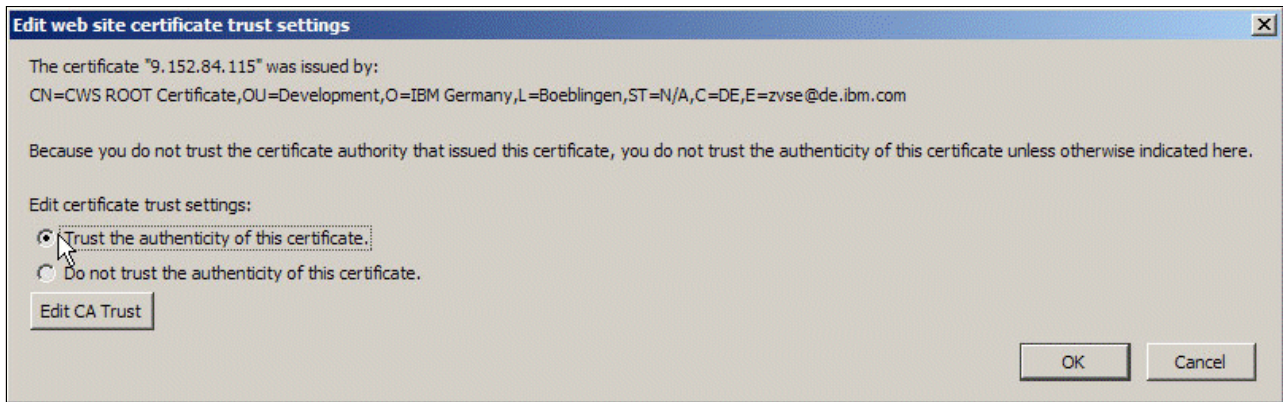


Figure 6-16 Edit trust setting

The status (purposes) that is shown in Figure 6-17 shows the certificate as trusted.

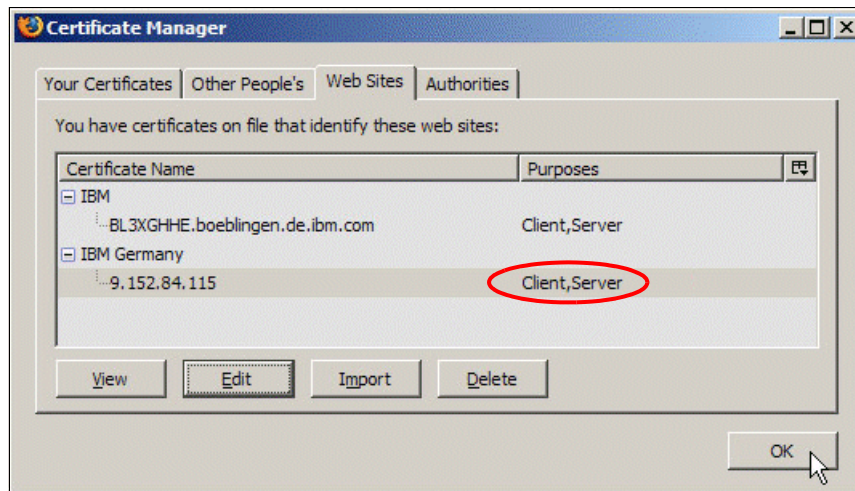


Figure 6-17 Show new certificate status

The z/VSE server certificate is now imported into Firefox and manually trusted this certificate.

Firefox is now ready for connecting to CWS with SSL server authentication.

### 6.4.3 Configuring cipher suites in Firefox

As of this writing, we have no information about how to influence the ciphers that are used in Mozilla Firefox.



## 6.4.4 Starting a secure session with Firefox

After CICS is restarted on the z/VSE side and the SSL TCP/IP service is open, as shown in Example 6-5, you can connect to CWS with SSL.

*Example 6-5 Control status of TCP/IP services in CICS*

---

```
msg f2,data=cent i tcpips
F2 0173
      TcpiPs(NOSSL  ) Bac( 00010 ) Con(0000) Por(01082)
      Ope Tra(CWXN) Urm( DFHWBADX ) Ipa(9.152.84.115  )           Wai
      TcpiPs(SSL    ) Bac( 00010 ) Con(0000) Por(01083) Ssl
      Ope Tra(CWXN) Urm( DFHWBADX ) Ipa(9.152.84.115  )           Wai
```

---

In the web browser address field, you must specify `https` and the secure port number 1083, as shown in Figure 6-18.

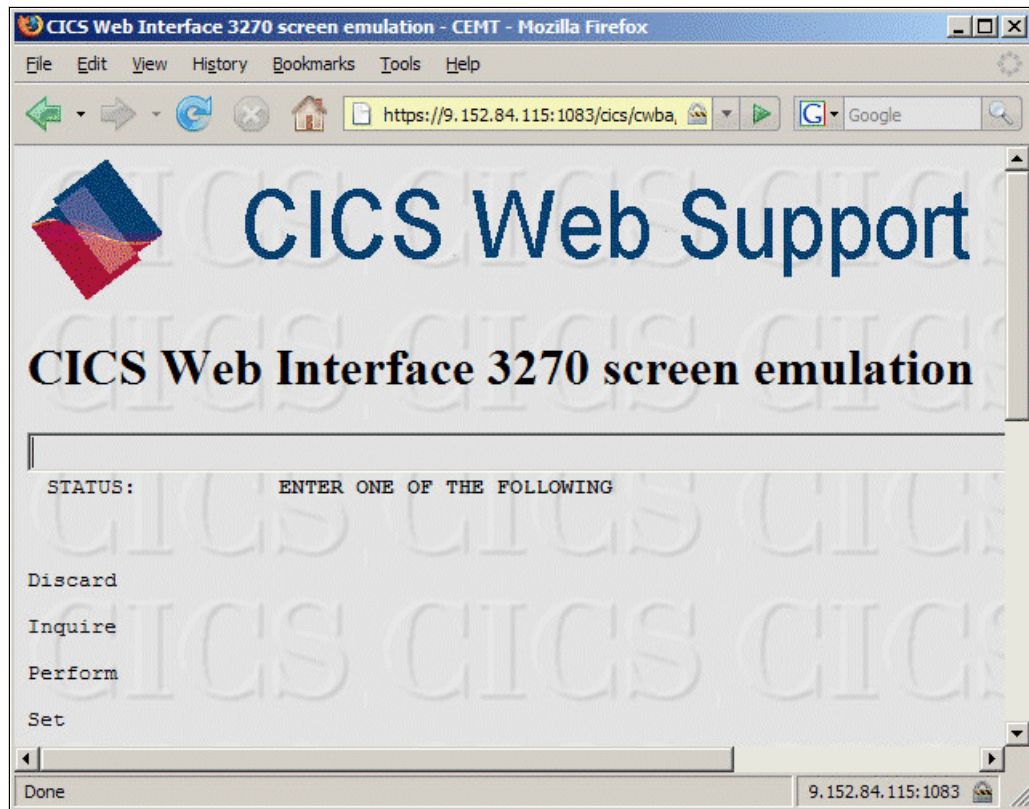


Figure 6-18 Firefox browser with CICS Web Support contents

When you double-click the closed lock icon at the right bottom corner of the browser window, the properties of the current SSL session are displayed, as shown in Figure 6-19 on page 237.



Figure 6-19 View properties of the current SSL session

## 6.4.5 Displaying SSL properties in Mozilla Firefox

Various add-ons are available for the Mozilla Firefox browser. One add-on, called Calomel, can display a summary of the SSL connection. You can download the add-on from this website:

<https://addons.mozilla.org/de/firefox/addon/207653/>

When you are connected by way of SSL, the Calomel toolbar button changes color, depending on the strength of encryption from red (weak) to green (strong). The drop-down window shows a detailed summary of the SSL connection.

## 6.5 Client setup with Microsoft Internet Explorer

This section describes the client setup with Microsoft Internet Explorer. The main difference from Mozilla Firefox is that you import the keyring file into the Windows certificate store.

In contrast to Firefox, Internet Explorer also works with weak or normal encryption as specified in the DFHSITSP.

As with Mozilla Firefox, the following methods are available for importing the z/VSE certificates into the browser's certificate store:

- ▶ Connect and then allow Internet Explorer to import the necessary certificates while establishing the SSL session.
- ▶ Import the certificates manually.

## 6.5.1 Importing the z/VSE certificates during session establishment

Internet Explorer displays a security alert (shown in Figure 6-20) when the received server certificate cannot be verified.



Figure 6-20 Security alert

If you click **Yes**, you are connected, but the certificate is not imported in this case. However, to import the certificate, click **View Certificate**.

In the next window (shown in Figure 6-21), click **Install Certificate**.

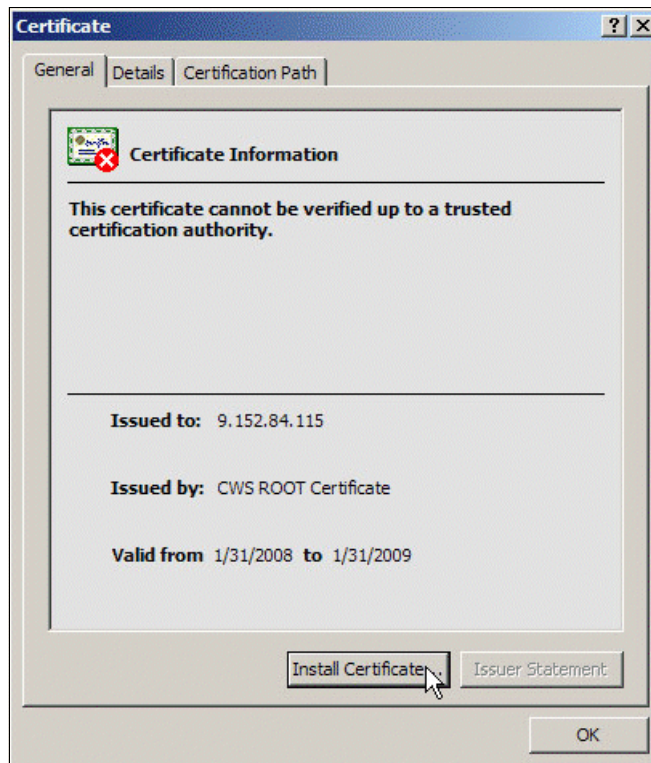


Figure 6-21 Install certificate



Click **Browse** to place the certificate in the Trusted Root Certification Authorities store, as shown in Figure 6-22.

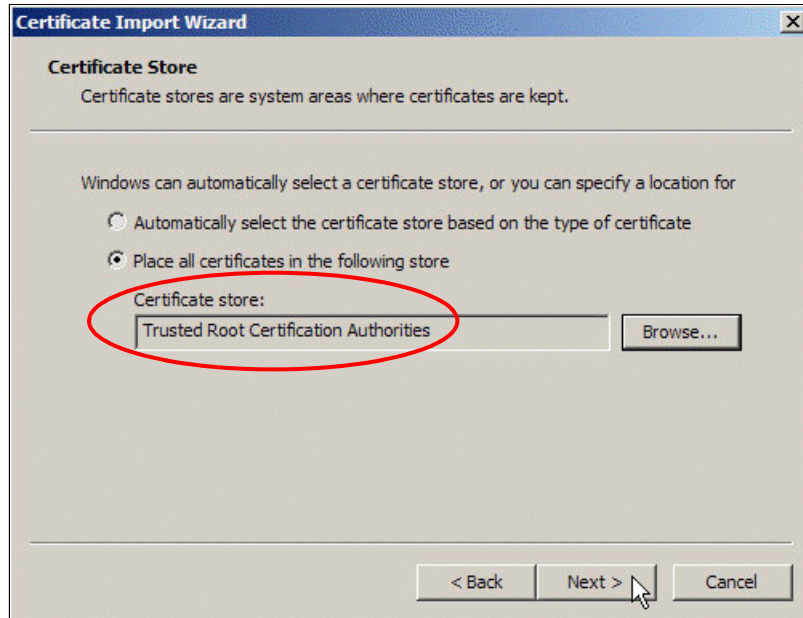


Figure 6-22 Certificate import wizard

## 6.5.2 Manually importing the z/VSE certificates into Internet Explorer

To import the certificates that were created in 5.1, "Generating the server key and certificates" on page 198, open the Windows Control Panel and double-click **Internet Options**. On the Content tab, click **Certificates**, as shown in Figure 6-23.

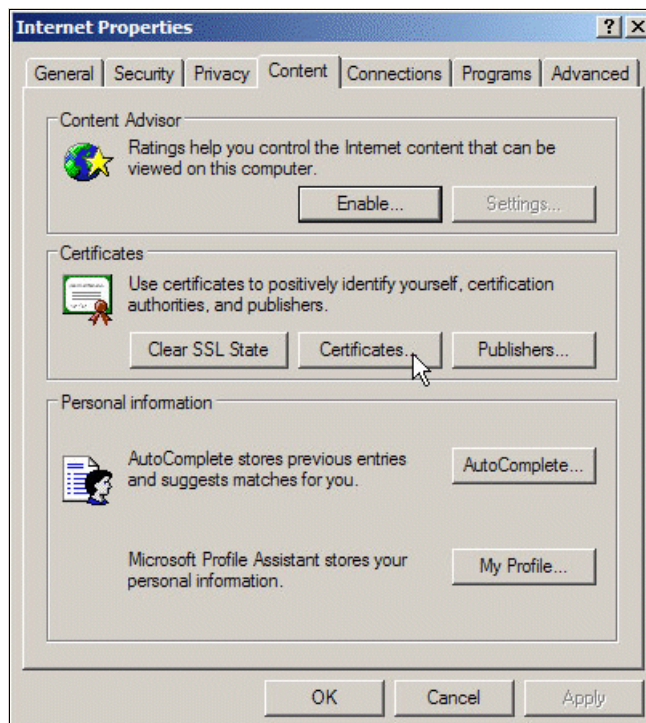


Figure 6-23 Internet properties

In the next window, click **Import** and follow the Import Wizard prompts (see Figure 6-24).

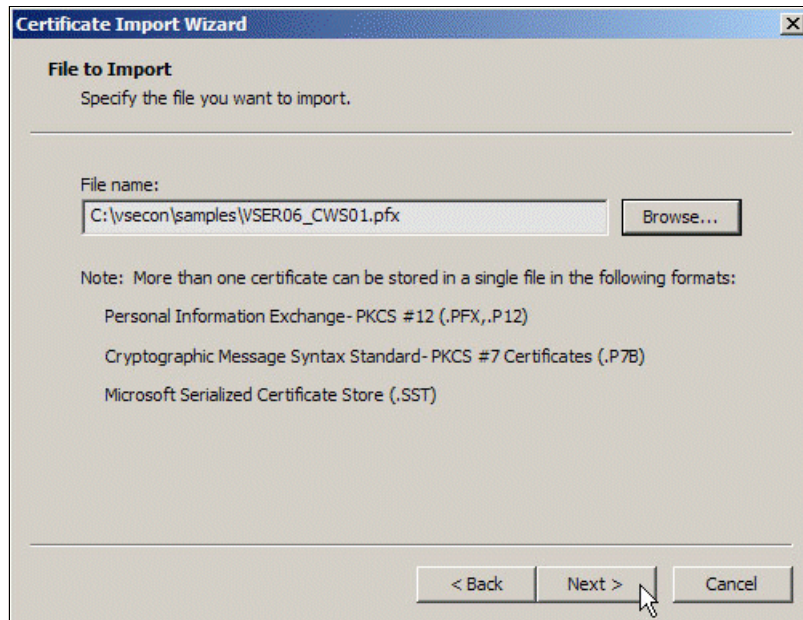


Figure 6-24 Certificate import wizard

Enter the keyring file password and click **Next**, as shown in Figure 6-25.

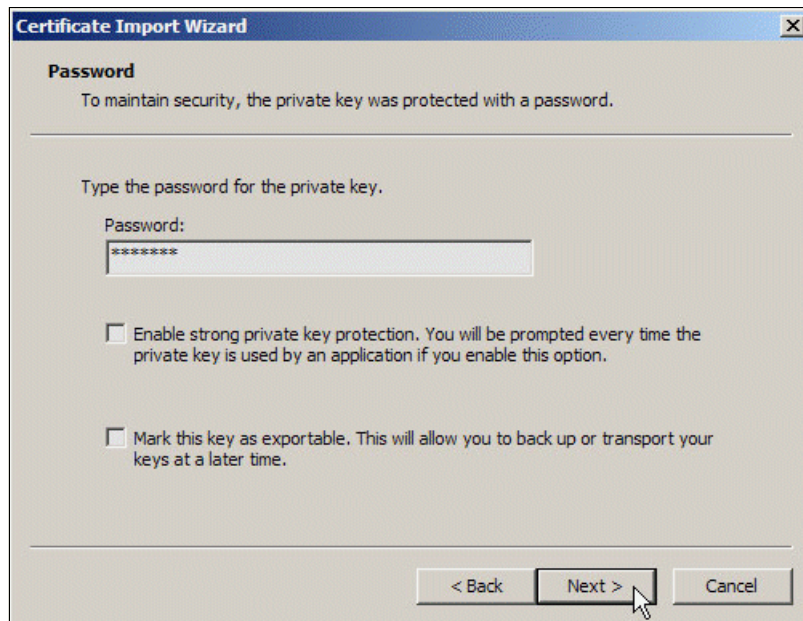


Figure 6-25 Enter password of keyring file



Select the **Automatically select the certificate store based on the type of certificate** option and then, click **Next**, as shown in Figure 6-26.

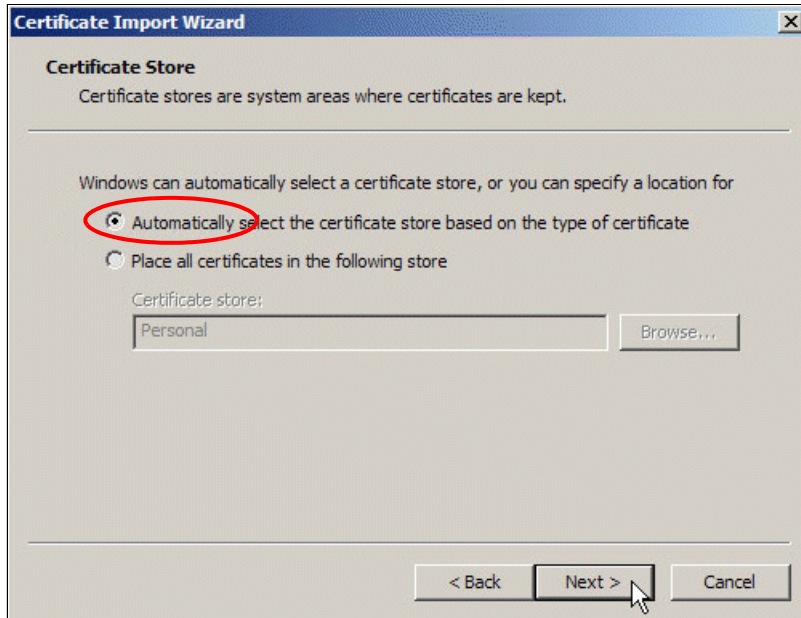


Figure 6-26 Select certificate store for personal certificate

Figure 6-27 shows that the z/VSE certificates are now imported as Personal certificates.

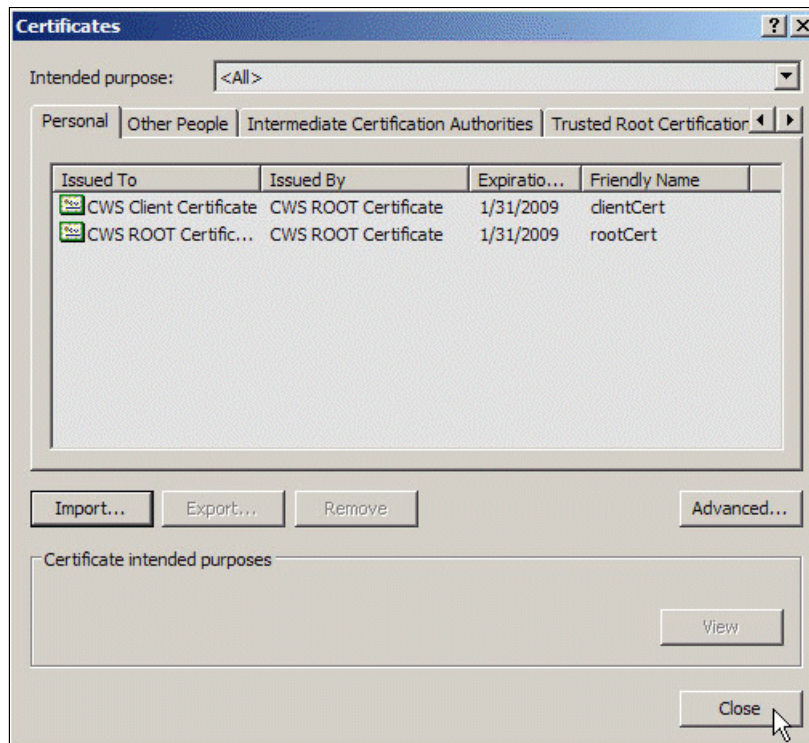


Figure 6-27 List of Personal certificates

Click **Close** to finish.

You see in section 6.6.1, “Using Internet Explorer” on page 244 that in this case you are asked to select the client certificate during session establishment, because Internet Explorer does not know which one of the two certificates will be sent to the server as the client certificate.

### 6.5.3 Configuring cipher suites in Internet Explorer

SSL cipher suites are not visible through any Internet Explorer window. Instead, they are defined in the Windows registry; therefore, they affect your entire computer. For more information, see the Microsoft Support website and search for keywords “SSL cipher suites”:

<http://support.microsoft.com>

### 6.5.4 Starting a secure session with Internet Explorer

You can now use Internet Explorer to connect to CWS with SSL server authentication, as shown in Figure 6-28.

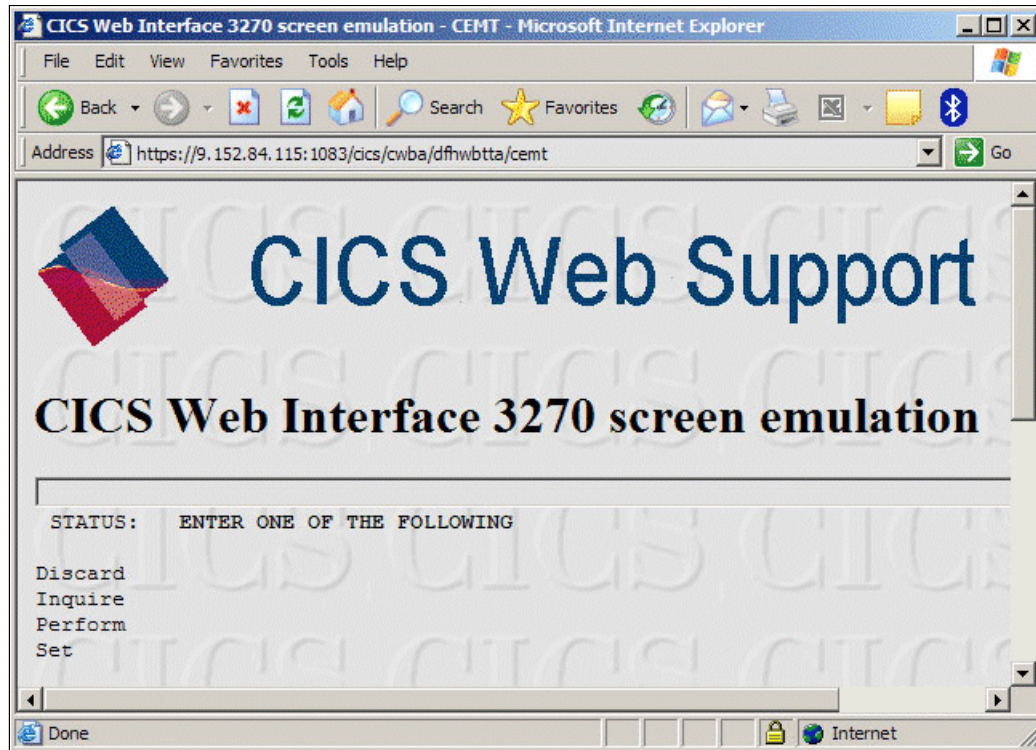


Figure 6-28 Internet Explorer with CICS Web Support contents

To view the actual encryption, select **File** → **Properties** in the Internet Explorer main window.

The Properties window opens, as shown in Figure 6-29.

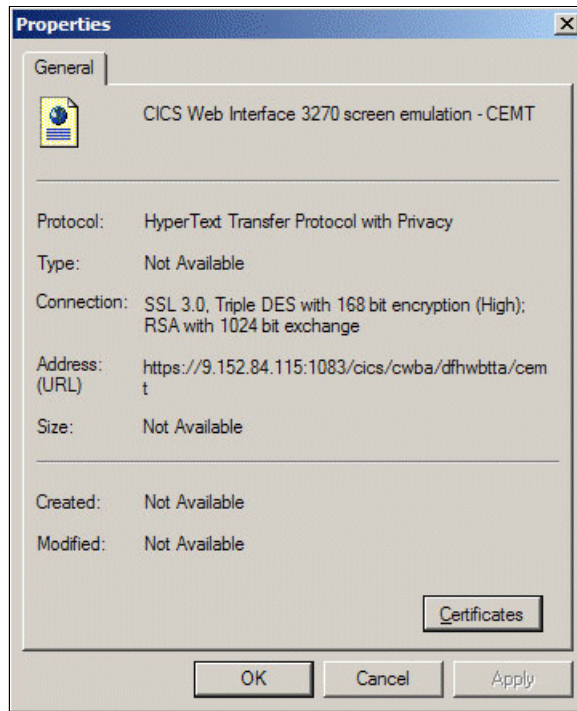


Figure 6-29 Show connection properties

## 6.6 Setting up for client authentication

SSL client authentication provides more security than server authentication because both communication partners provide a certificate to establish trust. To set up client authentication for CICS Web Support, you must change the TCP/IP service on the z/VSE side to enable client authentication. This step is done with the SSL parameter of the TCP/IP service definition, as shown in Figure 6-30.

```

OVERTYPE TO MODIFY                                     CICS RELEASE = 0411
CEDA ALTER TCpipservice( SSL                          )
TCpipservice    : SSL
Group           : EXTRACT
Description     ==>
Urm            ==> DFHWBADX
Portnumber     ==> 01083                               1-65535
Certificate    ==> CWS01
SStatus       ==> Open                                 Open ! Closed
SSL           ==> Clientauth                          Yes ! No ! Clientauth
Attachsec     ==> Verify                              Local ! Verify
TTransaction  ==> CWXN
Backlog       ==> 00010                               0-32767
TSqprefix     ==>
IpAddress     ==>
Socletclose   ==> No                                 No ! 0-240000

```

Figure 6-30 Set client authentication for TCP/IP service in CICS

Activate your changes, as shown in the following example:

```
CEMT SET TCPIPS(SSL) CLOSED  
CEDA INSTALL TCPIPS(SSL) GROUP(EXTRACT)
```

Ensure that the z/VSE client certificate is imported into the browser's certificate store.

For more information when Firefox is used, see "Importing the complete z/VSE keyring file" on page 231.

For more information when Internet Explorer is used, see 6.5.2, "Manually importing the z/VSE certificates into Internet Explorer" on page 239.

## 6.6.1 Using Internet Explorer

As described in 6.5.2, "Manually importing the z/VSE certificates into Internet Explorer" on page 239, Internet Explorer cannot determine which personal certificate to use as the client certificate. Therefore, you are prompted to select the client certificate, as shown in Figure 6-31.

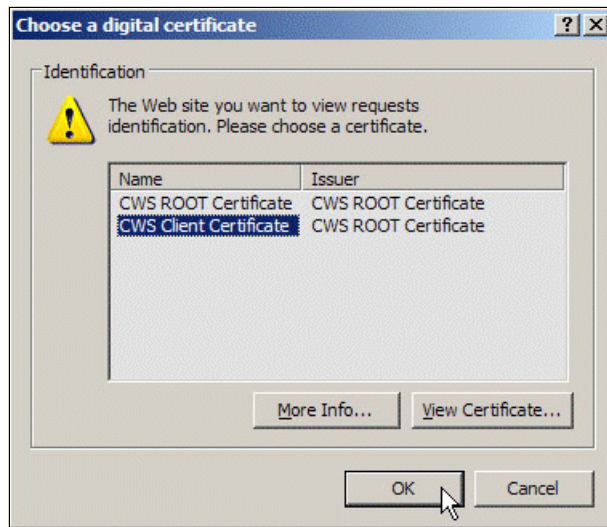


Figure 6-31 Select client certificate



To work around this prompt, manually delete the CWS root certificate from the *Personal* store and import the z/VSE keyring file again, this time selecting the option to place all certificates in the Trusted Root Certification Authorities store, as shown in Figure 6-32.

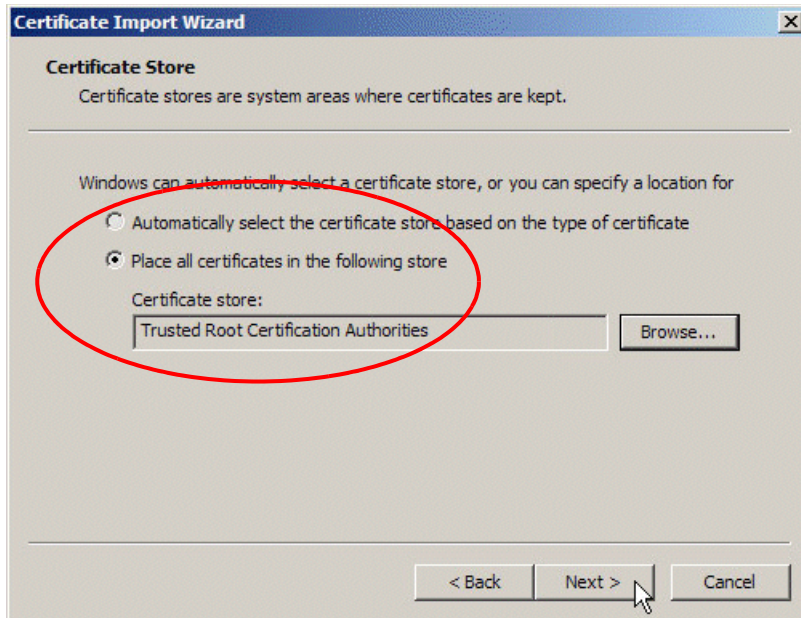


Figure 6-32 Place certificate in the storage of the trusted root certification authorities

This workaround imports only the CWS root certificate because this certificate is the only root certificate in the keyring file. You can now connect by using client authentication without being prompted.

## 6.6.2 Client authentication with user ID mapping

Mapping a client certificate to a z/VSE user ID enables the security manager, such as BSM, to check all actions against the authorization of this user.

To enable user ID mapping, complete the following steps:

1. Map the client certificate to a z/VSE user.
2. Upload the client certificate to z/VSE.
3. Activate the mapping on z/VSE.

These steps can be performed by using the Keyman/VSE tool.

First, map the created client certificate to a z/VSE user ID.

Right-click the CWS client certificate and select **Map to VSE User**, as shown in Figure 6-33.

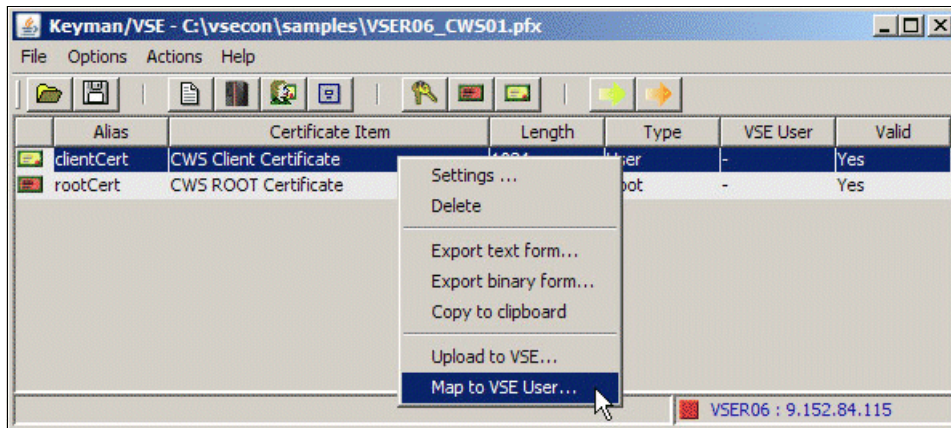


Figure 6-33 Map client certificate to z/VSE user ID

Specify the z/VSE user ID that is related to this client certificate, and click **OK**, as shown in Figure 6-34.

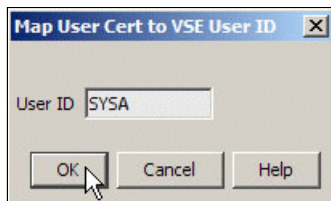


Figure 6-34 Specify z/VSE user ID for this certificate

Then, upload the client certificate to z/VSE, as shown in Figure 6-35.

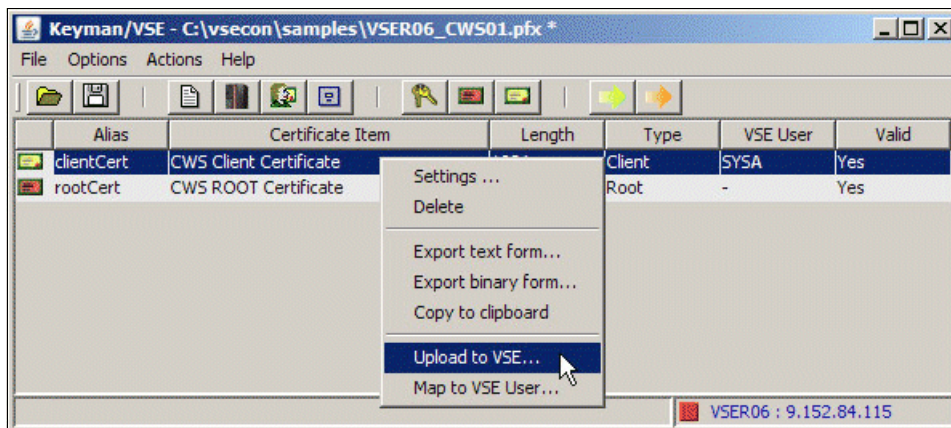


Figure 6-35 Upload the client certificate to z/VSE

Keyman uploads the client certificate to a z/VSE library member with member name equal to the z/VSE user ID and member type CCERT, as shown in Figure 6-36.

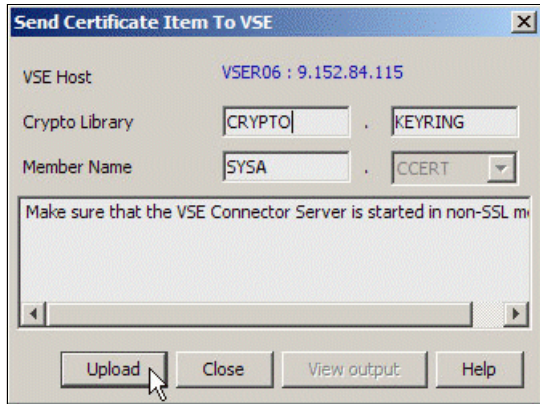


Figure 6-36 Send certificate to z/VSE

Uploading the client certificate results in two actions:

- ▶ Submitting a BSSDCERT job to catalog the certificate
- ▶ Submitting a second BSSDCERT job to activate the table of certificates

Keyman/VSE can now show the table of mapped certificates. This function is equal to the Interactive Interface dialog “Maintain Certificate - User ID List” (fastpath 284).

Click the **Client certificate mapping** button, as shown in Figure 6-37, to view more information about the certificates.

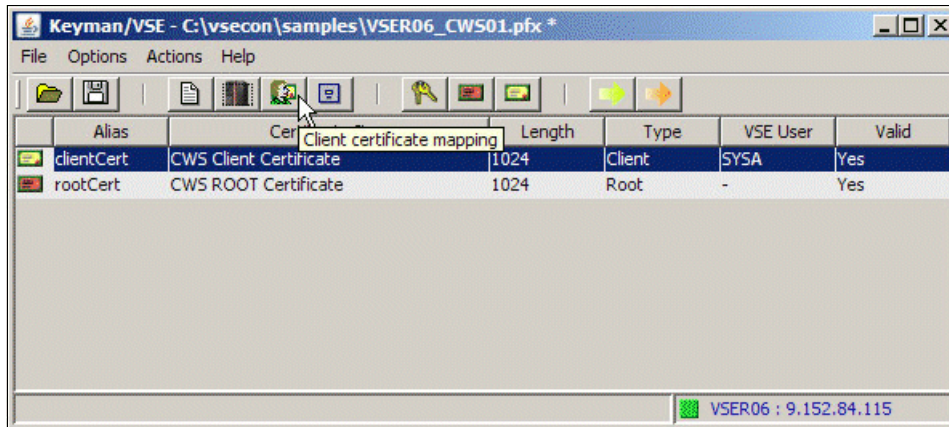


Figure 6-37 List of mapped certificates

A list of mapped client certificates opens, as shown in Figure 6-38.

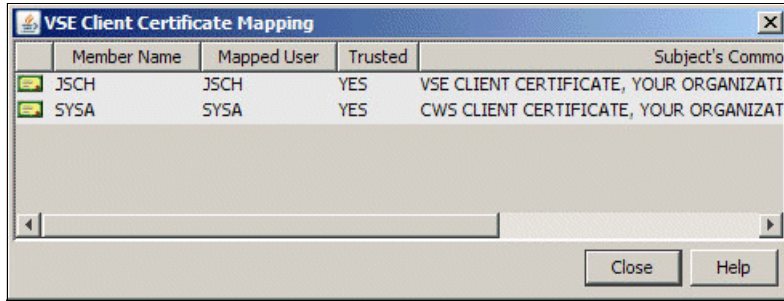


Figure 6-38 z/VSE client certificate mapping

All resource access attempts of this CWS client are now checked with the z/VSE security manager against the user's authorizations.

You can use the CICS EXTRACT CERTIFICATE command to get more information about a received client certificate, as described in *CICS Transaction Server for VSE/ESA Internet Guide*, SC34-5765.

Look for program DFH0WBCA, which is a COBOL sample program that shows the use of the EXTRACT CERTIFICATE function. Remove all code that you do not need from DFH0WBCA before using it.

## 6.7 Observations

This section describes several known problems and workarounds.

### 6.7.1 Abend AKEA in DFHSOSE

Review the symptom, possible reasons, and the workaround.

#### Symptom

The following error occurs when you are trying to connect to CWS with SSL:

```
F2 0173 SYSID=CIC1 APPLID=DBDCCICS
F2 0173 DFHS00001 DBDCCICS An abend (code 0C2/AKEA) has occurred at offset
      X'FFFF' in module DFHSOSE .
```

#### Possible reasons

The possible reasons are:

- ▶ PTFs UK30576 (z/VSE 4.1) or UK30575 (z/VSE 3.1) are missing
- ▶ An abend in DFHSOSE can be caused by previous CICS or TCP/IP related problems.

#### Workaround

If UK30576 / UK30575 is applied, it might help to clean the CICS local and global catalogs. You can use the job skeleton SKCICCLD of ICCF library 59 for this cleanup.



## 6.7.2 Abend code x'080C' in module DFHSO5E

Review the symptom and possible reason.

### Symptom

The following error occurs when you are trying to connect to CWS with SSL:

```
F2 0173 DFHS00002 DBDCCICS A severe error (code X'080C') has occurred in
      module DFHSO5E .
F2 0173 DFHME0116 DBDCCICS
      (Module:DFHMEME) CICS symptom string for message DFHS00002 is
      PIDS/564805400 LVLS/411 MS/DFHS00002 RIDS/DFHSO5E PTFS/UK20279
      PRCS/0000080C.
```

### Possible reasons

Your TCP/IP is not licensed for the cryptographic features on z/VSE. Check your LIBDEFs and ensure that the correct license key is accessed first.





## Connector security

z/VSE provides various connectors with which you can access z/VSE data or applications from a remote system, and access remote data and applications from z/VSE programs. The connectors use TCP/IP connections to transport access requests and data over the network.

Because the transported information and data can be security sensitive and confidential, the connectors must provide ways to secure and protect the remote access in terms of sign-in security, access control, and encryption of the transported data.

In this chapter, we describe the security options that are available and how they can be implemented for the following connector components:

- ▶ Java-based connector:
  - Sign-on security
  - Access protection for resources
  - Encryption using Secure Sockets Layer
- ▶ z/VSE script connector:
  - Encryption using Secure Sockets Layer
- ▶ Web services:
  - Transport Layer Security
  - Message layer security
- ▶ z/VSE Database Connector (DBCLI)
  - Sign-on security
  - DBCLI with SSL

This chapter includes the following topics:

- ▶ 7.1, “Java-based connector security” on page 252
- ▶ 7.2, “z/VSE script connector security” on page 255
- ▶ 7.3, “Web service security when using SOAP” on page 273
- ▶ 7.4, “z/VSE Database Connector (DBCLI)” on page 280

## 7.1 Java-based connector security

The Java-based connector provides remote access to z/VSE resources, such as VSAM files, VSE/POWER queues, librarian members, ICCF members, DL/1 data, and the console from any kind of Java application, including web applications.

The Java-based connector consists of the following parts:

- ▶ z/VSE Connector Client

This client part is implemented as a Java class library that provides a programming interface (API) for accessing z/VSE resources. Every supported z/VSE resource is represented by a Java class (bean) that can be used to access the corresponding resource to add, get, put, modify, delete, and so on.

- ▶ z/VSE Connector Server

This server part runs on z/VSE in a partition. It listens on a certain TCP/IP port and accepts connections from clients. When the client sends a request, the server receives it and runs it on behalf of the client.

Figure 7-1 shows these parts in a common environment.

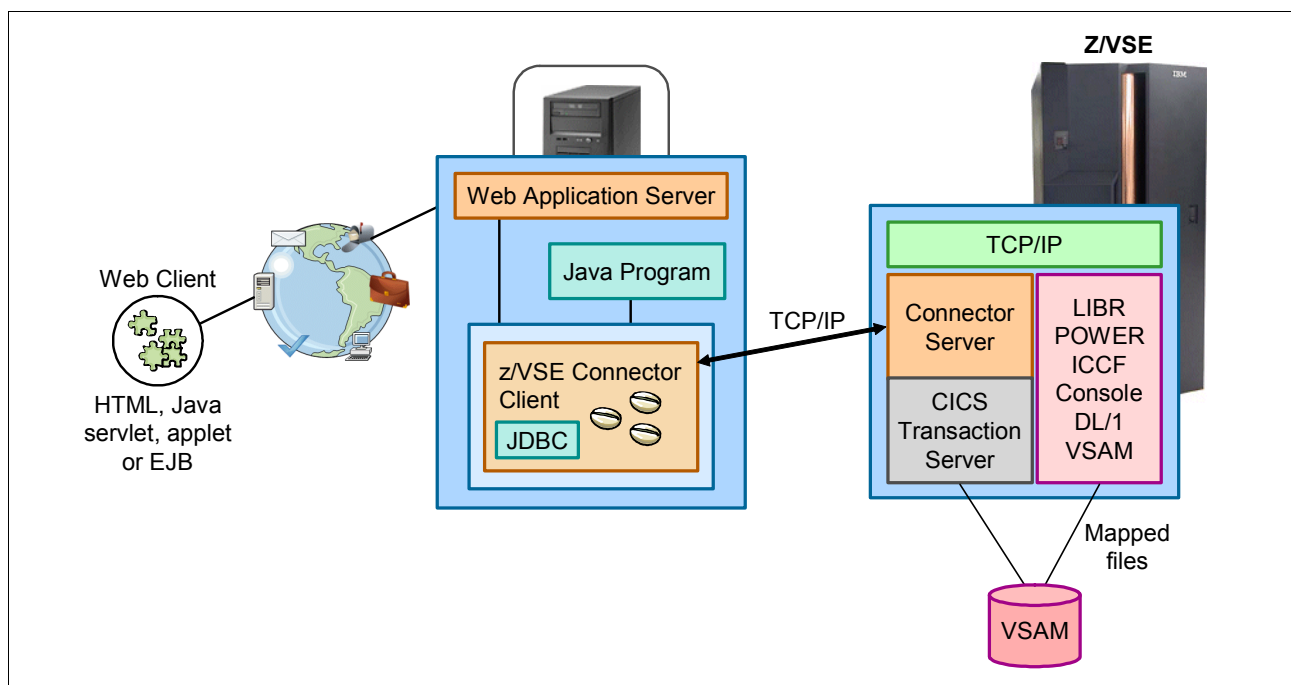


Figure 7-1 Java-based connector parts

The z/VSE Connector Client can also be deployed into a web application server, such as WebSphere Application Server as a J2C Resource Adapter. By doing so, it can be used from web applications, such as servlets, Java Server Pages (JSPs), Enterprise JavaBeans (EJBs), web services, or other kinds of web applications. The z/VSE Connector Resource Adapter also provides connection pooling capabilities.

For more information about the Java-based connector, see *z/VSE e-business Connectors, User's Guide*, SC33-8310.

## 7.1.1 Security features of the Java-based connector

Because the Java-based connector provides remote access to z/VSE resources, it must also provide a way to protect these resources from unauthorized access.

The Java-based connector provides the following types of security and access protection:

- ▶ Sign-on security based on user ID and password
- ▶ Access protection for resources through RACROUTE
- ▶ Special protection possibilities using FACILITY resources
- ▶ White list or black list based on IP address or user ID
- ▶ Transport Layer Security by Secure Sockets Layer (SSL/TLS)

These types of protection are described next.

### Sign-on security based on user ID and password

When a client connects to the server, the client must supply a valid user ID and password before it can access any resources. The user ID and password are checked against the active security manager, which can be the Basic Security Manager (BSM) or an external security manager.

The z/VSE Connector Server issues a RACROUTE call to perform the sign-on. If the sign-on was successful, the server runs every resource access under the user that signed in. This process ensures that, although the user accessed z/VSE remotely, the user can access only those resources for which the user has permissions.

The server also retrieves more information about the user ID, such as the user type; for example 1=administrator, 2=operator, 3=regular user. Based on the user type, the server rejects certain requests for non-privileged user IDs.

### Access protection for resources through RACROUTE

When the server receives a request to access a z/VSE resource from a client, it runs the resource access under the user that is associated with the connection from which the request was received. Before any resource access (such as read, write, update, or delete), the server issues a RACROUTE call to the active security manager to check whether that user is authorized to access the resource. If the user is not authorized to access that resource in the requested way, the server does not run the request and returns an appropriate error to the client.

### Special protection possibilities using FACILITY resources

Since z/VSE V4R1, the server also supports special security checks that are based on the resource class FACILITY. This support is activated by setting the security level to EXTENDED in the z/VSE Connector Server configuration. With this support, the security administrator can control, with the profile definition, the kind of resources a user can access. The server uses the following resource names of class FACILITY:

<b>VSE.CONNECTOR</b>	General access
<b>VSE.CONNECTOR.CONSOLE</b>	Console access
<b>VSE.CONNECTOR.ICCF</b>	ICCF access
<b>VSE.CONNECTOR.LIBR</b>	Librarian access
<b>VSE.CONNECTOR.POWER</b>	VSE/POWER access
<b>VSE.CONNECTOR.VSAM</b>	VSAM access

By giving a user the permission to access the FACILITY resource VSE.CONNECTOR.VSAM for read operations only, the user cannot modify or update any VSAM resource, regardless what permissions the user has on the specific VSAM data sets.

Each of these FACILITY resources protects an entire group of resources. By using VSE.CONNECTOR, a user can be permitted or rejected to access resources through the Java-based connector.

If no profile is defined for a certain FACILITY resource name that the server is using, access is permitted to the corresponding group of resources.

### **Configuring the level of security**

You can configure the level of security that the z/VSE Connector Server should use. Through the level, you can control which security checks the server is performing at sign-on and when accessing a resource.

The security level is defined in the main configuration member of the server (see skeleton SKVCSCFG in ICCF library 59). The following levels are available:

<b>EXTENDED</b>	Sign in, resource, user type, and FACILITY checking are done.
<b>FULL</b>	Sign in, resource and user type checking are done.
<b>RESOURCE</b>	Sign in and resource checking are done, but no user type.
<b>LOGON</b>	Only Sign in checking, no resource or user type.
<b>NO</b>	No security checks are done (setting this level is not recommended).

The default security level is FULL.

### **White list and black list based on IP address or user ID**

In addition to the security features, the z/VSE Connector Server provides a type of white list and black list where you can control which clients are allowed to establish a connection to the server. This listing is based on the IP address of the client that establishes a connection or the user ID.

By using these lists, you can allow or reject IP ranges or subnets, and individual IP addresses, to connect to the server. In addition, you can define which user IDs are allowed or denied to sign in remotely.

You can define the allowed or denied IP addresses or user IDs in the user security configuration member (see skeleton SKVCSUSR in ICCF library 59). The client or user is allowed to connect if its IP address matches at least one IP filter that is allowed to connect and its user ID matches at least one USER filter that is allowed to sign in. If its IP address matches one or multiple IP filters that are denied, or if its user ID matches one or multiple USER filters that are denied, the client is rejected. You can use wildcards at any place to build groups, ranges, or subnets.

Example 7-1 on page 255 shows that any client coming from an IP address that is in the 10.x.x.x subnet is allowed to connect, unless it is from subnet 10.4.x.x or it is the exact IP address 10.3.10.12. Any client that comes from a subnet other than 10.x.x.x is denied. If the client signs in with a user ID that starts with SYS or with B it is allowed to sign in, however, if it is BOBY, it is denied.

### Example 7-1 White list and black list using wildcards

```
* *****
* VSE CONNECTOR SERVER USER SECURITY CONFIGURATION MEMBER
* YOU CAN EITHER ALLOW OR DENY THE LOGON FOR A SPECIFIED
* USER OR IP OR GROUP OF USERS AND IP ADDRESSES.
* *****
* USERS FROM THESE IPS ARE ALLOWED TO LOGON
* UNCOMMENT THE SAMPLES AND MODIFY THEM
* *****
IP = 10.3.10.12,      LOGON = DENIED
IP = 10.4.*,         LOGON = DENIED
IP = 10.*,           LOGON = ALLOWED
* *****
* THESE USERS ARE ALLOWED TO LOGON
* UNCOMMENT THE SAMPLES AND MODIFY THEM
* *****
USER = BOBY,         LOGON = DENIED
USER = SYS*,         LOGON = ALLOWED
USER = B*,           LOGON = ALLOWED
```

These white lists and black lists do not affect other z/VSE functions; the lists affect only connection attempts that are made through the z/VSE Connector Server.

### Transport Layer Security by Secure Sockets Layer (SSL/TLS)

Because the Java-based connector can be used to access confidential data, it must provide a way to encrypt the data before it is sent over the network. This process is done by using Secure Sockets Layer (SSL) to encrypt and secure the connection between the client and the server.

The use of SSL is optional. If it is used, it must be configured by using the SSL configuration member. For more information, see the skeleton SKVCSSSL in ICCF library 59. There, you specify the SSL version to use, the name of the keyring library and the name of the SSL key to use, and what cipher suites are allowed.

The same or similar information must be specified on the client side inside the Java program that is using the z/VSE Connector Client API. Other than the specification of that SSL-specific information, the use of SSL is transparent to the applications or users.

To use SSL, you must create keys and certificates on the z/VSE side and the remote side. For more information, see Chapter 5, “Secure Sockets Layer with z/VSE” on page 197.

## 7.2 z/VSE script connector security

The Java-based connector provides direct access to the z/VSE host from any kind of Java program that is running on a Java platform, such as servlets, applets, and EJBs. You also can use the z/VSE script connector to access z/VSE host data from non-Java platforms. This feature is the main advantage of using the z/VSE script connector, although it can also be used to access z/VSE host data from most Java platforms.

The z/VSE script connector consists of the following components:

- ▶ A z/VSE script client that is running on a Java or non-Java platform, which can be one of the following components:
  - A user-written Java application; for example, a web service.
  - A user-written non-Java application; for example, a Windows C-program, a Windows CGI-program, or a COBOL application.
  - A batch or CICS program that is running on z/VSE.
  - An office product, such as a word-processing or spreadsheet program (for example, IBM Lotus® 1-2-3 or IBM Lotus Word Pro®) where a Visual Basic script is used to call a z/VSE script.
- ▶ The z/VSE script server that is running on the physical or logical middle tier of a 3-tier environment, which interprets and runs z/VSE script files.

The z/VSE script connector works as shown in the following process:

1. The z/VSE script client calls a z/VSE script to make a request for data that is stored on the z/VSE host. These z/VSE script batch files contain statements that are written with the z/VSE script language, which is a specialized programming language. The z/VSE script language can be used in any environment, even in Visual Basic scripts.
2. The z/VSE script server that is running on a Java-enabled platform reads, interprets, and translates, the z/VSE script file statements into z/VSE JavaBeans requests. The z/VSE script server uses the z/VSE JavaBeans to connect to the z/VSE Connector Server that is running on the z/VSE host, and to forward the z/VSE JavaBeans requests.
3. The z/VSE Connector Server accesses the required z/VSE data and functions, and sends the reply to the z/VSE script server.
4. The z/VSE script server converts the data to the format that the z/VSE script client can use, and returns the data to the z/VSE script client.

For more information about the z/VSE script connector, see *z/VSE e-business Connectors, User's Guide*, SC33-8310.

## 7.2.1 Security features of the z/VSE script connector

Because the z/VSE script connector provides remote access to z/VSE resources, it must provide a way to protect these resources from unauthorized access.

The following security features are available:

- ▶ Securing the connection between a z/VSE script client and the z/VSE script server
- ▶ Securing the connection between the z/VSE script server and the z/VSE Connector Server running on z/VSE

### Secure connection of z/VSE script client with z/VSE script server

A script client connects to the script server through a TCP/IP connection to send the name of the script to run, and input parameters it wants to pass to the script. When the script server runs the script, it sends back the output messages that the script produces over the TCP/IP connection.

Because both input parameters and output messages can contain confidential information, you might want to encrypt that information before it is transported over the network. Since z/VSE V4R2, the z/VSE script connector supports SSL and TLS for the connection between the client and the server. The use of SSL or TLS is optional.



To use SSL or TLS, you must create some keys and certificates on the script client side and the script server side. If the script client runs on z/VSE, you also must set up SSL on z/VSE. You can use the Keyman/VSE tool to do so. For more information, see Chapter 5, “Secure Sockets Layer with z/VSE” on page 197.

You configure the settings including the use of SSL or TLS for the connection between the z/VSE script client and the z/VSE script server in the file `VSEScriptServer.properties`.

### **Secure connection of z/VSE script server with z/VSE Connector Server**

When a z/VSE script is run by the z/VSE script server, it can access resources on z/VSE. To do that, the z/VSE script server communicates with the z/VSE Connector Server that is running on z/VSE. Because these resources might contain confidential information, you might want to protect the access to it.

To access resources on z/VSE, the z/VSE script server connects to the z/VSE Connector Server and signs in with a user ID and password. The z/VSE Connector Server verifies whether that user ID is allowed to access the requested resources. For more information, see 7.1, “Java-based connector security” on page 252.

You also might want to encrypt that information before it is transported over the network. Since z/VSE V4.2, the z/VSE script connector supports SSL and TLS for the connection between the z/VSE script server and the z/VSE Connector Server. The use of SSL or TLS is optional.

To use SSL or TLS, you must create keys and certificates on the script server side and the z/VSE Connector Server side. You can use the Keyman/VSE tool to do so. For more information, see Chapter 5, “Secure Sockets Layer with z/VSE” on page 197.

You configure the settings including the use of SSL or TLS for the connection between the z/VSE script server and the z/VSE Connector Server in the file `Connections.properties`.

## **7.2.2 Non-SSL setup with client on workstation**

In the following sections, we describe two scenarios with the z/VSE script connector where SSL is not used. Later, these setups are modified to use SSL. In the first scenario, the z/VSE script client runs on a workstation; in the second scenario, it runs on z/VSE. The z/VSE script server always runs on a Java-enabled platform.

Figure 7-2 on page 258 shows a three-tier scenario with a non-Java z/VSE script client, which can be a Microsoft or Lotus office application. The z/VSE script client communicates with the z/VSE script server that is running on a Java-enabled middle-tier platform. The z/VSE script server uses the z/VSE Connector Client to transfer data to and from z/VSE by way of the z/VSE Connector Server.

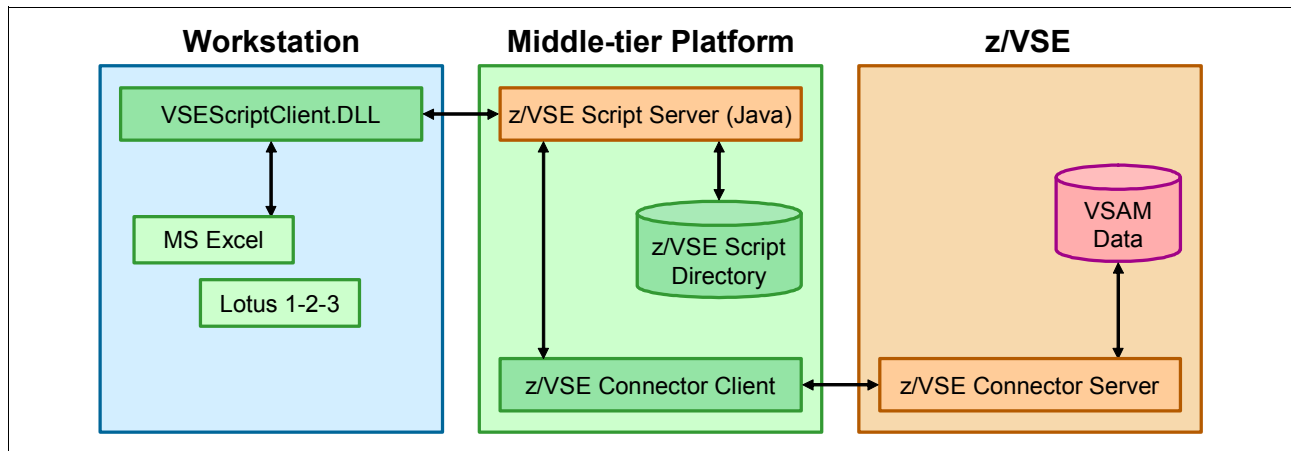


Figure 7-2 Three-tier solution with a non-Java z/VSE script client

### Setting up z/VSE script server

After installing the z/VSE script server, you must set up the following properties files:

- ▶ The server's properties file (VSEScriptServer.properties)
- ▶ The server's connection properties file (Connections.properties)

Normally, no changes are necessary in the server properties file.

In the connection properties file, you must define the IP address of your z/VSE system and the port number where the z/VSE Connector Server listens, as shown in Example 7-2.

Example 7-2 z/VSE script server connection properties

```

connection.1.name=z9_LPAR4
connection.1.ip=9.152.85.58
connection.1.port=2893
connection.1.userid=JSCH
connection.1.password=myspasswd
connection.timeout=100

```

**Note:** During the initial startup of the z/VSE script server, the password is encrypted and stored by using the property `connection.n.encpassword`.

### Setting up a z/VSE script client

The following steps must be completed to enable a non-Java application that is running as a z/VSE script client:

1. Make the z/VSE script client DLL accessible by the client application.
2. Define the z/VSE script in the client application.

#### ***z/VSE script client DLL***

VSEScriptClient.DLL must be accessible by the office application. In our test setup, we copied the DLL to `c:\windows`.

**Note:** The VSEScriptClient.DLL is not SSL-enabled.

### Setting up client script in MS Excel

In our setup, we use Microsoft Excel as the client application. Complete the following steps to include the IBM-provided sample script into MS Excel<sup>1</sup>:

1. Create an MS Excel worksheet and open it.

Ensure that the Control Toolbox is visible (see Figure 7-3).

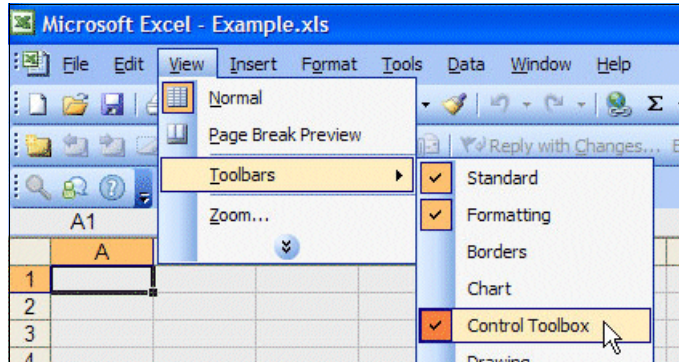


Figure 7-3 Select Control Toolbox

2. Insert a new Command Button, as shown in Figure 7-4.

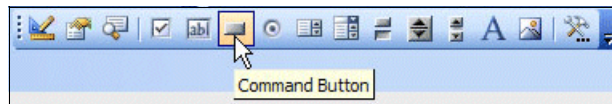


Figure 7-4 Insert new Command Button

3. Right-click the button and select **View Code** (Figure 7-5).

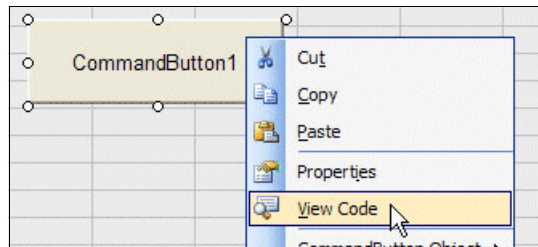


Figure 7-5 Select View Code

The Visual Basic design sheet opens. Ensure that the two drop-down list boxes show CommandButton1 and Click, as shown in Figure 7-6 on page 260.

<sup>1</sup> For an example with Lotus 1-2-3, see *z/VSE e-business Connectors, User's Guide*, SC33-8310.

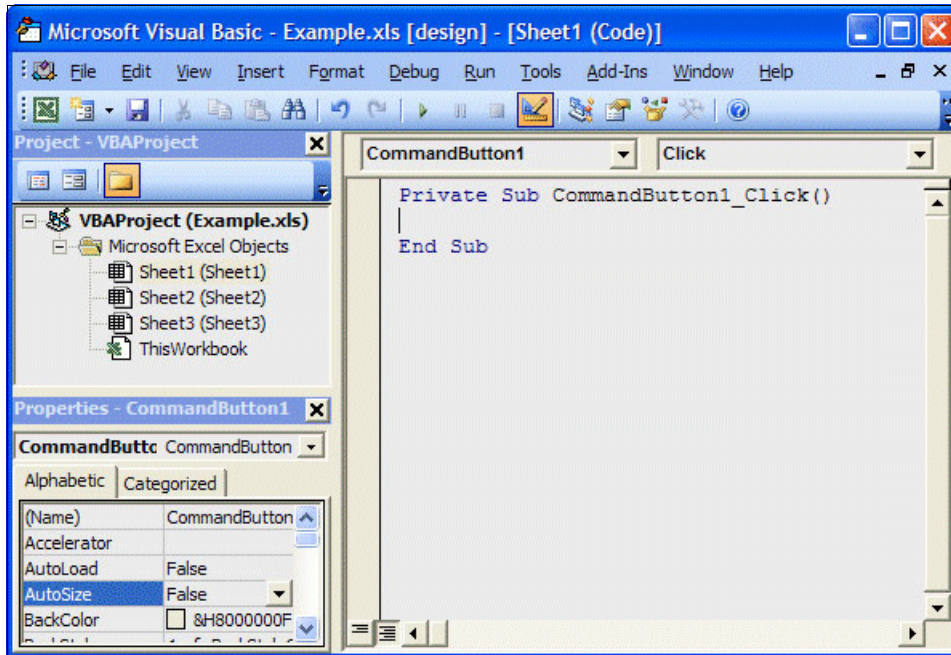


Figure 7-6 Open Visual Basic design sheet

4. Paste the body of the Visual Basic sample script VSEScriptClient.bas that is in the z/VSE script server installation directory into the text area. The result should resemble the sheet that is shown in Figure 7-7.

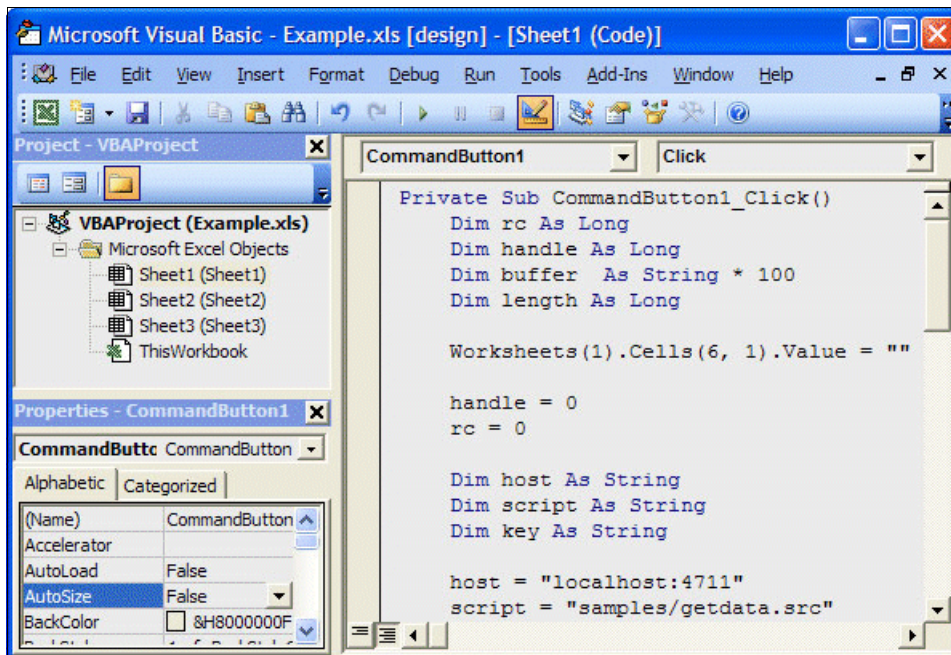


Figure 7-7 Complete script in design sheet

This code assumes that you are running the z/VSE script server on the same workstation as the z/VSE script client. If not, you must change the following line:

```
host = "localhost:4711"
```



It also assumes that the server script is in the scripts/samples directory of your script server installation directory, as shown in the following example:

```
script = "samples/getdata.src"
```

If not, replace it with the full path name.

5. Select **(General)** and **(Declarations)** in the two drop-down list boxes (see Figure 7-8).

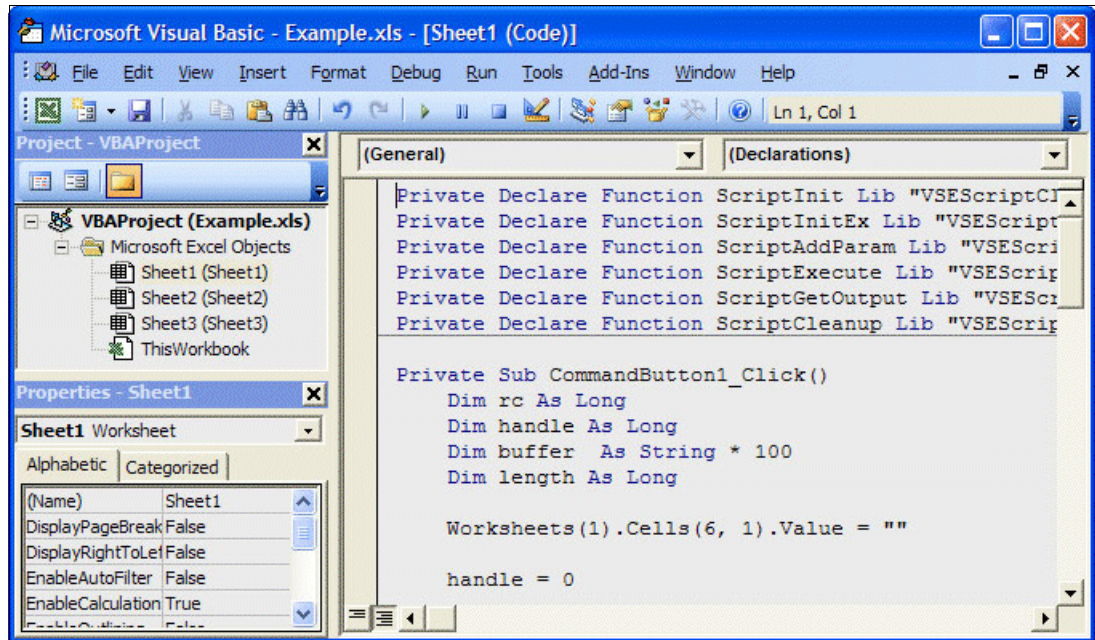


Figure 7-8 Copy variable declarations from the sample script

Then, copy the variable declarations from the provided sample script.

**Note:** The Visual Basic code that is provided with the z/VSE script server installation did not work with MS Excel 2003. We removed the following first line of the sample code:

```
Attribute VB_Name = "Module1"
```

We also changed the declarations from Public to Private. For more information about the error, see “Visual Basic script syntax error” on page 271.

Save your changes and exit.

### **Setting up the server script**

In our setup, we used the provided sample server script GetData.src, which is in the scripts/samples directory of the script server installation directory.

Here, you must change some parameters according to your environment. Open the script with a text editor and ensure that you have the right host and file name defined.

For example:

```
String host = "z9_LPAR4";  
String file = "VSESP.USER.CATALOG\\VSAM.CONN.SAMPLE.DATA\\USED CARS";
```

For more information about how to define the sample VSAM cluster and fill it with sample data, see “Setting up sample VSAM data”.

Save your changes and exit.

### **Start z/VSE Connector Server on z/VSE**

Now, ensure that the z/VSE Connector Server is started on z/VSE in non-SSL mode (job STARTVCS).

### **Setting up sample VSAM data**

Some sample VSAM data is provided with job SKVSSAMP in ICCF library 59. Job SKVSSAMP defines the new VSAM file VSAM.CONN.SAMPLE.DATA in the VSESP.USER.CATALOG. It automatically uses the IDCAMS RECMAP command to define a VSAM map that contains the field names of the sample data.

Also, the following statements are in job SKVSSAMP:

```
// LIBDEF *,SEARCH=(PRD1.BASE)
// EXEC VSAMSMPD,SIZE=AUTO
```

VSAMSMPD is a small IBM-provided program that fills the VSAM file with sample records. The VSAM record layout matches the structure that is used in the client script.

### **Start the z/VSE script server locally**

Now, start the z/VSE script server on your workstation. Remember that the client script contains the following line:

```
host = "localhost:4711"
```

This line assumes that the script server is running locally on your workstation.

Double-click the runserver.bat file in the script connector installation directory or start the server from the Start Programs menu.

The server displays some output similar to the output that is shown in Example 7-3.

#### *Example 7-3 z/VSE script server output*

---

```
VSEScriptServer starting...
Dec 8, 2009 2:29:08 PM (1) - Listening socket created on port 4711
Dec 8, 2009 2:29:08 PM (1) - Starting ConnectionManager
Dec 8, 2009 2:29:08 PM (1) - Connection 'Z9_LPAR4' has been bound to '9.152.85.58'
Dec 8, 2009 2:29:08 PM (1) - Connection 'DEM02' has been bound to '9.12.34.78'
Enter 'quit' to stop the server
Dec 8, 2009 2:29:08 PM (8) - Waiting for connections...
```

---

### **Running the client script**

You can now run the script by clicking the command button in the Excel worksheet. The following line in the script reads the key of the VSAM data from cell (1,1):

```
key = Worksheets(1).Cells(1, 1).Value
```

Enter a valid key in cell (1,1) and click **CommandButton1**, as shown in Figure 7-9.

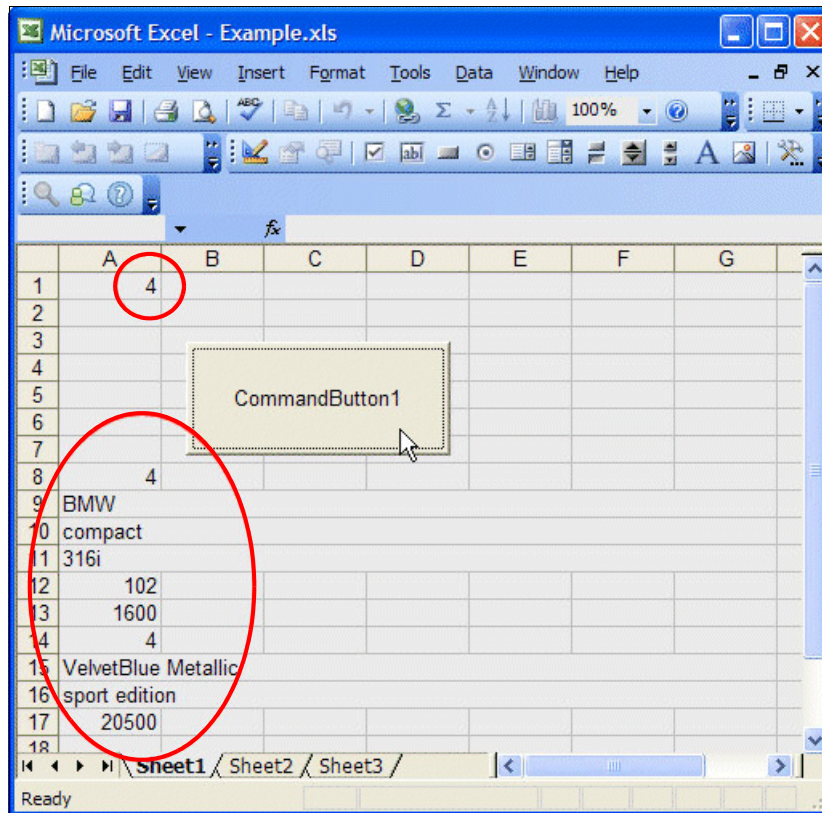


Figure 7-9 z/VSE script output in the spreadsheet

Now, the spreadsheet contains the transferred VSAM data.

The script server shows the status of the z/VSE connection (see Example 7-4).

*Example 7-4 z/VSE script server connection status*

---

```
Dec 8, 2009 4:07:18 PM (8) - Waiting for connections...
Dec 8, 2009 4:23:24 PM (8) - Client connection request from 127.0.0.1
Dec 8, 2009 4:23:24 PM (22) - Client has been accepted.
Dec 8, 2009 4:23:24 PM (22) - Connection has been accepted from 127.0.0.1
Dec 8, 2009 4:23:24 PM (22) - Using default system codepage.
Dec 8, 2009 4:23:24 PM (22) - Executing script 'samples/getdata.src'
Dec 8, 2009 4:23:24 PM (22) - Connection has been terminated from 127.0.0.1
Dec 8, 2009 4:23:24 PM (22) - Client has been disconnected.
```

---

You can run the **STATUS** command to see the connection status in the script server command line (see Example 7-5).

*Example 7-5 z/VSE script server status command and output*

---

```
status
Status Information:

Clients:

    0 clients connected.
```

ConnectionManager:

1. Connection: Z9\_LPAR4 (unused, current timeout=96)

Number of used Connections: 0

Number of unused Connections: 1

---

In Example 7-5 on page 263, the output shows that the server caches the z/VSE connection for further reuse; for example, when pressing the button again, the previously established connection is reused.

In the next section, the same script is run on z/VSE by using the z/VSE-based script client IESSCBAT.

### 7.2.3 Non-SSL setup with a client on z/VSE

This section describes the second scenario, where the script client runs on z/VSE. This scenario is intended for controlling any processes on the platform where the script server runs; for example, access to a local database. The point is that you can start the process from z/VSE, similar to the REXEC function that is provided by TCP/IP. However, it is even possible to retrieve z/VSE data by way of the Java-based connector.

Figure 7-10 shows the following z/VSE clients:

- ▶ IESSCBAT, which runs in batch.
- ▶ IESSCCIC, which runs in CICS, but is not considered here.

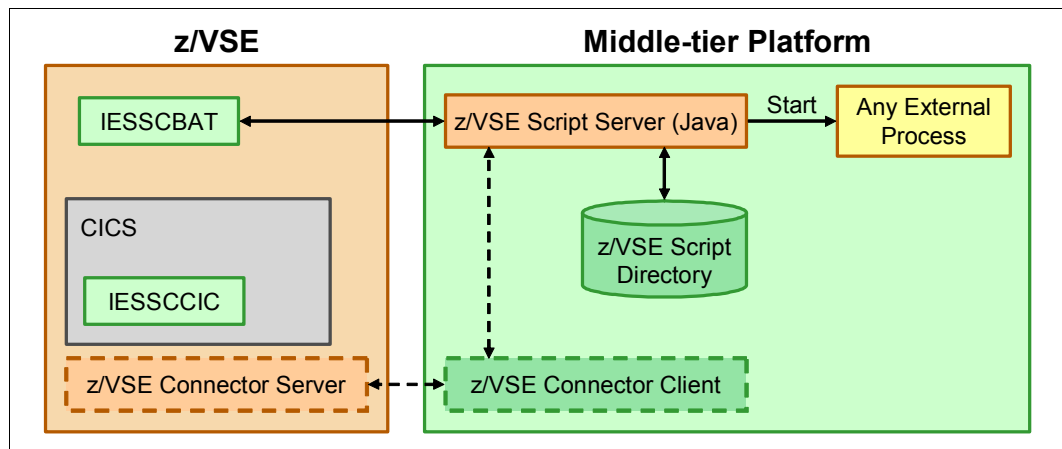


Figure 7-10 Non-SSL setup with client on z/VSE

In the following sections, we assume that the basic setup is complete. The VSAM file is set up, and the z/VSE Connector Server and z/VSE script server are running.



## Setting up IESSCBAT

You can find various examples of how to use IESSCBAT in *z/VSE e-business Connectors, User's Guide*, SC33-8310. We used the JCL from Example 7-6 in our test environment.

### Example 7-6 IESSCBAT JCL

---

```
* $$ JOB JNM=IESSCBAT,CLASS=A,DISP=D
// JOB IESSCBAT
// EXEC IESSCBAT,PARM='CODEPAGE=CP1047 SHOWERROR=YES'
9.152.222.37:4711
SAMPLES/GETDATA.SRC
7
/*
/&
* $$ EOJ
```

---

In Example 7-6, the IP address is the IP address of the workstation that is running the z/VSE script server. The IP address is followed by the name of the server script to be started. The script name is followed by the input argument (key for VSAM file).

**Note:** The server script GETDATA.SRC is still on the workstation, although the script is started from z/VSE.

The job output in Example 7-7 shows the retrieved data.

### Example 7-7 IESSCBAT output

---

```
// JOB IESSCBAT
// LIBDEF *,SEARCH=PRD1.BASE
// EXEC IESSCBAT,PARM='CODEPAGE=CP1047 SHOWERROR=YES'
1S54I PHASE IESSCBAT IS TO BE FETCHED FROM PRD1.BASE
7
Ford
Escort
ZX2 2 Door Coupe
150
2000
6
White
Sp.Seats,Zetec Eng.
15715
1S55I LAST RETURN CODE WAS 0000
EOJ IESSCBAT MAX.RETURN CODE=0000
```

---

## Set up IESSCCIC

For more information about how to use the IBM provided CICS client IESSCCIC, see *z/VSE e-business Connectors, User's Guide*, SC33-8310.

## 7.2.4 General SSL setup

The following sections describe the basic SSL setup for the z/VSE script connector with server and client authentication.

## Overview of SSL connections

Figure 7-11 shows which connections can be secured with SSL. Also shown are two z/VSE systems, where one z/VSE runs the script client and the other z/VSE is used to retrieve data by way of the z/VSE Connector Client. It also is possible that the script client and the z/VSE Connector Client run on the same z/VSE system.

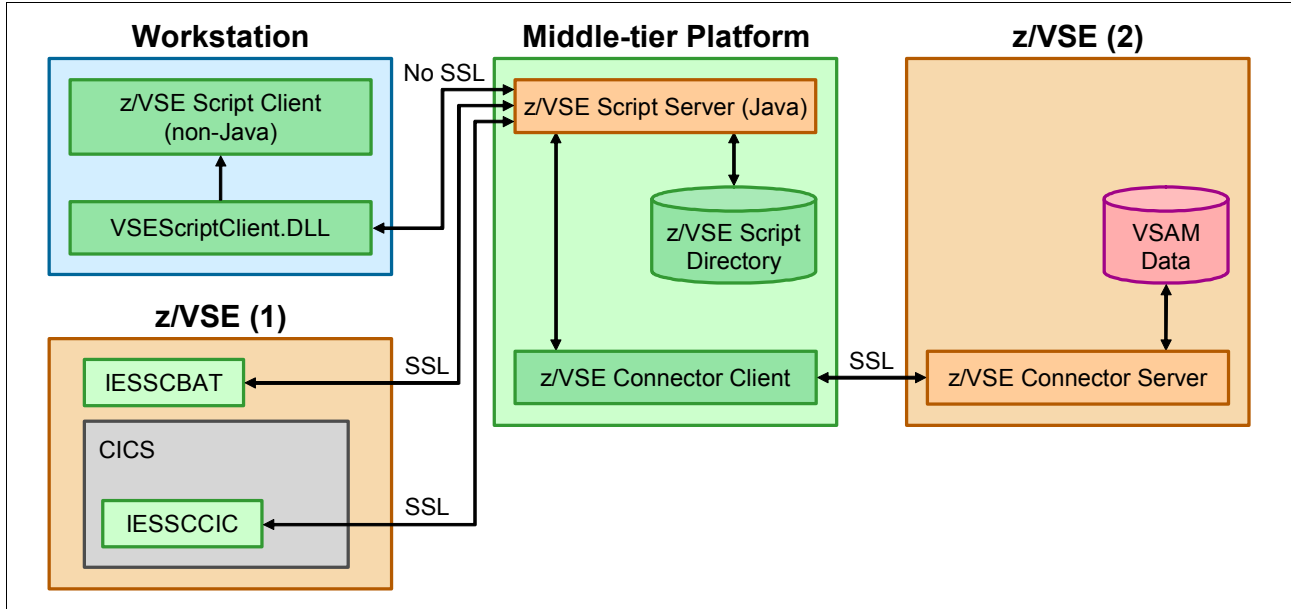


Figure 7-11 Overview of SSL connections

The connection between VSEScriptClient.DLL and the z/VSE script server cannot be SSL-enabled.

In the following section, we focus on the SSL setup between IESSCBAT and the z/VSE script server.

## Overview of keys and certificates

Figure 7-12 shows how the keys and certificates must be stored on the various platforms.

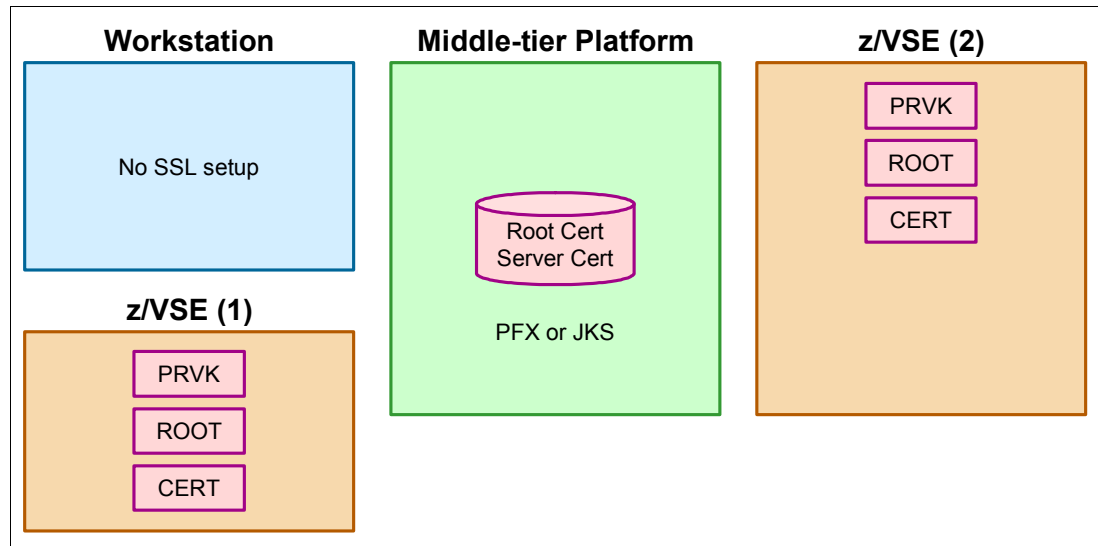


Figure 7-12 Overview of keys and certificates

No keys and certificates are necessary on the client platform, because VSEScriptClient.DLL is not SSL-enabled.

A keyring file (PFX or JKS) is created with the Keyman/VSE tool on the middle-tier platform where the script server runs. After uploading the RSA key to the z/VSE systems, it is no longer needed in the keyring file.

The RSA key, the z/VSE server certificate, and the root certificate are transferred to the z/VSE systems.

The next steps include:

- ▶ Generating the server key and certificate.
- ▶ Creating the local keyring file (PFX or JKS).
- ▶ Uploading the certificate items to z/VSE.

For more information about these steps, see "Creating the keys and certificates" on page 378.

### 7.2.5 SSL setup with client on z/VSE

This section describes how to set up the keys and certificates for SSL between z/VSE batch client IESSCBAT and the z/VSE script server.

Figure 7-13 on page 268 shows the z/VSE batch client and the z/VSE script server together with the related parts.

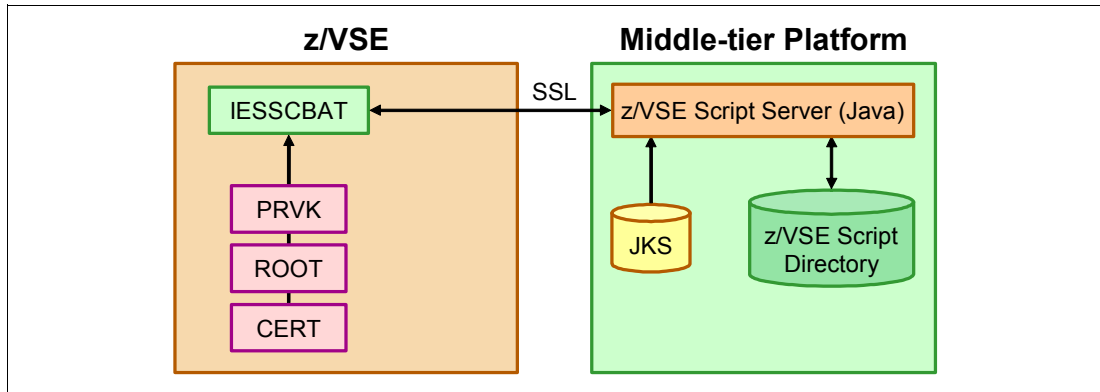


Figure 7-13 SSL setup for IESSCBAT

### Setting up IESSCBAT for SSL

The SSL parameters for IESSCBAT are directly specified in the JCL. With the JCL that is shown in Example 7-8, IESSCBAT assumes that the script server is running in SSL server authentication mode. For more information about how to enable IESSCBAT for client authentication, see “Using SSL client authentication” on page 269.

Example 7-8 IESSCBAT job with SSL

```
* $$ JOB JNM=IESSCBAT,CLASS=A,DISP=D
// JOB IESSCBAT
// EXEC IESSCBAT,PARM='CODEPAGE=CP1047 SHOWERROR=YES SSL=YES'
9.152.222.37:4711
SSL30
CRYPTO.KEYRING
SCRIPT01
0A2F
86440
SAMPLES/GETDATA.SRC
7
/*
/&
* $$ EOJ
```

The keyring member name SCRIPT01 must match the name you used before when uploading the key and certificates to z/VSE.

### Setting up the z/VSE script server for SSL

To enable SSL for the z/VSE script server, you must modify the `VSEScriptServer.properties` file. Uncomment the following lines and specify the appropriate values for your environment:

```
sslversion=SSL
keyringfile=c:\\vsecon\\samples\\z9_LPAR4_SCRIPT01.jks
keyringpwd=ypasswd
ciphersuites=SSL_RSA_WITH_3DES_EDE_CBC_SHA,TLS_RSA_WITH_AES_128_CBC_SHA
```

If you leave the comment for cipher suites, all available cipher suites are enabled.

Cipher suites 01, 02, 08, 09, and 62 provide only weak encryption; therefore, they should no longer be used in today’s computing environments.

By default, Java does not allow the use of AES-256. If you want to use cipher suite 35, follow the instructions that are provided in “Using encryption with AES-256” on page 212.

Now, restart the z/VSE script server. You can now run IESSCBAT by using SSL.

## Testing the connection

When now running IESSCBAT again, output that is similar to the output that is shown in Example 7-9 should be produced.

### *Example 7-9 Output from IESSCBAT*

---

```
// JOB IESSCBAT
// LIBDEF *,SEARCH=PRD1.BASE
// EXEC IESSCBAT,PARM='CODEPAGE=CP1047 SHOWERROR=YES SSL=YES'
1S54I  PHASE IESSCBAT IS TO BE FETCHED FROM PRD1.BASE
7
Ford
Escort
ZX2 2 Door Coupe
150
2000
6
White
Sp.Seats,Zetec Eng.
15715
1S55I  LAST RETURN CODE WAS 0000
```

---

Example 7-10 shows the z/VSE script server output.

### *Example 7-10 Output from z/VSE Script Server*

---

```
Dec 9, 2009 4:26:10 PM (8) - Waiting for connections...
Dec 9, 2009 4:26:16 PM (8) - Client connection request from 9.152.85.58
Dec 9, 2009 4:26:16 PM (10) - Client has been accepted.
Dec 9, 2009 4:26:16 PM (10) - Connection has been accepted from 9.152.85.58
Dec 9, 2009 4:26:16 PM (10) - Session established with cipher suite:
TLS_RSA_WITH_AES_128_CBC_SHA
Dec 9, 2009 4:26:16 PM (10) - Using script-requested codepage CP1047
Dec 9, 2009 4:26:16 PM (10) - Executing script 'SAMPLES/GETDATA.SRC'
Dec 9, 2009 4:26:16 PM (10) - Created new Connection for 'Z9_LPAR4 (unused,
current timeout=100) '
Dec 9, 2009 4:26:16 PM (10) - Connection has been terminated from 9.152.85.58
Dec 9, 2009 4:26:16 PM (10) - Client has been disconnected.
```

---

Depending on the specified list of cipher suites in IESSCBAT, some error messages can occur. For more information, see “SSLHandshakeException: Invalid padding” on page 272.

## Using SSL client authentication

If you want to use client authentication, you must modify the server’s properties file (VSEScriptServer.properties). Activate the following parameter:

```
clientauthentication=true
```

Then, change the IESSCBAT job as follows:

```
// EXEC IESSCBAT,PARM='CODEPAGE=CP1047 SHOWERROR=YES SSL=YES          X
                          ALLOWCLIENTAUTH=YES'
```

Now, restart the z/VSE script server.

## Testing the connection with client authentication

When running IESSCBAT with client authentication, the server output informs you about the client, as shown in Example 7-11.

### *Example 7-11 z/VSE Script Server output with SSL*

---

```
Dec 10, 2009 10:35:17 AM (8) - Waiting for connections...
Dec 10, 2009 10:35:35 AM (8) - Client connection request from 9.152.85.58
Dec 10, 2009 10:35:35 AM (10) - Client has been accepted.
Dec 10, 2009 10:35:35 AM (10) - Connection has been accepted from 9.152.85.58
Dec 10, 2009 10:35:36 AM (10) - Session established with cipher suite:
TLS_RSA_WITH_AES_128_CBC_SHA
Dec 10, 2009 10:35:36 AM (10) - Client Certificate:
Dec 10, 2009 10:35:36 AM (10) - Subject: CN=VSE Server Certificate,
OU=Development, O=IBM, L=Boeblingen, ST=Baden-Wuerttemberg, C=DE,
EMAILADDRESS=zvse@de.ibm.com
Dec 10, 2009 10:35:36 AM (10) - Issuer: CN=VSE ROOT Certificate, OU=IBM
Germany, O=IBM, L=Boeblingen, ST=Baden-Wuerttemberg, C=DE,
EMAILADDRESS=zvse@de.ibm.com
Dec 10, 2009 10:35:36 AM (10) - Serial No: 1936789053
Dec 10, 2009 10:35:36 AM (10) - Valid from: Wed Dec 09 13:36:46 CET 2009
Dec 10, 2009 10:35:36 AM (10) - Valid to: Sat Mar 13 13:36:46 CET 2010
Dec 10, 2009 10:35:36 AM (10) - Valid: true
Dec 10, 2009 10:35:37 AM (10) - Using script-requested codepage CP1047
Dec 10, 2009 10:35:37 AM (10) - Executing script 'SAMPLES/GETDATA.SRC'
Dec 10, 2009 10:35:37 AM (10) - Created new Connection for 'Z9_LPAR4 (unused,
current timeout=100)'
Dec 10, 2009 10:35:37 AM (10) - Connection has been terminated from 9.152.85.58
Dec 10, 2009 10:35:37 AM (10) - Client has been disconnected.
Dec 10, 2009 10:37:17 AM (9) - Destroyed Connection for 'Z9_LPAR4 (unused, current
timeout=0)'
```

---

## Client authentication with multiple z/VSE clients

When you use multiple z/VSE clients, you must create a unique keyring for each involved z/VSE system. The JKS keystore on the script server side must contain copies of all root certificates that are stored on the different z/VSE systems. You can delete all RSA keys and z/VSE certificates (“yellow” certificates in Keyman/VSE) from the JKS file.

Figure 7-14 on page 271 shows the setup.

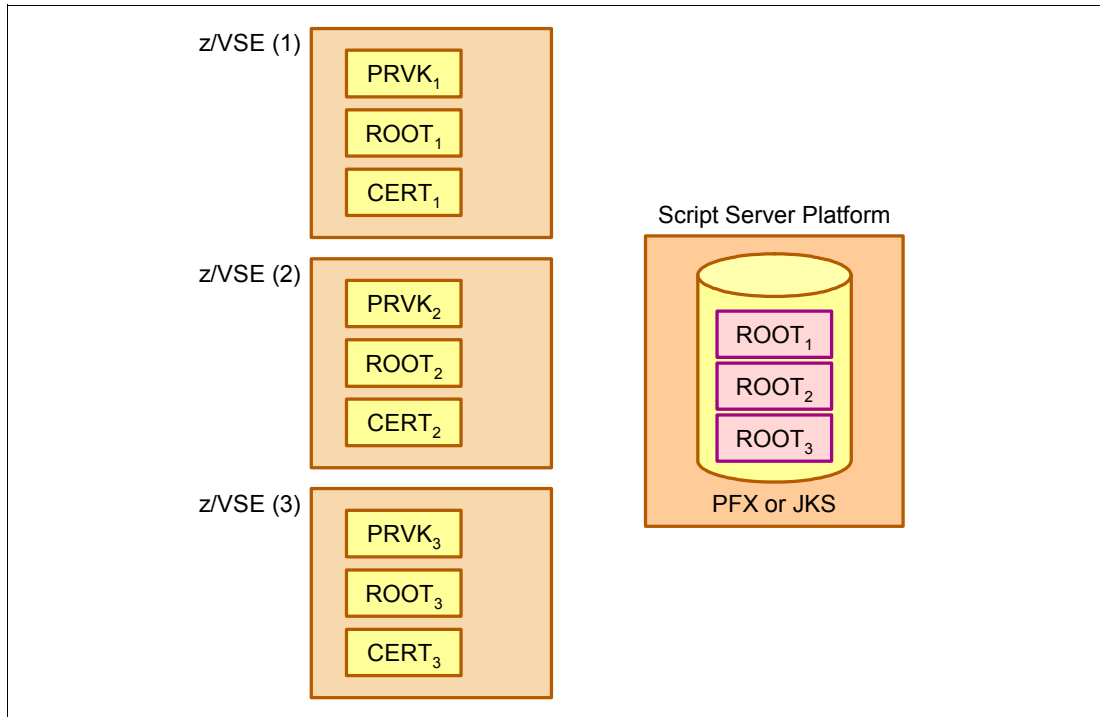


Figure 7-14 SSL setup with multiple z/VSE systems

You can now run IESSCBAT on each z/VSE system by using its own keyring, and thus, its own unique client certificate. When you remove a root certificate from the JKS file, the related z/VSE client cannot connect.

If a z/VSE client certificate (stored in the CERT member on z/VSE) cannot be verified on the server side, because the related root certificate was deleted from the JKS file, error messages occur, as shown in Example 7-12.

*Example 7-12 SSL handshake error*

---

```

Dec 23, 2009 3:35:05 PM (11) - Connection has been accepted from 9.152.85.58
Dec 23, 2009 3:35:05 PM (11) - Error during SSL handshake:
javax.net.ssl.SSLHandshakeException: sun.security.validator.ValidatorException:
PKIX path building failed:
sun.security.provider.certpath.SunCertPathBuilderException: unable to find
  valid certification path to requested target
Dec 23, 2009 3:35:05 PM (11) - Connection has been terminated from 9.152.85.58
Dec 23, 2009 3:35:05 PM (11) - Client has been disconnected.

```

---

## 7.2.6 Observations

This section describes some errors that occurred in our test setup.

### Visual Basic script syntax error

The symptom and possible reason are listed.

#### **Symptom**

MS Excel displays an error message, as shown in Figure 7-15.

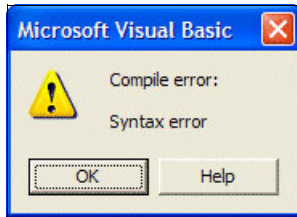


Figure 7-15 Visual Basic script syntax error

### **Possible reason**

The IBM-provided Visual Basic client script did not work with the MS Excel version that we were using. A compile error occurred, which was caused by the first line in the declarations part.

The following line was not recognized:

```
Attribute VB_Name = "Module1"
```

After removing this line, the error message that is shown in Figure 7-16 was displayed.

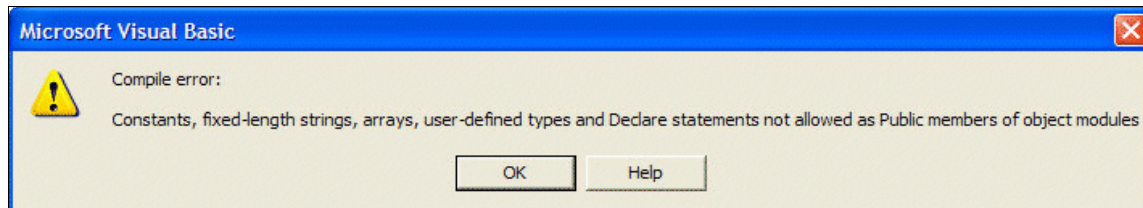


Figure 7-16 Visual Basic script compile error

This error can be resolved by defining the routines as private. For more information, see Figure 7-8 on page 261.

### **SSLException when starting z/VSE script server**

The symptom and possible reason are listed.

#### **Symptom**

The z/VSE script server shows the following error:

```
Dec 9, 2009 1:37:41 PM (8) - Error: javax.net.ssl.SSLException: No available certificate or key corresponds to the SSL cipher suites which are enabled.
```

#### **Possible reason**

The local keyring file was stored in the PFX file format, which cannot be read correctly by the Java runtime environment from Sun Microsystems. The use of the JKS file format instead solves the problem.

### **SSLHandshakeException: Invalid padding**

The symptom and possible reason are listed.

#### **Symptom**

The z/VSE script server shows:

```
Dec 9, 2009 4:27:59 PM (10) - Error during SSL handshake: javax.net.ssl.SSLHandshakeException: Invalid padding
```



### **Possible reason**

IESSCBAT used cipher suites 08090A2F35. Trying several combinations of cipher suites showed that cipher suite 08 causes the problem. Removing 08 solves the problem. At the moment, we do not know the reason why cipher suite 08 fails. Suppressing the use of hardware instructions (CPACF) with a \$SOCKOPT phase did not help.

However, as cipher suite 08 uses 40-bit DES, it should no longer be used.

### **SSL client authentication does not work**

The symptom and possible reason are listed.

#### **Symptom**

SSL client authentication does not work with an IBM or Sun Java runtime environment. The z/VSE script server shows:

```
Dec 9, 2009 4:09:24 PM (8) - Error during SSL handshake:  
javax.net.ssl.SSLHandshakeException: null cert chain
```

The IESSCBAT job ends with rc = 14 (see Example 7-13).

#### *Example 7-13 SSL error with client authentication*

---

```
// JOB IESSCBAT  
// LIBDEF *,SEARCH=PRD1.BASE  
// OPTION DUMP,NOSYSDMP  
// UPSI 1  
// EXEC IESSCBAT,PARM='CODEPAGE=CP1047 SHOWERROR=YES SSL=YES'  
1S54I PHASE IESSCBAT IS TO BE FETCHED FROM PRD1.BASE  
VSEScriptClient SSL Error: gsk_secure_soc_init: -3100  
Error: ScriptInit: SSL handshake failed.  
1S55I LAST RETURN CODE WAS 0014
```

---

#### **Possible reason:**

Parameter ALLOWCLIENTAUTH=YES was not specified in IESSCBAT.

## **7.2.7 Debugging hints**

If an error occurs, you can get useful trace information on z/VSE and on your workstation. For more information, see “Debugging SSL/TLS connections” on page 218.

## **7.3 Web service security when using SOAP**

SOAP is a standard, XML-based protocol that allows applications to exchange information over the internet through HTTP.

XML is a universal format that is used for structured documents and data on the web. It is independent of the web client’s operating system platform and the programming language that is used. HTTP is supported by all internet web browsers and servers.

SOAP combines the benefits of XML and HTTP into one standard application protocol. As a result, you can send information to and receive it from various platforms.

By using web browsers, you can view information that is contained on websites. However, by using SOAP, you can also perform the following tasks:

- ▶ Combine the contents of different websites and services.
- ▶ Generate a complete view of all the relevant information.

z/VSE supports the SOAP protocol; therefore, it allows you to implement web services.

An example of the use of SOAP might be when a travel agent requires a combined view of the web services that cover hotel reservation, flight booking, and car rental. After the travel agent enters the required data, all three web services from the three providers are processed in one transparent step. This process is an example of how a *business-to-business* (B2B) relationship can be implemented.

You do not usually have to be concerned with the tagging that is described here because it is automatically generated by the following components:

- ▶ The SOAP client, and then converted to native data by the SOAP server
- ▶ The SOAP server, and then converted to native data by the SOAP client-processor

However, you might require knowledge of the SOAP tagging for debugging purposes. The information in this section provides only an overview.

A SOAP message is a standard XML document containing the following main parts:

- ▶ The SOAP envelope, which defines the content of the message.
- ▶ The SOAP header (optional), which contains header information.
- ▶ The SOAP body, which contains call and reply information.

The following types of elements are used for the SOAP message:

- ▶ The `<Envelope>` element is the root element of a SOAP message, and defines the XML document to be a SOAP message.
- ▶ The `<Header>` element can be used to include more application-specific information about the SOAP message or security-specific information. The information here is user-defined. For example, it might be used to define the language that is used for the message.
- ▶ The `<Body>` element is used to define the message.
- ▶ The `<Fault>` element can be optionally used within the `<Body>` element. It is used to supply information about any errors that might occur when the SOAP message was processed.

Example 7-14 shows the main parts of the SOAP syntax.

*Example 7-14 SOAP syntax*

---

```
<soap:Envelope>
  <soap:Header>
    ...
  </soap:Header>
  <soap:Body>
    <GetStock>
      <Company>IBM</Company>
    </GetStock>
  </soap:Body>
</soap:Envelope>
```

---

The SOAP XML document in Example 7-14 on page 274 is used for requesting the IBM share price and is simplified for example purposes.

### 7.3.1 Transport Layer Security and message layer security

Web service security can be described under:

- ▶ Transport Layer Security
- ▶ Message layer security

Transport layer and message layer security can provide security features for:

- ▶ Authentication and authorization
- ▶ Data encryption and signatures

Transport Layer Security secures the network communication between the communication partners by encrypting the data that is transmitted over the network. In addition, data integrity, authentication, and confidentiality can be achieved. Transport Layer Security typically uses digital signatures, PKI certificates, and secure hash functions to prevent messages from being *camouflaged*, passwords from being hacked, and transactions from being denied.

In situations where an environment consists of several hops, the communication between each hop must be considered separately in terms of Transport Layer Security.

As shown in Figure 7-17, the connections between each hop might use different Transport Layer Security methods or even no transport security for certain connections.

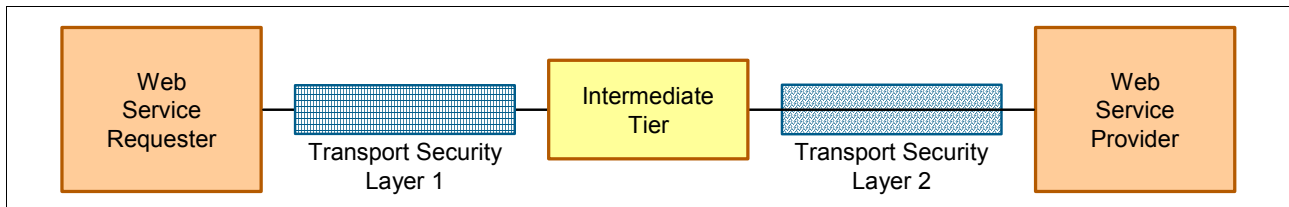


Figure 7-17 Different Transport Layer Security methods

Transport Layer Security does not *span* multiple hops. Therefore, an intermediate hop might read the message. To achieve end-to-end security, you must use message layer security. By using message layer security, the message is secure and does not change when it is sent over multiple hops.

Transport Layer Security can be implemented by using any of the industry-wide protocols, such as the following examples:

- ▶ Secure Sockets Layer (SSL), which is denoted by HTTPS
- ▶ VPN/IPSec which is transparent to applications

Message layer security includes security-related information in the SOAP message or more specifically, within the SOAP header.

The use of authentication allows a service provider to check who is using the requested service. In addition, the service provider can use this information to run the service under a specific user ID, with its associated access rights (authorization).

To fully comprehend authentication and authorization, the following concepts are important to understand:

▶ Authentication

The process of identifying an individual that uses the credentials of that individual.

▶ Authorization

The process of determining whether an authenticated client is allowed to access a resource or perform a task within a security domain. Authorization uses information about a client's identity and roles to determine the resources or tasks that a client can perform.

▶ Credentials

A set of claims that is used to prove the identity of a client. They contain an identifier for the client and a proof of the client's identity, such as a password. They might also include information, such as a signature, to indicate that the issuer certifies the claims in the credential.

▶ Identification

The use of an identifier that allows a system to recognize a particular subject and distinguish it from other users of the system.

Authentication can be performed by using one of the following methods:

▶ Transport layer authentication

Here, the transport layer carries information about who is requesting the service. The following implementations are available:

- HTTP authentication (Basic and Digest Access Authorization, see RFC 2617)
- The use of SSL client authentication with SSL/HTTPS

▶ Message layer authentication

Here, the SOAP message carries information about who is requesting the service. The following implementations are available:

- Direct authentication, by using plain text passwords or a password digest.
- Brokered authentication by using an X.509 certificate, Kerberos, security token services, or SAML Assertion.

Brokered authentication by using an X.509 certificate carries the X.509 certificate as part of the SOAP header (see Example 7-15).

*Example 7-15 Certificate as part of the SOAP header*

---

```
<soap:Header>
  <Security xmlns="...secext-1.0.xsd"
    <BinarySecurityToken EncodigType= "wsse:Base64Binary"
      ValueType= "wsse:X509v3">
      MIICuzCCAiqCBF...
      ...
    </BinarySecurityToken>
  </Security>
  ...
```

---

Direct authentication defines the following ways of transporting the password:

- ▶ Plain text password, in which UsernameToken is used to transport the actual password.

If you use plain text password configuration, you must use a secure transport method, such as HTTPS. Example 7-16 shows how the plain text password is defined.

*Example 7-16 Plain text password*

---

```
<soap:Header>
  <Security xmlns="...secext-1.0.xsd"
    <UsernameToken>
      <Username>John Smith</Username>
      <Password>Pass12wd</Password>
    </UsernameToken>
  </Security>
  ...
```

---

- ▶ Password digest, in which the communicating parties, the requester, and the service use an insecure transport channel.

Steps must be taken to protect the passwords from being made available to others. Here, the requester creates a digest of the actual password that is concatenated with a set of random bytes (called *nonce* here) and another value that depends on the creation-time (field *created*). This digest is computed as shown in the following example:

```
digest = Base64_encode(SHA-1(nonce+created+password))
```

To authenticate the request, the service computes the digest value by using the password that is bound to the received Username. It compares the received digest value with the computed digest value. Example 7-17 shows the related SOAP definitions.

*Example 7-17 Using a password digest*

---

```
<soap:Header>
  <Security xmlns="...secext-1.0.xsd"
    <UsernameToken>
      <Username>John Smith</Username>
      <Password Type="...#PasswordDigest">AFHHF23wger=</Password>
      <Nonce>ksSDGF1jdfD=</Nonce>
      <Created>2010-07-15T07:12:19.573Z</Created>
    </UsernameToken>
  </Security>
  ...
```

---

From z/VSE 4.2 and later, z/VSE supports:

- ▶ Transport layer authentication that uses:
  - HTTP authentication (Basic and Digest Access Authorization)
  - SSL Client Authentication with HTTPS
- ▶ Message layer authentication that uses:
  - UsernameToken (plaintext password or password digest)
  - X.509 Certificate (BinarySecurityToken)

To use SSL, you must create keys and certificates on z/VSE and the remote side by using the Keyman/VSE tool. For more information, see Chapter 5, “Secure Sockets Layer with z/VSE” on page 197.

## 7.3.2 Web service security features with z/VSE as the SOAP server

When z/VSE acts as the SOAP server, consider the following areas as web service provider:

- ▶ Transport layer encryption

When transport-layer authentication is used, z/VSE acts as an HTTP server. This configuration is implemented by using CICS Web Support (CWS) as the HTTP server. CWS passes the SOAP request to the z/VSE SOAP Engine for further processing. Because CWS implements support for HTTP over SSL (HTTPS), the SOAP Engine inherits the security features from CWS. To use HTTPS, the following prerequisites must be met:

- Configure TCPIP SERVICE in CICS for use with SSL
- Create the required keys and certificates

- ▶ Transport layer authentication

CWS supports SSL client authentication (HTTPS) and HTTP Basic Authentication; therefore, the z/VSE SOAP Engine inherits the security features from CWS. To force a client to use HTTP basic authentication, you must configure the TCPIP SERVICE to use the CICS-provided converter program DFH\$WBSB and specify URM=DFH\$WBSB. In addition, the z/VSE SOAP Engine extracts authentication information, which is the user ID and password for HTTP basic authentication, or the mapped user ID for SSL client authentication. This information can be used by the converter code to check if transport layer authentication was used. If authentication was not used, the converter code might reject the request.

- ▶ Message layer authentication

To support message layer authentication, the z/VSE SOAP Engine (that is, the z/VSE SOAP Server) extracts the authentication token from the SOAP header after parsing the XML data stream. In case of UsernameToken, the user ID and password must be verified against a local identity store. To perform this verification, the identity store must compare the plain text password or password digest against its stored password. If user authorization also is to be performed, a user mapping must be performed to map the received user name to a z/VSE user ID.

Also, a CICS SIGNON must be performed by using the mapped user to allow the transaction to run under that user. In addition to UsernameToken, the use of certificates for authentication is possible. In this case, the converter code maps the received certificate to a z/VSE user. z/VSE supports this functionality as part of its support for SSL client authentication.

The z/VSE SOAP Engine does not perform the authentication. Instead, it extracts the security information and passes it to the converter code or user application. It is the user application's responsibility to perform authorization checking or sign-on processing.

The converter program receives the security-related information as part of the COMMAREA it gets called with by the SOAP Engine. In addition to other information, the COMMAREA contains the following fields that are used for authentication:

- ▶ AUTH-TYPE
- ▶ AUTH-USER
- ▶ AUTH-PWD

The AUTH-TYPE specifies what kind of authentication was used by the requester. The following possible values are available:

<b>AUTH_NONE</b>	No authentication has been used
<b>AUTH_HTTP_BASIC</b>	Basic HTTP authentication
<b>AUTH_WS_PLAIN</b>	Web service authentication by using plain password
<b>AUTH_WS_DIGEST</b>	Web service authentication by using password digest
<b>AUTH_WS_CERT</b>	Web service authentication by using X.509 certificate
<b>AUTH_SSL_CERT</b>	SSL client authentication

For AUTH\_HTTP\_BASIC and AUTH\_WS\_PLAIN, the fields AUTH-USER and AUTH-PWD specify the user ID and password that were received from the client side by using the specified method. For AUTH\_WS\_DIGEST, field AUTH-USER contains the user ID, and AUTH-PWD specifies the name of a TS-QUEUE that holds the following entries:

- ▶ Password digest in base64
- ▶ Created time stamp as text
- ▶ Nonce value in base64

For AUTH\_WS\_CERT and AUTH\_SSL\_CERT, the field AUTH-USER contains the user ID that was mapped for the certificate or blanks if no user was mapped. The field AUTH-PWD contains the name of a TS-QUEUE that holds one entry, which is the binary certificate data.

The converter code can use this information to decide whether the used authentication mechanism is sufficient. If it is not sufficient, it might reject the request and return an appropriate error. If the authentication information is sufficient, it can use it to perform local authentication by using available CICS security functions (for example, EXEC CICS SIGNON).

### 7.3.3 Web service security features with z/VSE as the SOAP client

When z/VSE acts as the SOAP client, as web service requester you should consider the following areas:

- ▶ Transport-Layer Encryption

When transport-layer authentication is used, z/VSE acts as an HTTP client. The HTTP client that is implemented in z/VSE V4R2 then supports HTTPS. To use HTTPS:

- The URL must specify: `https://`
- You must provide a public-private key pair, together with certificates. For more information about how to specify the keys, see the skeleton SKSOAPPOP in VSE/ICCF Library 59.

- ▶ Transport-Layer Authentication

From z/VSE V4R2, and later, the HTTP Client supports SSL/HTTPS. Therefore, you can use SSL client authentication using certificates. If requested, the SSL protocol can send the client's certificate to the server (service provider). If required, the server can use the client's certificate to perform authentication and authorization. In addition, HTTP basic authentication is supported by the z/VSE HTTP Client.

- ▶ Message-Layer Authentication

From z/VSE V4R2 and later, the z/VSE SOAP Engine (that is, the z/VSE SOAP Client), supports the authentication token in the SOAP header. For the UsernameToken, the user application or converter code that is requesting the service must pass the user name and password to the SOAP Engine. If authentication is done by using a certificate, the certificate name must be provided. Code for passing this information can be part of the user application or converter code.

The web service requester program can put security-related information into the COMMAREA that is used to call the SOAP Engine. In addition to other information, the COMMAREA contains the following fields that are used for authentication:

- ▶ AUTH-TYPE
- ▶ AUTH-USER
- ▶ AUTH-PWD

The AUTH-TYPE specifies the type of authentication to use, including the following possible values:

<b>AUTH_NONE</b>	No authentication was used
<b>AUTH_HTTP_BASIC</b>	Basic HTTP authentication
<b>AUTH_WS_PLAIN</b>	Web service authentication by using plain password
<b>AUTH_WS_DIGEST</b>	Web service authentication by using password digest
<b>AUTH_WS_CERT</b>	Web service authentication by using X.509 certificate
<b>AUTH_SSL_CERT</b>	SSL client authentication

For AUTH\_HTTP\_BASIC, AUTH\_WS\_PLAIN, and AUTH\_WS\_DIGEST, the fields AUTH-USER and AUTH-PWD specify the user ID and password to be sent to the server by using the chosen method. For AUTH\_WS\_CERT, the field AUTH-USER specifies the name of the certificate member to use; for example, CRYPTO.KEYRING(CERTNAME). In this case, the field AUTH-PWD is not used and is ignored. AUTH\_SSL\_CERT is not supported for SOAP client operations. However, the server side can decide to use the client certificate that is used with SSL and perform client authentication.

## 7.4 z/VSE Database Connector (DBCLI)

Starting with z/VSE 5.1, DBCLI allows z/VSE applications to access a relational database on any suitable database server. You can choose a database server (IBM DB2, Oracle, Microsoft SQL Server, MySQL, and so on) that runs on a platform other than z/VSE. This z/VSE database connector is available through the z/VSE Database Call Level Interface (DBCLI), which provides an application programming interface (API) for z/VSE applications that are written in Assembler, COBOL, PL/I, C, or REXX. It supports batch and CICS application.

The z/VSE DBCLI solution consists of two main parts: the DBCLI client on z/VSE, and the DBCLI server on a Java platform. Both parts are connected by way of TCP/IP. The interaction of z/VSE and the Data Base server by using the DBCLI client part is shown in Figure 7-18 on page 281.



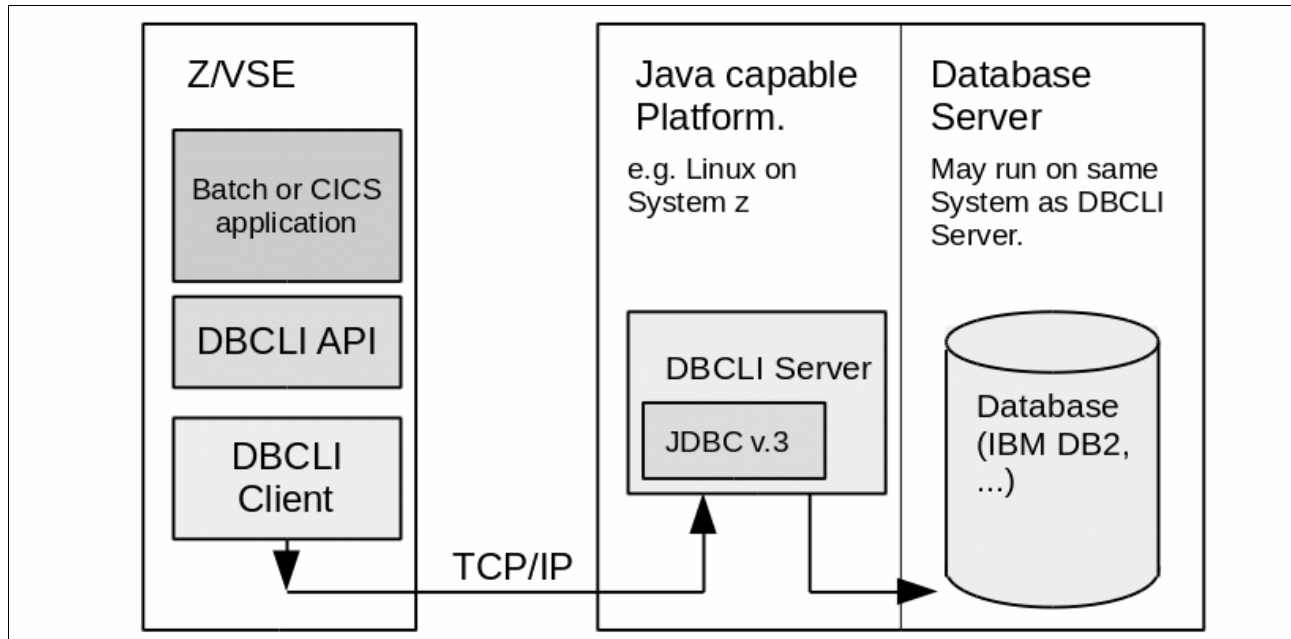


Figure 7-18 Access relational Data Base by way of DBCLI

### 7.4.1 z/VSSE DBCLI security features

#### Sign-on security based on user ID and password

When a client connects to the server, the client must supply a valid user ID and password before it can access any resources. The user ID and password are checked by the database server.

#### DBCLI with SSL

Starting from z/VSSE 6.1, the DBCLI supports SSL/TLS connections to the database server. To use that functionality, the client application must use the CONNECTSSL function to connect to the server. CONNECTSSL establishes an *encrypted connection* to the DBCLI server (DBCLIServer) by using the SSL or TLS protocol.

If SSL is used, you must call the INITSSL function before calling the CONNECTSSL function. The CONNECTSSL allocates a connection handle that represents the connection and sets the CON-HANDLE parameter. The application must pass the connection handle to all subsequent functions that require a connection.





## TCP/IP security

In this chapter, we describe the security concept of TCP/IP for VSE/ESA 1.5<sup>1</sup> and how you can use it.

We show you the security components of TCP/IP, how security features can be activated or deactivated, and how the complete TCP/IP security can be enabled or disabled.

The TCP/IP security concept allows you to define user IDs and resources within TCP/IP or restrict the user IDs for a special type of use. Also, the security exit interface of TCP/IP allows you to start security manager functions.

We also show you the security exit routine (provided by IBM) and how you can extend it for your special requirements.

This chapter includes the following topics:

- ▶ 8.1, “TCP/IP security concept” on page 284
- ▶ 8.2, “Defining user IDs” on page 287
- ▶ 8.3, “Security exit points and security managers” on page 288

---

<sup>1</sup> IBM product number 5686-A04

## 8.1 TCP/IP security concept

A security implementation often considers the following issues:

- ▶ Defining user IDs and passwords.
- ▶ Definitions of the resources you want to protect. These resources can be a functional entity, such as an application or a physical entity, such as a file.
- ▶ Authorizations defining which users can perform which functions and which users can access which resources.

This concept is also implemented in TCP/IP. Since TCP/IP 1.5 Service Pack E with APAR PQ87041, all processes run under user IDs and passwords explicitly or by default.

The following processes use default user IDs:

- ▶ Automation (event) processing

You can use DEFINE EVENT to automatically send VSE/POWER list queue entries through email, File Transfer Protocol (FTP), or Line Printer Remote (LPR). In this case, the default user ID \$EVENT and password \$EVENT is used. You can override these values with DEFINE EVENT as command or in a script. If you do not overwrite it, add the following command to your initialization deck:

```
DEFINE USER, ID=$EVENT, PASSWORD=$EVENT, LPR=YES
```

- ▶ Line Printer Remote (LPR) processing

The LPR client can be used to send documents to a remote printer or Line Printer Daemon (LPD). It uses as default user ID \$LPR and password \$LPR. The user can overwrite these values with the LPR commands SET USERID and SET PASSWORD explicitly or with a script. If you do not overwrite it, add the following command to your initialization deck:

```
DEFINE USER, ID=$LPR, PASSWORD=$LPR, LPR=YES
```

- ▶ Line Printer Daemon (LPD) processing

The line printer daemon can be used to receive print data from remote systems. As the default user ID and password, it uses \$LPD for both. If you are using LPD without the parameters USERID= and PASSWORD= on the DEFINE LPD command, add the following command to your initialization deck:

```
DEFINE USER, ID=$LPD, PASSWORD=$LPD, LPD=YES
```

- ▶ HTTP (Hypertext Transfer Protocol) processing

With the HTTP daemon, z/VSE can be used as a web server. It uses as default user ID \$WEB and password \$WEB. If you are using an HTTP daemon without using the parameters USERID= and PASSWORD= on the DEFINE HTTPD command, add the following command to your initialization deck:

```
DEFINE USER, ID=$WEB, PASSWORD=$WEB, WEB=YES
```

- ▶ Email processing

The EMAIL client can be used to send data as an email to a Simple Mail Transfer Protocol (SMTP) server. It uses as default user ID \$EMAIL and password \$EMAIL. If you are using email processing and do not use the parameters USERID=, PASSWORD=, LUSERID=, and LPASSWORD= on the EMAIL command and do not overwrite the default user ID with the SET command explicitly or with a script, add the following command to your initialization deck:

```
DEFINE USER, ID=$EMAIL, PASS=$EMAIL
```

► POWER processing

When TCP/IP accesses VSE/POWER (for example, to put a job in the RDR queue during an FTP session), the VSE/POWER user ID and password is used. You can specify it with SET POWERUSERID= and SET POWERPASSWORD=. If it is not specified, the default user ID is SYSTCPIP and the default password is XL8'00'.

To enable security, use the **SECURITY ON** command. By default, security is off, and any user ID or password are accepted.

**Note:** SET SECURITY=ON might be used with older versions of TCP/IP. This command is no longer recommended, but still valid for compatibility reasons.

Starting with TCP/IP 1.5 Service Pack E and APAR PQ87041, you can use the **SECURITY** and **QUERY SECURITY** commands to control and monitor the security functions. If you do not know whether security is active, issue the **QUERY SECURITY** command.

If the response indicates that security is OFF, you can use the following command to activate security in WARN mode:

```
SECURITY ON MODE=WARN AUTO=ON
```

This command creates a message log that can help identify the user IDs that are used and resources that are accessed. It still allows all logins and accesses because MODE=WARN is specified.

The next section describes the SECURITY command.

### 8.1.1 Control the security functions with the SECURITY command

The TCP/IP for VSE/ESA command **SECURITY** is used to control and manage TCP/IP for VSE/ESA security. This command contains one or more options, which are separated by blanks or commas, that define or modify current security settings.

The most important command options are:

- |                       |   |
|-----------------------|---|
| <b>ON</b>             | Activates global security processing in the TCP/IP partition.   |
| <b>BATCH=ON</b>       | Activates security processing in external batch partitions, for example FTPBATCH.   |
| <b>MODE=WARN/FAIL</b> | Controls whether incorrect security requests are allowed with a <i>warning</i> or refused with a <i>failure</i> .   |
| <b>LOCK</b>           | Locks all security settings to their current values. After this command is issued, the security settings cannot be changed or switched off by the operator. |

TCP/IP includes an automatic security manager. Automatic security can be used with the **DEFINE USER** command to allow or prevent specific user access to data or to control system-level resources where no user ID and password were established.

For example, the following command allows or prevents z/VSE from responding to incoming ICMP PING requests:

```
ASECURITY ICMP=YES/NO
```

The options on the SECURITY command for controlling the automatic security manager are:

- AUTO=ON/OFF** Controls loading and activating of automatic security manager.
- ADATA=** Specifies a 40-byte character string to be passed to the automatic security manager with each call.

For more information about the auto security manager, see *TCP/IP for VSE V1R5.0 Installation Guide*, SC33-6762.

TCP/IP provides a security exit interface to allow more security functions. This exit interface can be used for a user-created security exit phase or the IBM provided security exit phase BSSTISX, which uses the BSM, or an external security manager (ESM) from an ISV for its security decisions. The most important command options for controlling another security exit routine are:

- EXIT=ON/OFF** Controls loading and activation of the security exit.
- PHASE=** Specifies the name of the security exit phase.
- XDATA=** Specifies a 40-byte character string to be passed to the security exit with each call.

For more information about the SECURITY command, see *TCP/IP for VSE V1R5.0 Installation Guide*, SC33-6762.

### Mapping old security commands to new SECURITY command

The commands that were defined in previous releases are still valid. Existing initialization decks and procedures continue to function. However, they all can be replaced with the SECURITY command of TCP/IP for VSE/ESA V1.5 service pack E, as listed in Table 8-1.

Table 8-1 Security commands

Old security-related commands	New SECURITY command
DEFINE SECURITY	SECURITY PHASE=nnn XDATA=xxx EXIT=ON
DELETE SECURITY	SECURITY EXIT=OFF
SECURITYARP	SECURITY ARP=
SET SECURITYARP=	SECURITY ARP=
SECURITYIP	SECURITY IP=
SET SECURITYIP=	SECURITY IP=
SECURITY ONX	SECURITY ON BATCH=ON

For example, the following command sequence can be replaced:

```
DEFINE SECURITY, DRIVER=USEREX, DATA='ABCD'
SET SECURITY_ARP=ON
SET SECURITY_IP=ON
SET SECURITY ON
```

The command can be replaced with:

```
SECURITY ON, PHASE=USEREX, XDATA='ABCD', ARP=ON, IP=ON, EXIT=ON
```

## 8.2 Defining user IDs

After a **SECURITY ON** command is issued, the easiest security that can be implemented is to activate user identification with passwords. A user ID and password then are required before access is granted to secured TCP/IP for VSE/ESA applications, such as FTP, Telnet, and HTTP.

TCP/IP for VSE/ESA enables you to explicitly or implicitly define user IDs and passwords. You can explicitly create user IDs by using the TCP/IP for VSE/ESA **DEFINE USER** command, or you can use a security manager through the security exit to implicitly define users to TCP/IP for VSE/ESA. The security exit gets control and can permit or deny access based on user IDs and passwords, regardless of whether the user IDs and passwords are defined with **DEFINE USER** command.

### 8.2.1 Explicitly defining user IDs

Since TCP/IP 1.5 Service Pack E with APAR PQ87041, user IDs can be associated with specific uses. For example, having a valid user ID for TN3270 access does not automatically permit FTP access.

The explicit mechanism that is used to define user IDs is the TCP/IP for VSE/ESA **DEFINE USER** command.

This command features the following syntax:

```
DEFINE USER, ID=id[, PASSWORD=pswd] [, DATA='data']  
      [, FTP=YES/NO] [, LPR=YES/NO] [, WEB=YES/NO]  
      [, TELNET=YES/NO] [, ROOT='directory']
```

You can issue this command anywhere you can issue TCP/IP for VSE/ESA operator commands, including the console, an include deck, the IPNETCMD command processor, or the TCP/IP for VSE/ESA initialization deck. This approach implies the following loopholes in security:

- ▶ Anyone who can issue z/VSE operator commands can define user IDs. To prohibit the use of the z/VSE operator console as a mechanism for issuing TCP/IP for VSE/ESA operator commands, use the **SET PASSWORD** command in the TCP/IP for VSE/ESA initialization deck.
- ▶ If you choose to define your user IDs in the TCP/IP for VSE/ESA initialization deck, anyone with access to the initialization deck also has access to all user IDs and all passwords.
- ▶ When the **DEFINE USER** command is entered from the console, the password that is specified in the **DEFINE USER** statement displays on SYSLST. You can suppress the password by starting the command with a plus sign (+) character. Any command that starts with a + does not display on SYSLST.

Despite these loophole considerations, the **DEFINE USER** command can provide effective protection for identifying who is accessing TCP/IP for VSE/ESA resources.

The **PASSWORD** option can be any value up to 16 bytes. It is not case-sensitive when checked before calling the security manager, although the user security manager can enforce case sensitivity.

The **DEFINE USER** command supports limiting a user ID to the following options:

<b>FTP=YES/NO</b>	Controls FTP access by this user.
<b>LPR=YES/NO</b>	Controls LPR access by this user.

**WEB=YES/NO** Controls web page access by this user.  
**TELNET=YES/NO** Controls Telnet menu access by this user.

**Note:** If *none* of the options is explicitly coded, *all* functions are permitted. If *any* of the options are used, the default for all uncoded options is NO.

Consider the following examples:

- ▶ User can access everything:  
DEFINE USER ID=ABC,PASS=XYZ
- ▶ User can access *only* FTP:  
DEFINE USER ID=ABC,PASS=XYZ,FTP=YES
- ▶ User *cannot* access anything:  
DEFINE USER ID=ABC,PASS=XYZ,FTP=NO

A root directory can be defined for a user with the **DEFINE USER** command by using the **ROOT=** option. For example, to restrict the user ID to a specific directory, use:

```
ROOT='directory'
```

This command restricts the user to that directory or lower. For example, to restrict a user ID to only the VSE/POWER LST queue class D, you enter:

```
DEFINE USER, ID=ABC, PASS=XYZ, .ROOT= '/POWER/LST/D' , FTP=YES
```

Setting a ROOT of forward slash ('/') or backslash ('\') starts the user in either UNIX or VSE mode. Also, the use of the ROOT= parameter requires that SECURITY ON is active.

For more information about the DEFINE USER command, see *TCP/IP for VSE V1R5.0 Installation Guide*, SC33-6762.

## 8.3 Security exit points and security managers

TCP/IP for VSE/ESA provides security exit points for comprehensive security control. A security exit routine can be called at various points in TCP/IP for VSE/ESA processing, including validation of user IDs and passwords, and access to resources by a specific user ID.

A security exit routine is called from a security exit point. Based on the return code from the security exit routine, the operation status is permitted, warned, or denied. The security exit can also log the accessed resource. The following security functions can be activated with the SECURITY command:

- ▶ TCP/IP for VSE/ESA automatic security manager
- ▶ User-created security exit routine
- ▶ Security exit routine (provided by IBM) to use the BSM or an ESM
- ▶ Other vendor-provided security exit routines

Regardless of which security function is activated, the TCP/IP for VSE/ESA exit passes over control and information about the resource that is accessed in a common security exit block (SXBLOK).

Before you create your own user security exit routine, first closely review the automatic security exit that is provided with TCP/IP for VSE/ESA.



For more information about the auto security manager, see *TCP/IP for VSE V1R5.0 Installation Guide*, SC33-6762.

The automatic security manager and one other security exit routine can be active and user-created, provided by IBM, or from another vendor.

Security functions are activated and controlled with the SECURITY command.

### 8.3.1 Flow of a security request

The sequence of a security request can be important, depending on which security functions are used and activated. The flow from security through the various security functions includes the following steps:

1. A TCP/IP for VSE/ESA application (for example, FTP) creates an SXBLOK in the security exit.
2. User ID/password (if present) is checked against DEFINE USER information. The result is set in SXBLOK along with a default return code.
3. If specified, *automatic* processing is performed. The result is set in the SXBLOK and overrides any return code set in step 2.
4. If specified, user exit processing is performed. The user exit can consider the result of the preceding steps, or it can override it.

### 8.3.2 Using Basic Security Manager with TCP/IP

IBM provides the security exit phase BSSTISX as an interface between TCP/IP and Basic Security Manager (BSM).

Figure 8-1 shows the integration of BSSTISX.

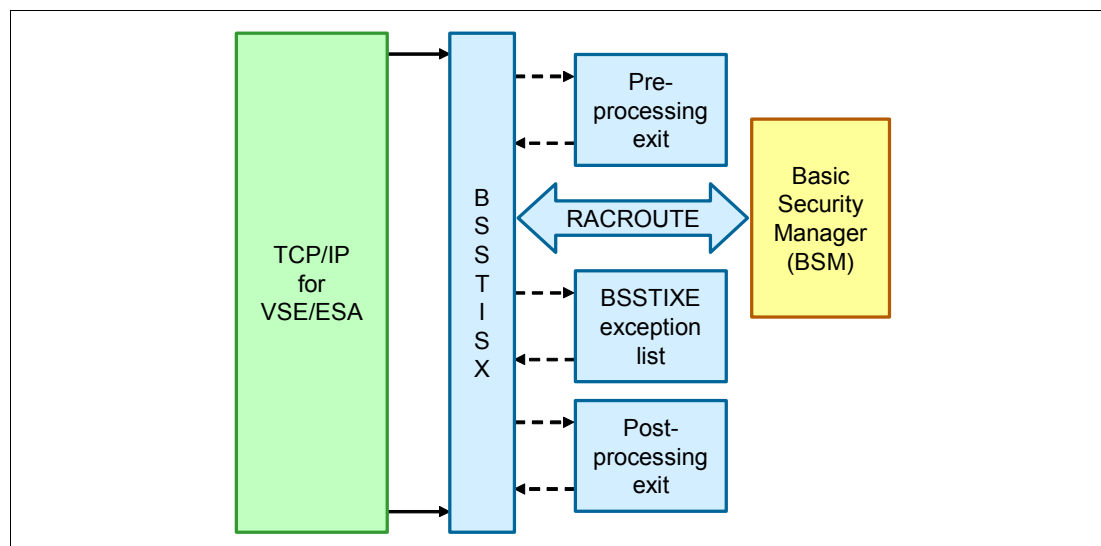


Figure 8-1 Security exit BSSTISX overview

The phase BSSTISX uses the BSM capabilities. It issues RACROUTE requests to process user identification and user authentication, and resource access control for z/VSE files, libraries, and members. It also allows limited access control to VSE/POWER spool files and the SITE command.

Access to VSE/POWER spool files are allowed for administrators and users, where either of the following statements is true:

- ▶ The user ID matches the FROM or TO user ID of the requested spool file.
- ▶ TO=ANY was specified.

The user ID that is assigned to ANONYMOUS cannot access these files.

The **SITE** command can be used by an administrator only.

Certainly, various other checks are possible through the TCP/IP exit points, which are not covered by BSSTISX. Therefore, BSSTISX provides a pre- and post-processing exit interface. Customers who require more checks can then write their own pre- or post-processing routines for BSSTISX.

### Exception list BSSTIXE

In general, the exit BSSTISX rejects *all* access requests that cannot be evaluated by this exit. However, accepting certain requests might be necessary. These requests can be specified in this exception list by the client. The exception list must be assembled and linked as phase BSSTIXE (see SKEXCLST in library 59).

The IBM distributed phase and the related source member BSSTIXE.A are in sublibrary IJSYSRS.SYSLIB.

A request is defined by the SXBLOK fields SXTYPE and SXFTYPE. The SXBLOK describes the interface between TCP/IP and BSSTISX. The layout of the SXBLOK is distributed together with TCP/IP for VSE/ESA.

**Important:** Consider the following points:

- ▶ Define only requests in the exception list that cannot be evaluated by BSSTISX and are therefore rejected by BSSTISX.
- ▶ The requests that are defined in the exception list are *not* security-checked.
- ▶ Be sure to add *only* requests to the exception list that do not affect the security requirements of your installation.

Instead of the exception list, you can use the BSSTISX PRE and POST-PROCESSING EXITS to add your installation-specific security checks.

### Activating the security exit BSSTISX

To activate the security exit, enter the following TCP/IP command:

```
SECURITY ON,PHASE=BSSTISX,XDATA='data',EXIT=ON
```

This command activates security processing and gives control to BSSTISX for initialization.

BSSTISX loads more parts into storage and initializes its control blocks according to the parameters that are specified in data.

Now, TCP/IP passes information to the exit routine BSSTISX for verification.

Add BATCH=ON to the **SECURITY** command to activate security processing in external batch partitions; for example, FTPBATCH.

The parameter `XDATA=` of the **SECURITY** command contains the initialization parameter for BSSTISX. BSSTISX features the following syntax:

```
XDATA=' [anonym_uid] [, [anonym_pwd] [, [preproc] [, [postproc]]] ]'
```

The parameters include:

- |                   |   |
|-------------------|---|
| <b>anonym_uid</b> | Can be used to specify a user ID, which is defined to BSM. When a client logs on with user ID ANONYMOUS, your specified user ID and its access rights are used. |
| <b>anonym_pwd</b> | Can be used to specify the password of the BSM defined user ID for user ANONYMOUS.  |
| <b>preproc</b>    | If you want to use a self-written preprocessing exit, use this parameter to specify the name of your preprocessing exit phase.                                  |
| <b>postproc</b>   | For a self-written post-processing exit, you must specify the name of your post-processing exit phase here.   |

## Deactivating the security exit

To deactivate the security exit, you must enter one of the following TCP/IP commands:

```
SECURITY OFF  
SECURITY ON,EXIT=OFF
```

This command stops the security processing and gives control to BSSTISX for cleanup and termination. BSSTISX clears its control blocks and frees the storage of its extra parts.

**Note:** If you want to use a new version of the same security exit, you should shut down TCP/IP and restart it again before you use `SECURITY ON,EXIT=ON`.

## Using pre- and post-processing exits

The preprocessing exit gets control after the BSSTISX initialization and later on at the beginning of each request. The post-processing exit gets control at the end of each request, except the termination request. Both exits get the required information from the TCP/IP created SXBLOK.

The SXBLOK describes the interface between TCP/IP's exit point and the security exit. The mapping of the SXBLOK is shipped with TCP/IP for VSE/ESA. Be sure that you use this level of the SXBLOK, which was shipped with the version of TCP/IP you are using for the BSSTISX pre- and post-processing exits.

For the complete interface description of the pre- and post-processing exits, see *TCP/IP for VSE/ESA - IBM Program Setup and Supplementary Information*, SC33-6601.

## Issue RACROUTE requests from your pre- or post-processing exit

If you want to use security information from the BSM in your exit, you must use RACROUTE requests to perform the following tasks:

1. Authenticate the user.
2. Check user's authorization for the resource you want to control in your exit.

**Note:** If you are not familiar with RACROUTE, you can use the RACROUTE encapsulation services for TCP/IP security exit BSSTISX, which are available at this web page:

<http://www.ibm.com/systems/z/os/zvse/downloads/tools.html#main>

We show you how to use RACROUTE in a simplified example. Assume that you want to control that only users from a certain IP address, for which they are explicitly authorized, can access z/VSE. Complete the following steps:

1. Define your own resource names.

Resources use general-purpose resource class FACILITY. You can define a name, as shown in the following example:

```
MYTCPIP.IP.9.123.45.123
```

The first qualifier, MYTCPIP, should identify the application to avoid conflicts with other applications that use also resource class FACILITY.

2. Define the profile for this resource in BSM and authorize the users.

Specify READ authority on the access list of this profile for users that are allowed to use z/VSE from this IP address. For more information, see 2.3.4, “Resource profile definition” on page 34.

3. In your exit, you must:

- a. Verify the user ID and password through a RACROUTE VERIFY CREATE request or use the service BSSTXSON. If the verification was successful you get an ACEE, which is a control block that describes the user.
- b. If the IP address of this user is 9.123.45.123, check that this user is authorized for this address. First, issue a RACROUTE AUTH request or use service BSSTXAUT with:
  - Resource class FACILITY
  - Resource name MYTCPIP.IP.9.123.45.123
  - The requested authorization READ
  - ACEE

Then, according to the result, reject or allow the request.

- c. Delete the ACEE with RACROUTE VERIFY DELETE or service BSSTXSOF.

Depending on your requirements, you can do this also with other information that is provided by the SXBLOK.

### Performance hints

Depending on the TCP/IP usage, BSSTISX might have to issue many user verifications with the same user IDs. For this condition, activating the BSM cache is useful, as shown in the following example:

```
MSG xx,DATA=DBSTARTCACHE
```

Where xx indicates the partition ID of the security server partition (default is FB).

### External security managers

The TCP/IP security exit BSSTISX can also be used with external security manager (ESM) from an ISV, if this ESM supports the RACROUTE requests that are issued by BSSTISX.



## Secure Telnet

This chapter describes the setup of secure Telnet 3270 sessions in scenarios with z/VSE acting as server. Telnet 3270 describes the process of sending 3270 data streams through the Telnet protocol. Because this process is different from standard Telnet as used in UNIX environments, special Telnet 3270 clients are needed for communication with IBM mainframes.

In this chapter, we show the setup with the following Telnet 3270 clients:

- ▶ IBM Personal Communications
- ▶ Attachmate EXTRA! X-treme

Unless otherwise noted, we use the term *Telnet* as synonym for *Telnet 3270*.

This chapter includes the following topics:

- ▶ 9.1, “Introduction” on page 294
- ▶ 9.2, “Setting up a Telnet daemon, TELNETD” on page 294
- ▶ 9.3, “z/VSE host setup for secure Telnet” on page 296
- ▶ 9.4, “Client setup with Personal Communications” on page 298
- ▶ 9.5, “Client setup with Attachmate EXTRA! X-treme” on page 309

## 9.1 Introduction

Secure Telnet requires RSA key pairs and digital certificates on the server and on the client side. For more information about creating keys and certificates, see 5.1, “Generating the server key and certificates” on page 198.

In the examples, we use the following software:

- ▶ z/VSE V4R2
- ▶ TCP/IP for VSE/ESA 1.5F
- ▶ z/VSE Connector Server as part of z/VSE V4R2 (job STARTVCS)
- ▶ Microsoft Windows XP Professional, SP2
- ▶ Java 1.4.2 from Sun Microsystems
- ▶ Keyman/VSE, update from 08/2007
- ▶ IBM Personal Communications 5.7 for Windows
- ▶ Attachmate EXTRA! X-treme V9 Evaluation for Windows

The following fix is necessary for secure Telnet: ZP15F202 (APAR PK33472) for TCP/IP 1.5F. The initially shipped TCP/IP 1.5F does not support secure Telnet connections.

## 9.2 Setting up a Telnet daemon, TELNETD

First, we perform the basic setup for unsecure Telnet. Later, we add the definitions for secure Telnet. The following command defines a standard Telnet daemon (TELNETD):

```
DEFINE TELNETD, ID=LU, TERMNAME=TELNLU, TARGET=DBDCCICS, PORT=23, COUNT=4, -  
    LOGMODE=S3270, LOGMODE3=D4B32783, LOGMODE4=D4B32784, -  
    LOGMODE5=D4B32785, POOL=YES
```

Example 9-1 shows the daemon startup on the z/VSE console.

*Example 9-1 Output of the Telnet daemon startup*

---

```
F7 0097 0030: TEL900I Daemon Startup Telnet Termname: TELNLU04 Port: 23  
F7 0097 002F: TEL900I Daemon Startup Telnet Termname: TELNLU03 Port: 23  
F7 0097 002E: TEL900I Daemon Startup Telnet Termname: TELNLU02 Port: 23  
F7 0097 002D: TEL900I Daemon Startup Telnet Termname: TELNLU01 Port: 23
```

---

We can now immediately use this daemon with a Telnet 3270 capable client.

Figure 9-1 shows Personal Communications when setting up the session with the z/VSE IP address.

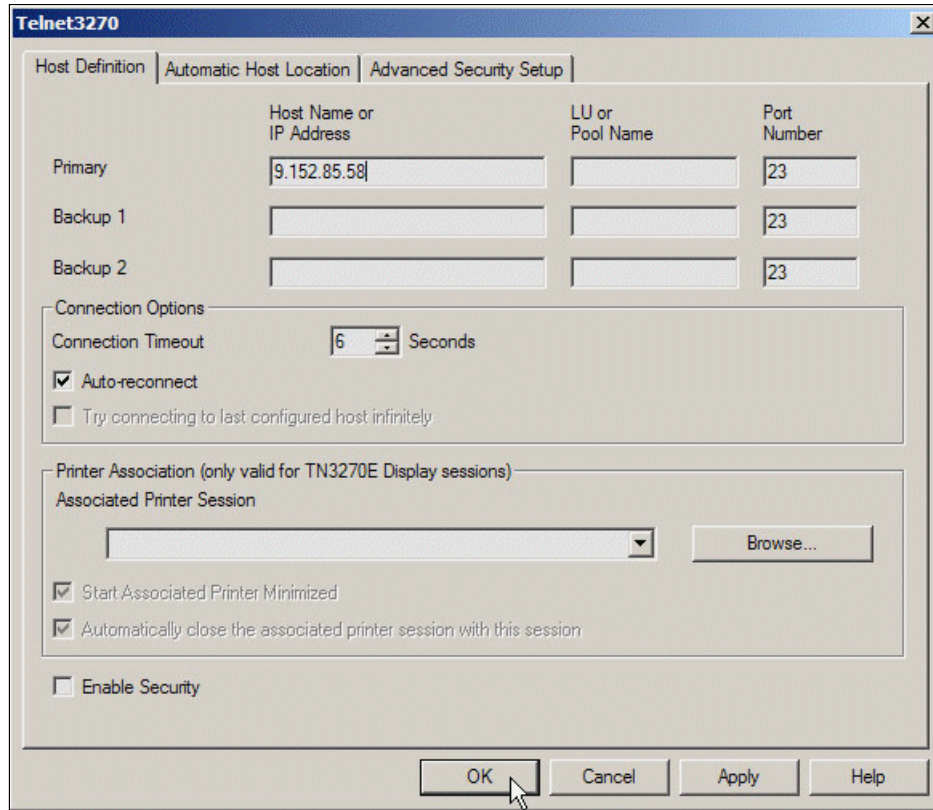


Figure 9-1 Setting up the host definition in Personal Communications

Figure 9-2 shows a z/VSE sign-on panel that is connected through Telnet.

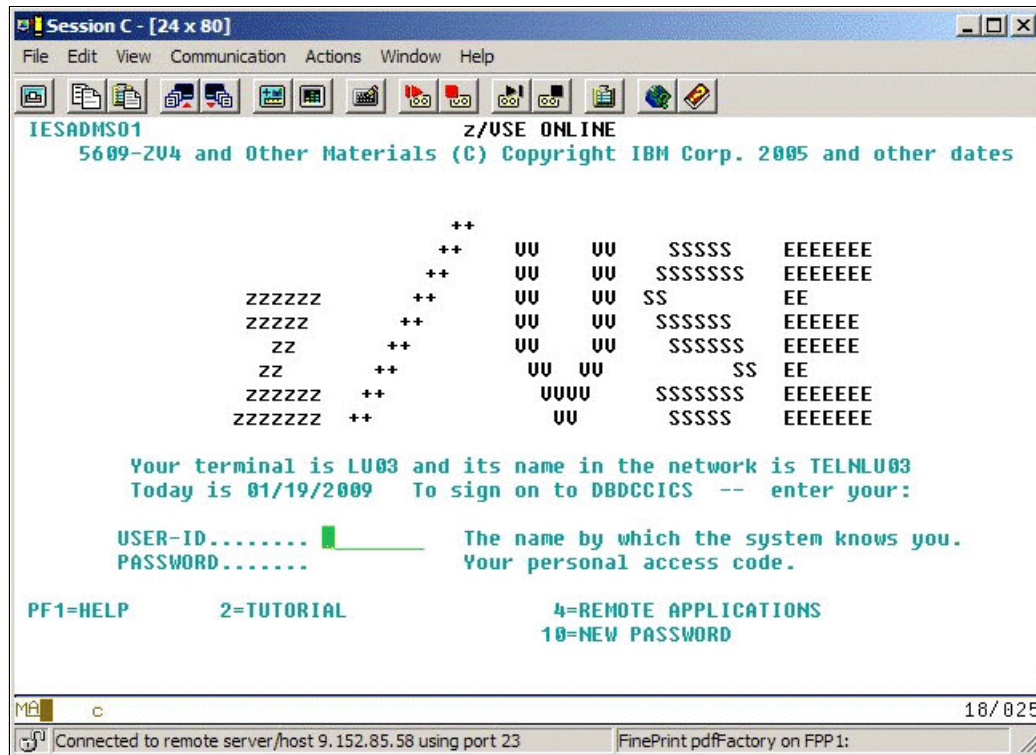


Figure 9-2 z/VSE sign on panel

For more information about how the previously defined Telnet daemon is SSL-enabled, see 9.3, “z/VSE host setup for secure Telnet” on page 296.

### 9.3 z/VSE host setup for secure Telnet

Setting up SSL for the Telnet protocol is based on the SSL daemon that is provided by TCP/IP for VSE/ESA. The following modes are available:

- ▶ SSL in *native* mode. SSL traffic goes directly to the SSL-enabled daemon on z/VSE. Native mode is supported by the HTTP daemon only. In addition, you must define a TLS daemon on the same port. The HTTP daemon obtains all SSL-related settings from the related TLS daemon.
- ▶ SSL in *pass-through* mode, which must be used for secure Telnet, but can also be used for HTTP. Here, we also must define a TLS daemon. The difference with native mode is that we use the `PASSPORT` parameter to route SSL traffic from an unsecured daemon to the SSL daemon.



### 9.3.1 Setting up pass-through mode with a TLS D

The following TLS daemon (TLS D) definition is used to start the Telnet server on the z/VSE side:

DEFINE TLS D, ID=TLS DTELNET,	ID of this SSL/TLS daemon
PORT=992,	Secure Telnet port
PASSPORT=23,	Port that data is passed to
CIPHER=2F350A096208,	Allowed cipher suites
CERTLIB=CRYPTO,	Library name
CERTSUB=KEYRING,	Sublibrary name
CERTMEM=SECTELN,	Member name
TYPE=1,	SSL server authentication
MINVERS=0300,	Minimum version required
DRIVER=SSLD	Driver phase name

Be sure to specify the same member name (here, it is SECTELN) when uploading the keyring files to z/VSE. For more information, see 5.1.2, “Creating the z/VSE key and certificates” on page 200. On the z/VSE console, you see the daemon startup, as shown in Example 9-2.

*Example 9-2 SSL daemon startup*

---

```
F7 0097 0036: TLS900I Daemon Startup Transport Security Layer SSLD
=81640040
```

---

Parameter CIPHER in the TLS D definition lists the hex codes of the ciphers you want to use with this TLS D. When the secure Telnet session is established, the client and server negotiate one of these cipher suites to be used. The session can fail if no cipher suite is supported by both sides.

### 9.3.2 Setting up SSL native mode

SSL native mode is used automatically when the DEFINE TLS D contains the same port number for the PORT and PASSPORT parameters, as shown Example 9-3.

*Example 9-3 Define TLS D with SSL native mode*

---

```
DEFINE TLS D, ID=TLS DTELNET, -
    PORT=992, -
    PASSPORT=992, -
    CIPHER=2F350A0962, -
    MINVERS=0300, -
    CERTLIB=CRYPTO, -
    CERTSUB=KEYRING, -
    CERTMEM=SECTELN, -
    TYPE=1, -
    DRIVER=SSLD

DEFINE TELNETD, ID=TELVSSL, TCPAPPL=TNSSL0, TARGET=DBDCCICS, -
    COUNT=3, MENU=MENUZ, TYPE=VTAM, POOL=YES, PORT=992, -
    DRIVER=TELNETD
```

---

As defined in Example 9-3, TELNETD natively supports SSL when you take the necessary SSL configuration information from the DEFINE TLS D keywords.

### 9.3.3 Setting up a Telnet listener daemon

When more security is required, such as restricting Telnet traffic to specific IP addresses, you can use the following variant of setting up secure Telnet on z/VSE.

This scenario requires TCP/IP for VSE/ESA 1.5F with fix ZP15F202 (APAR PK33472).

As shown in Example 9-4, the listener daemon accepts connections from the specified IP address only, which often belongs to a dedicated Terminal server. This configuration is possible with TN3270E only.

*Example 9-4 Telnet listener daemon*

---

```
DEFINE TELNETD, TN3270E=L, PORT=992, POOL=YES, DRIVER=TELNETD, -  
    ID=TLSSL1, IPADDR=3.196.98.105  
*  
DEFINE TLSL, ID=TLS2, PORT=992, PASSPORT=992, CIPHER=090A62, -  
    CERTLIB=CRYPTO, CERTSUB=KEYRING, CERTMEM=SECTELO1, TYPE=1, DRIVER=SSLD, -  
    MINVERS=0300  
*  
DEFINE TELNETD, TERMNAME=TLS32AA, ID=TLS32AA, POOL=YES, -  
    TN3270E=E, TYPE=VTAM, MENU=MENU2, LOGMODE=SP3272QN, -  
    LOGMODE3=SP3272QN, LOGMODE4=SP3272QN, LOGMODE5=SP3272QN, PORT=992, -  
    DRIVER=TELNETD
```

---

This JCL that is shown in Example 9-5 adds the related VTAM definitions.

*Example 9-5 VTAM definitions for Telnet*

---

```
* $$ JOB JNM=CAT, CLASS=0  
// JOB CAT TEST  
// EXEC LIBR, PARM='A S=PRD2.CONFIG'  
CATALOG TLSL.B REPLACE=YES  
TLSL VBUILD TYPE=APPL  
TLS32AA APPL AUTH=(ACQ), MODETAB=IESINCLM, EAS=1  
TELNU01 APPL AUTH=(ACQ), MODETAB=IESINCLM, EAS=1  
TELNU02 APPL AUTH=(ACQ), MODETAB=IESINCLM, EAS=1  
TELNU03 APPL AUTH=(ACQ), MODETAB=IESINCLM, EAS=1  
/+  
/&  
* $$
```

---

## 9.4 Client setup with Personal Communications

You must import the self-signed root certificate into the Personal Communications key database. Importing the PFX file into the Personal Communications key database is important so that the contained private keys are not lost. For more information, see 5.1, “Generating the server key and certificates” on page 198.

### 9.4.1 Importing the z/VSE certificates into Personal Communications

Open the IBM Key Management tool, which is part of the Personal Communications installation. You find the tool under **Utilities** → **Certificate Management**.

## Location of Personal Communications key database in Windows

On Windows, the Personal Communications key database is in the following folder:

C:\Documents and Settings\All Users\Application Data\IBM\Personal Communications

This folder is only visible through the Windows Explorer when your current folder options specify to show hidden files and folders. You can change this setting in the Windows Explorer by clicking **Tools** → **Folder options**.

Figure 9-3 shows the folder options. Select the **Show hidden files and folders** option.

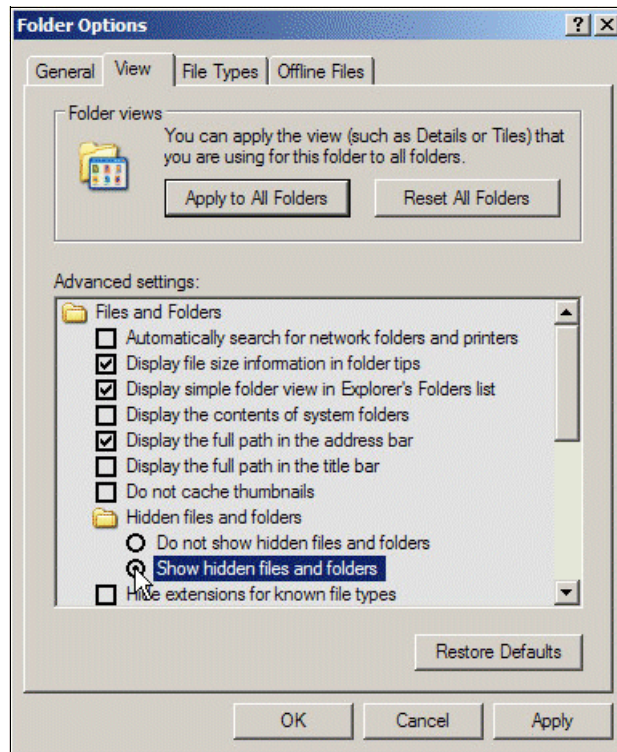


Figure 9-3 Folder options to show hidden files and folders

## Opening the Personal Communications key database

Now, open the Personal Communications key database, as shown in Figure 9-4.

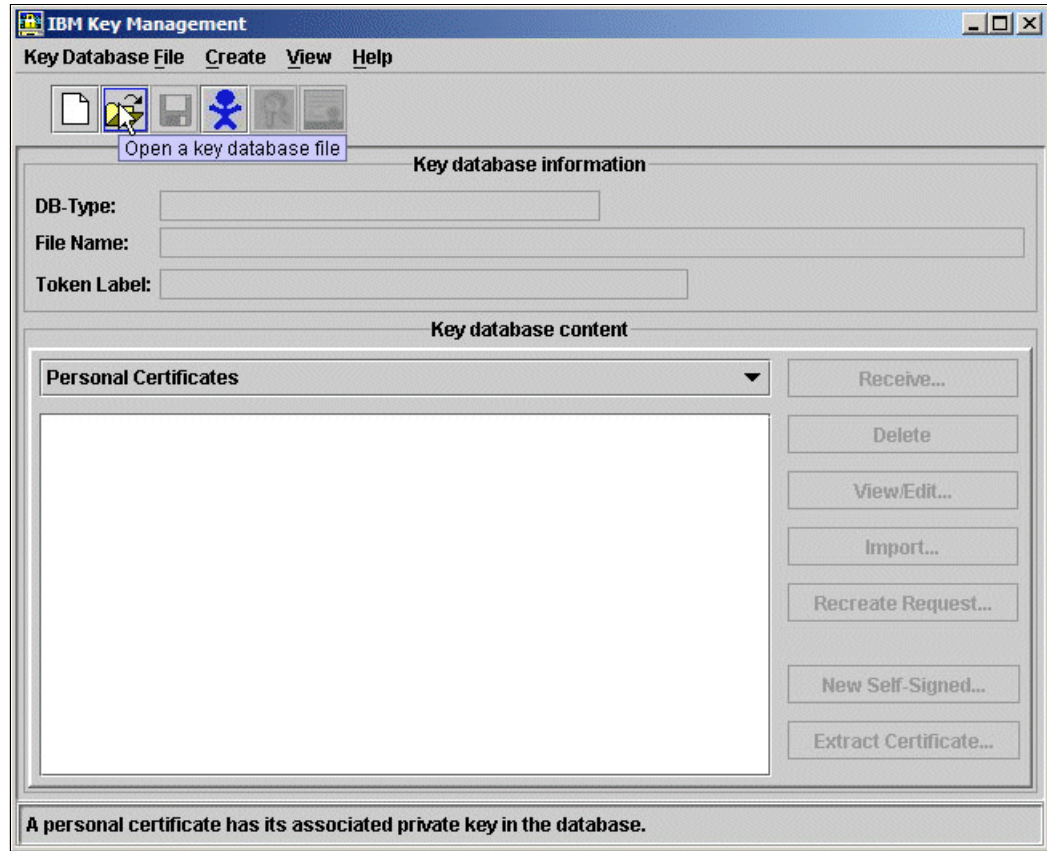


Figure 9-4 Key Management window

Figure 9-5 shows the prompt to open the key database. If the fields in the window are empty, browse to the KDB location (for more information, see “Location of Personal Communications key database in Windows” on page 299). Then, click **OK**.

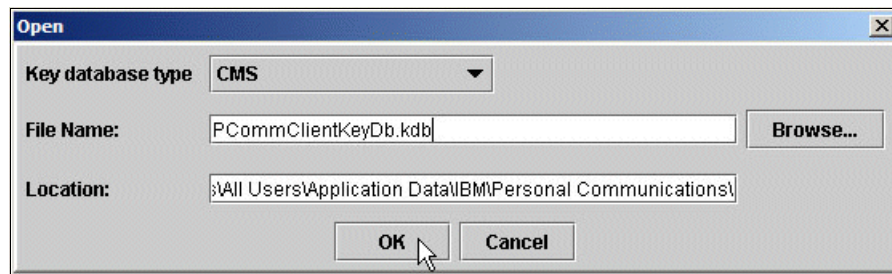


Figure 9-5 Open key database

You are now prompted for the key database password, as shown in Figure 9-6. The default password for the Personal Communications key database is pcomm.

Enter the password and click **OK**.

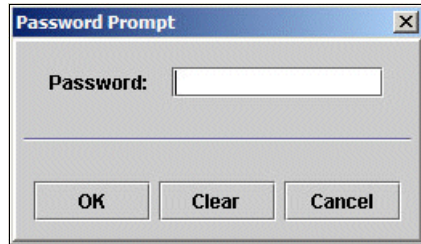


Figure 9-6 Password prompt

## Importing the PFX file into Personal Communications key database

**Note:** You import the certificate with its private key, which means you must import the certificate as a Personal certificate.

In the Personal Communications key management GUI, select **Personal Certificates** and click **Import**, as shown in Figure 9-7. This task also imports the z/VSE client certificate, which is required later for client authentication.

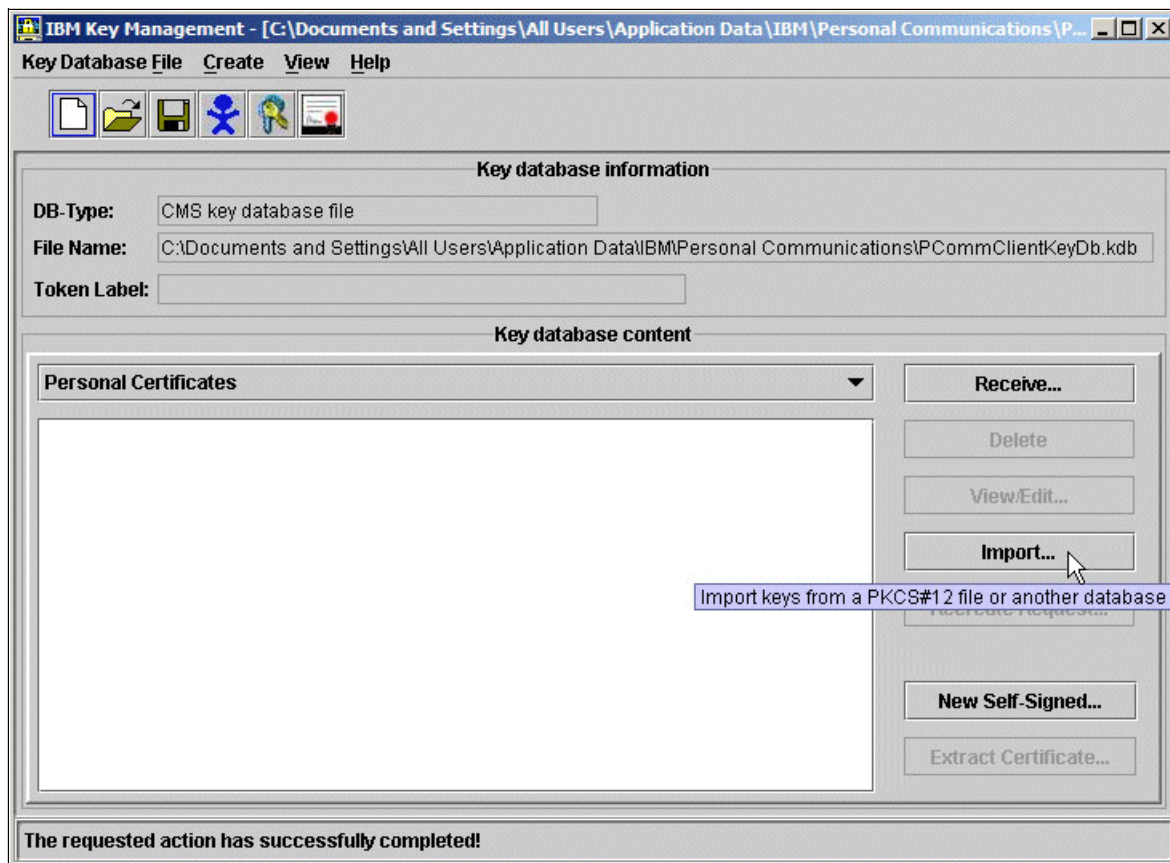


Figure 9-7 Import PFX file

Figure 9-8 shows the next window. Select key file type **PKCS12**, enter the file name and location of the z/VSE keyring file. Click **OK**.

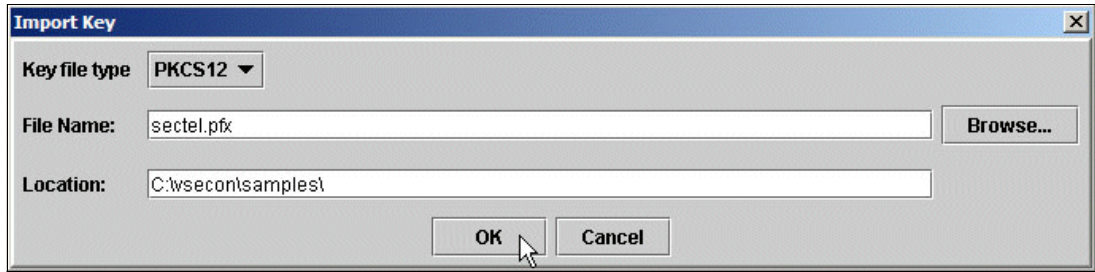


Figure 9-8 Import key

In the next window (see Figure 9-9), enter the password of the keyring file and click **OK**.

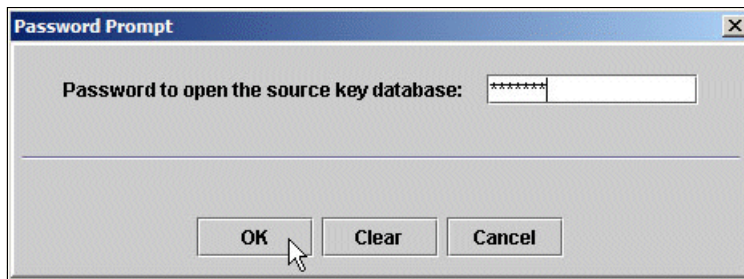


Figure 9-9 Enter password

**Note:** Exporting the z/VSE certificates in base64 text form from Keyman/VSE and using the Add function in the Personal Communications key management tool results in losing the private keys. The certificates are imported as signer certificates, rather than as Personal certificates. In this case, client authentication fails.



The key management is successfully completed, as shown in Figure 9-10.

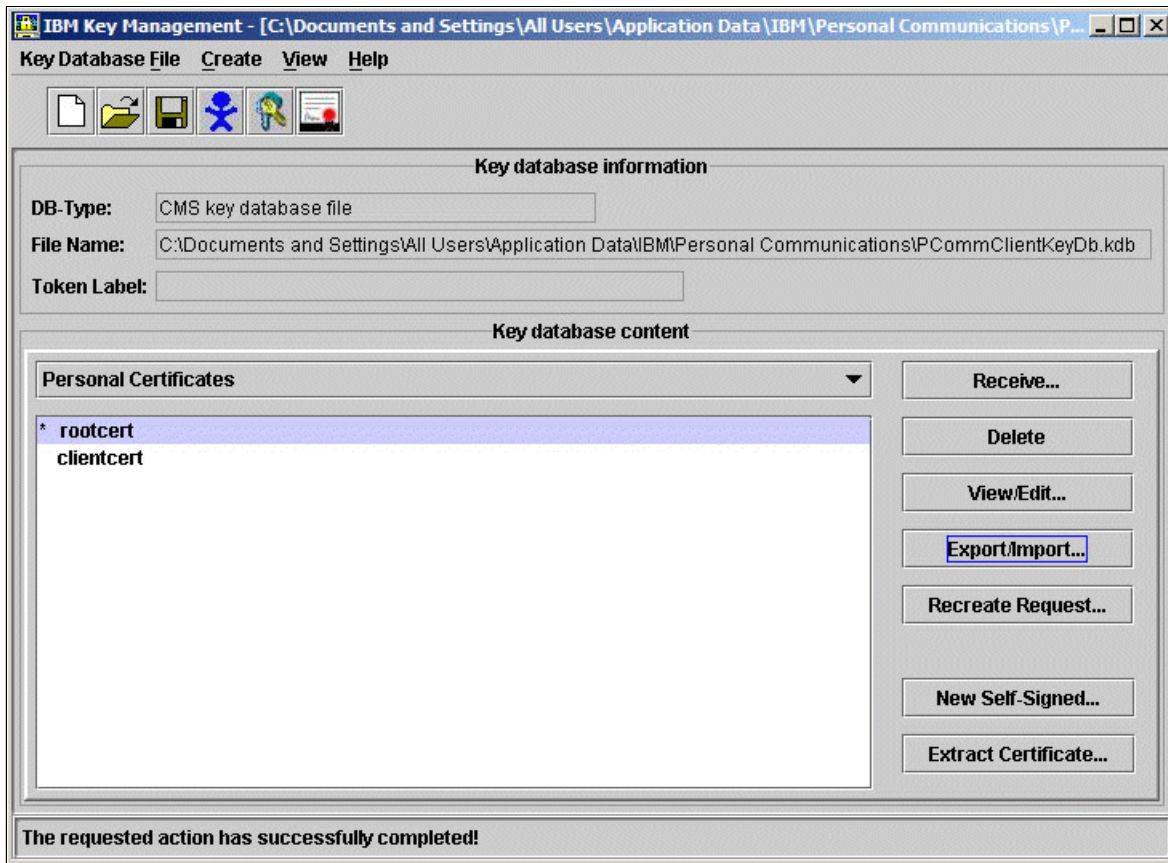


Figure 9-10 Key management successfully completed

Save the key database file and close the Personal Communications certificate management tool. A good practice is to change the default password before saving the KDB. You can change the password by clicking **Key Database File** → **Change password**.

After the password is changed, use the option Stash password to save an encrypted copy of the password in a separate file for verification of the password in the future.

To save the KDB, select **Key Database File** → **Save As** and proceed with prompts.

As shown in Figure 9-11, specify the information and click **OK**.

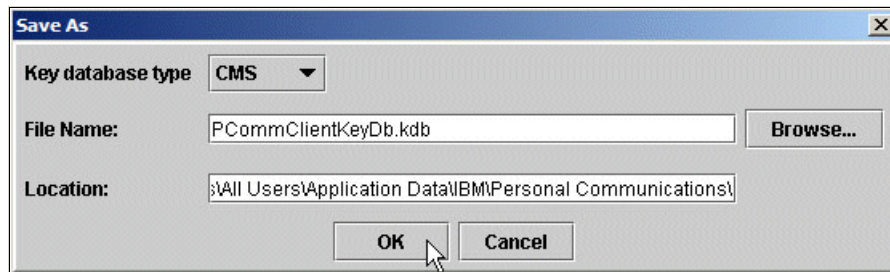


Figure 9-11 Save AS prompt

In the next window, enter the KDB password and click **OK**, as shown in Figure 9-12.

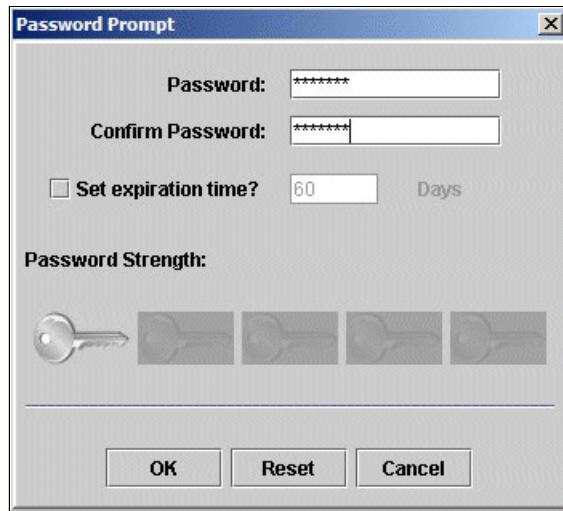


Figure 9-12 Password prompt

## 9.4.2 Starting a secure session

In the Personal Communications session window, change the port number to the secure Telnet port 992 and select the **Enable Security** option, as shown in Figure 9-13.

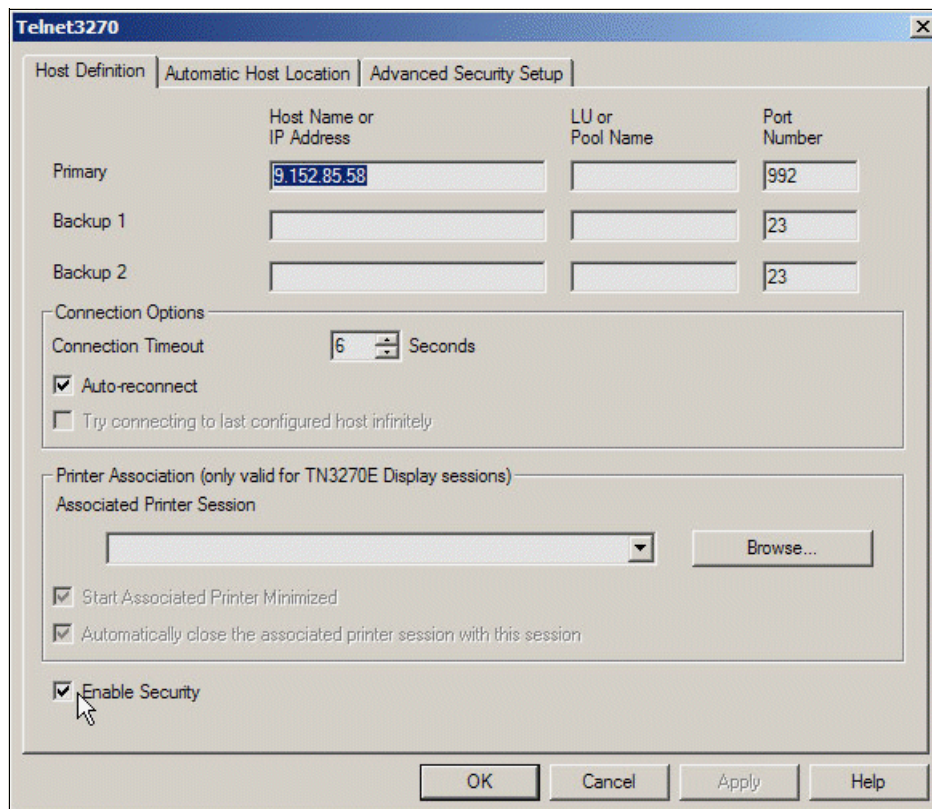


Figure 9-13 Set port and enable security

Now, the connection is secured when you connect again to z/VSE.



**Note:** If the Enable Security option cannot be selected, SSL support is not installed. Check Personal Communications Installation information to determine whether an error occurred during product installation.

Figure 9-14 shows that a closed-lock icon is displayed in the lower left corner of the session window to indicate that the session is encrypted.

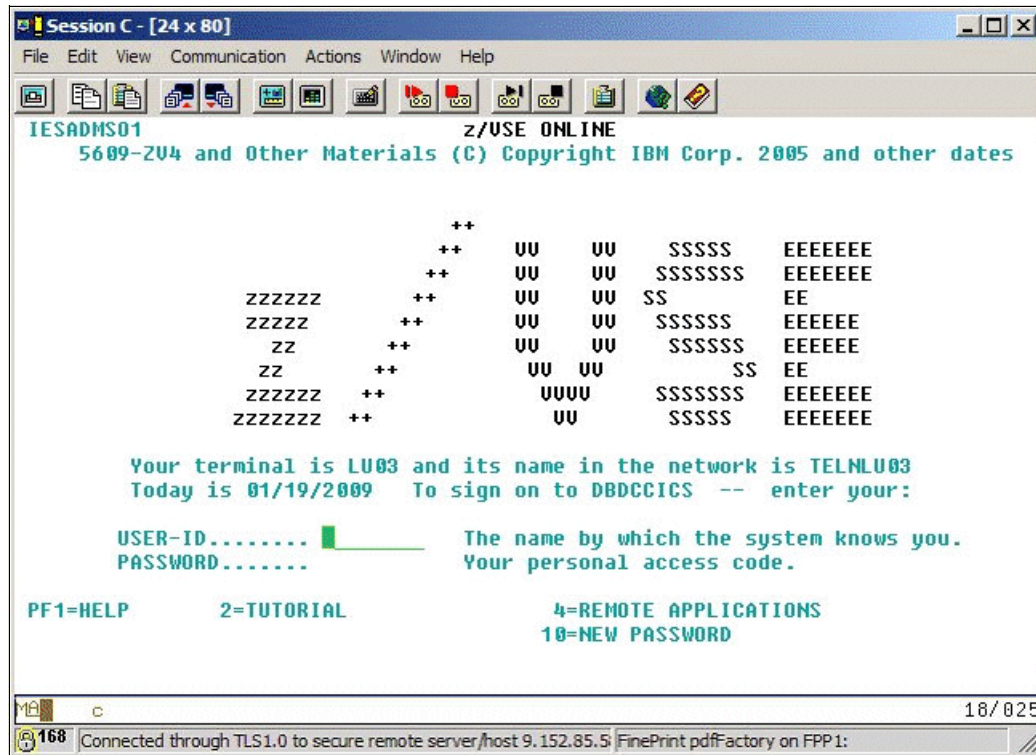


Figure 9-14 z/USE sign on panel with a secured connection

Next to the lock is the number 168, which indicates the use of triple-DES (3 x 56 bits DES) in this session.

The following values are typically used:

- ▶ 0 = no encryption
- ▶ 40 = DES with 40-bit key
- ▶ 56 = DES with 56-bit key
- ▶ 128 = AES with 128-bit key
- ▶ 168 = triple-DES

For a list of supported ciphers, see Table 4-2 on page 99.

### 9.4.3 Setting up for client authentication

SSL client authentication provides more security than server authentication because both communication partners provide a certificate to establish trust.

Complete the following steps to set up client authentication for secure Telnet:

1. Change the TLS definition on the z/USE side to enable client authentication. This process is done through the TYPE parameter of the TLS definition.

2. Ensure that your client certificate is contained in the Personal Communications key database. For more information, see 9.4.1, "Importing the z/VSE certificates into Personal Communications" on page 298. During the SSL handshake, the server requests the client's certificate.
3. Restart the TLS D in client authentication mode (TYPE=2).
4. Change the Personal Communications session definition for client authentication.

### Change TLS D for client authentication

In the TLS D definition, you must change the TYPE to 2 (client authentication):

<pre> DEFINE  TLS D, ID=TLS DTELNET,         PORT=992,         PASSPORT=23,         CIPHER=2F350A096208,         CERTLIB=CRYPTO,         CERTSUB=KEYRING,         CERTMEM=SECTELN,         TYPE=2,         MINVERS=0300,         DRIVER=SSLD </pre>	<pre> ID of this SSL/TLS daemon Secure Telnet port Port data is passed to Allowed cipher suites Library name Sublibrary name Member name SSL client authentication Minimum version required Driver phase name </pre>
---	--

### Restart TLS D for client authentication

Restart the TLS D with a changed TYPE parameter, as shown in Example 9-6.

#### Example 9-6 Restart TLS D

---

```

101 delete tlsd,id=TLS DTELNET
F7-0101 IPN300I Enter TCP/IP Command
F7 0097 00E1: TLS903I Daemon Shutdown TLS Id:TLS DTELN
101 DEFINE  TLS D, ID=TLS DTELNET, PORT=992, PASSPORT=23, CIPHER=2F350A096208, -
F7-0101 IPN300I Continue TCP/IP Command
101 CERTLIB=CRYPTO, CERTSUB=KEYRING, CERTMEM=SECTELN, TYPE=2, -
F7-0101 IPN300I Continue TCP/IP Command
101 MINVERS=0300, DRIVER=SSLD
F7-0101 IPN300I Enter TCP/IP Command
F7 0097 0212: TLS900I Daemon Startup Transport Security Layer SSLD
      =81640040

```

---

Example 9-7 shows how you verify that the TLS D is now started for client authentication.

#### Example 9-7 Verify TLS D status

---

```

101 q tlds
F7 0097 IPN253I << TCP/IP TLS Daemons >>
F7 0097 IPN617I  ID:  TLS DTELNET Cipher: 2F350A096208
F7 0097 IPN618I  Port: 992  Passport: 23 Type: Server_Auth
F7 0097 IPN619I  Driver: SSLD  Minimum version: 0300

```

---

Do not be confused: Type Server\_Auth is displayed for client authentication, whereas only Server is displayed for server authentication.

## Setting up Personal Communications session for client authentication

Enable client authentication in the Personal Communications session. Select the **Advanced Security Setup** tab. Then, select **Send Personal Certificate to Server if Requested** and **Select or Prompt for Personal Client Certificate**. Then, click **Select now**, as shown in Figure 9-15 on page 307.

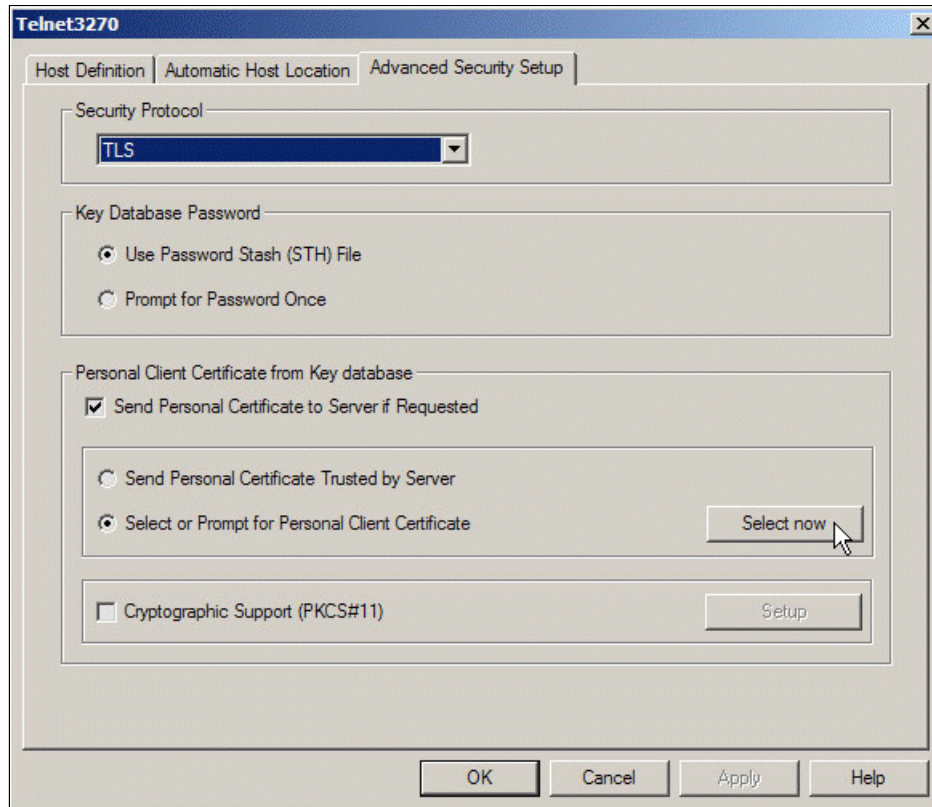


Figure 9-15 Set up Personal Communications for client authentication

Figure 9-16 shows the next window for the Personal Communications key database password and selecting the z/VSE client certificate. You might need to enter the password, leave the prompt by clicking **OK**, and then, reopen the prompt to view the list of certificates. The prompt remembers the password for subsequent use. When you are done, click **OK**.



Figure 9-16 Enter the key database password

**Note:** Your certificates might not appear in the list for the following reasons:

- ▶ They are not correctly imported into the Personal Communications key database with their private keys.
- ▶ You entered the incorrect password.

### Connect by using client authentication

After leaving the session configuration boxes by clicking **OK**, you are now ready to connect to the z/VSE TLSO with client authentication.

The z/VSE sign-on panel in a Personal Communications session with client authentication is identical to the panel shown in Figure 9-14 on page 305.

## 9.4.4 Taking a Personal Communications trace

If you encounter problems while attempting to connect to z/VSE, you might want to trace and analyze the problem. Personal Communications provides a trace function that is activated with the session window by selecting **Actions** → **Launch** → **Trace Facility**.

In the trace window, specify a TCP/IP trace, as shown in Figure 9-17.

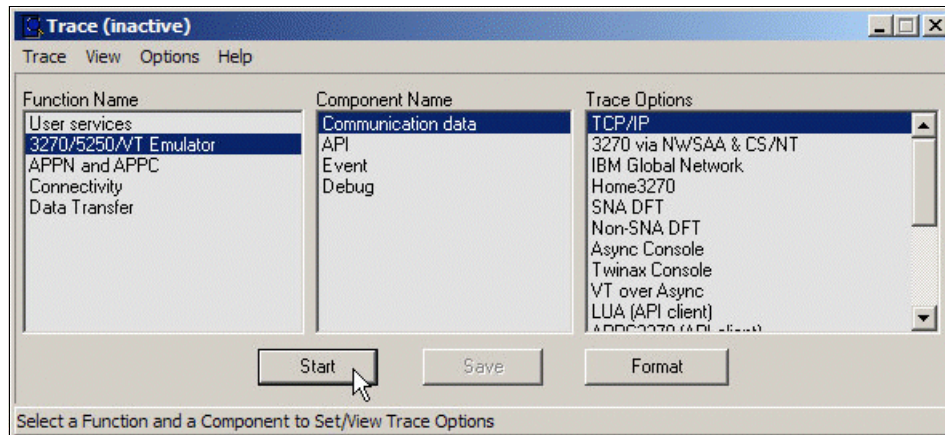


Figure 9-17 Personal Communications trace settings

Click **Start** and try to connect again.

Then, stop and format the trace. Place the trace output in a folder that is easy to find (see Figure 9-18).

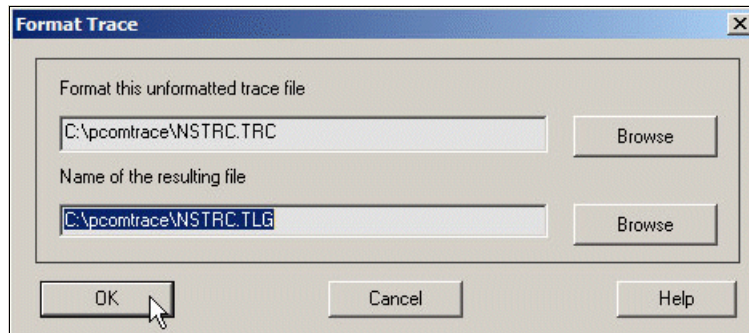


Figure 9-18 Specify Personal Communications trace file



You can now use the Personal Communications trace viewer to view the formatted trace, as shown in Figure 9-19.

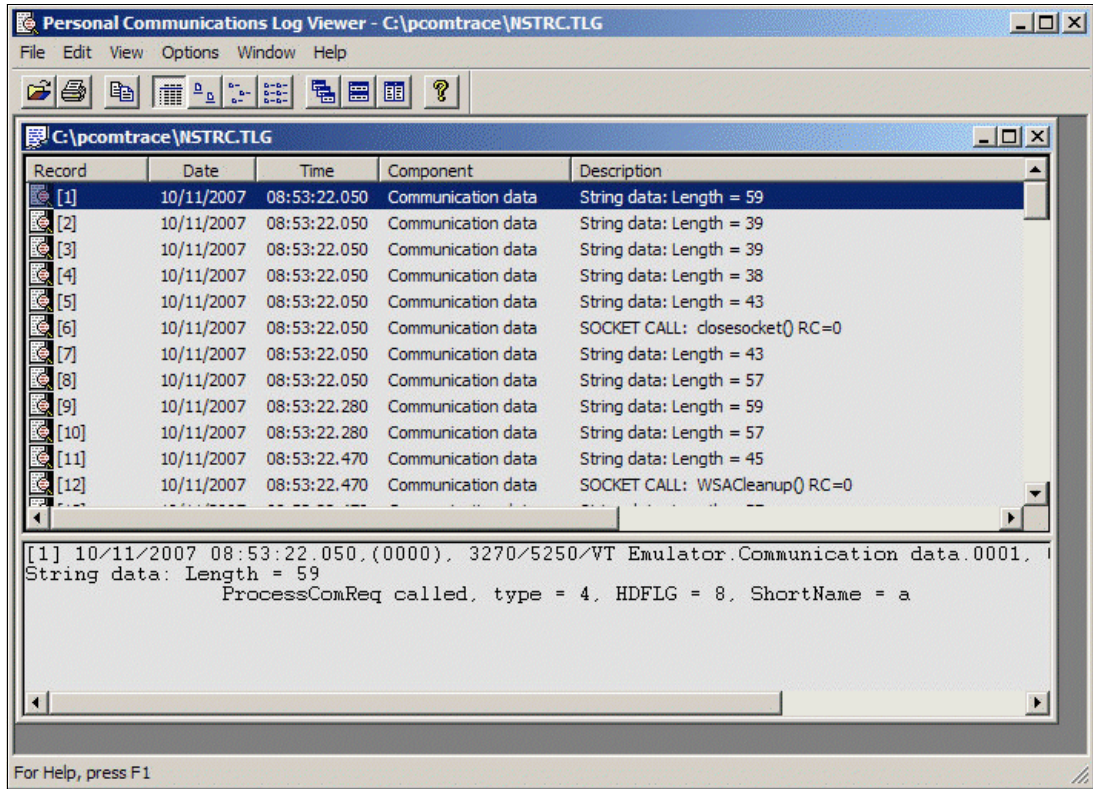


Figure 9-19 Personal Communications log viewer

## 9.5 Client setup with Attachmate EXTRA! X-treme

This section describes the secure Telnet setup with Attachmate EXTRA! X-treme V9 Evaluation. The evaluation version of the emulator was downloaded from:

<http://www.attachmate.com/en-US/Evals/Evaluate.htm>

The main difference between Attachmate and Personal Communications is that Attachmate uses the certificate store of Windows instead of providing its own certificate management tool. You see later that certain differences also exist in the session setup.

## 9.5.1 Importing certificates into the Windows certificate store

To import the certificates that were created as described in 5.1.2, “Creating the z/VSE key and certificates” on page 200, open the Windows Control Panel and double-click **Internet Options**. In the Content tab, click **Certificates**, as shown in Figure 9-20.

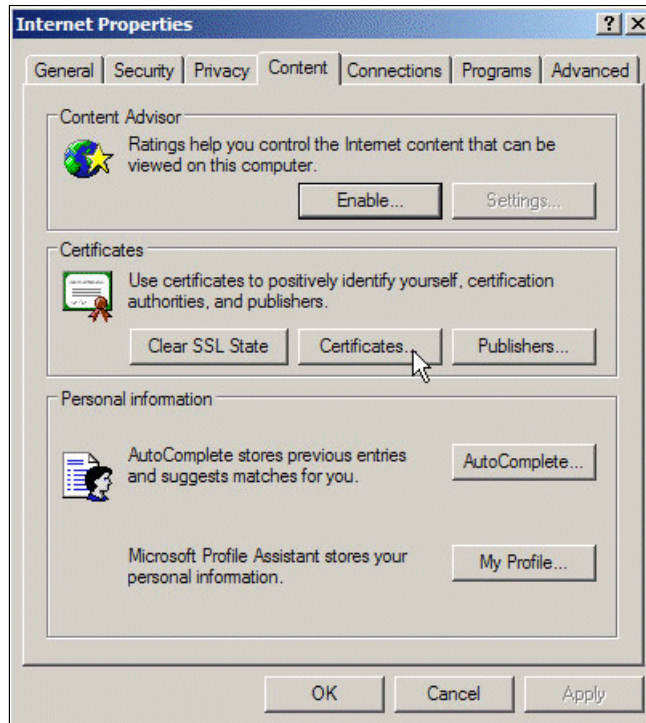


Figure 9-20 Change Internet properties

In the next window, click **Import** and follow the Import Wizard prompts.

As shown in Figure 9-21 on page 311, file secte12.pfx is a new keyring file with a server certificate where the Common Name is identical to the z/VSE IP address. This file is the original file that you created as described in 5.1.2, “Creating the z/VSE key and certificates” on page 200. It does not fulfill this condition.

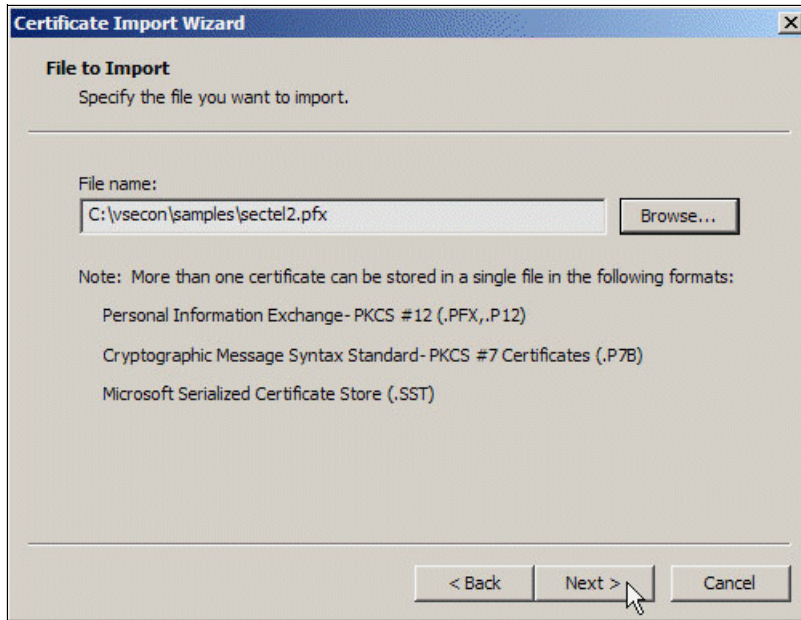


Figure 9-21 Specify import file name

In the window, select the **Place all certificates into the following store** option to place the z/VSE root certificate into the Trusted Root Certification Authorities store, as shown in Figure 9-22.

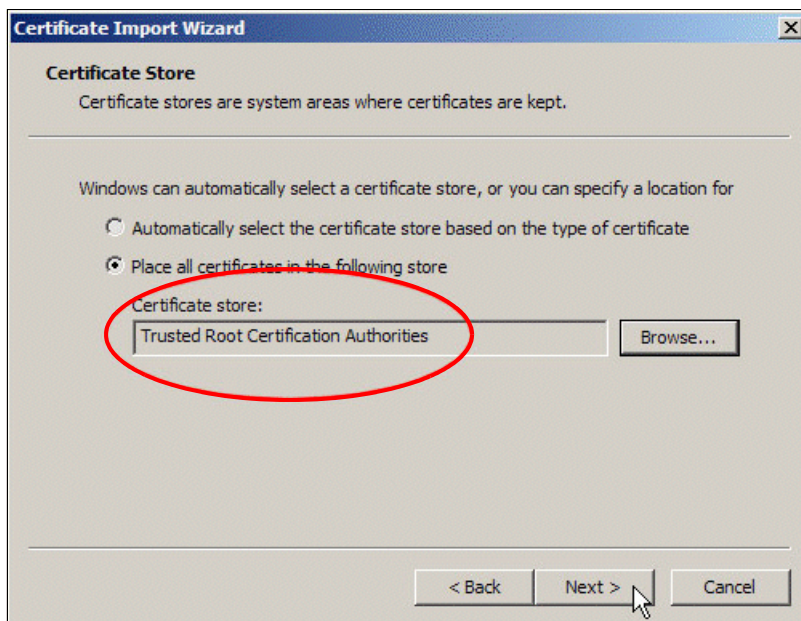


Figure 9-22 Import root certificate

If you select the first option, Windows places the certificates into the Personal certificate store. This option has the same effect as not importing the root certificate at all. For more information, see “Missing root certificate” on page 314.

Click **Next**.



The z/VSE root certificate is now available in the Windows certificate store (see Figure 9-23). The contained server certificate was not imported into this store because it is not self-signed or signed by a known CA.

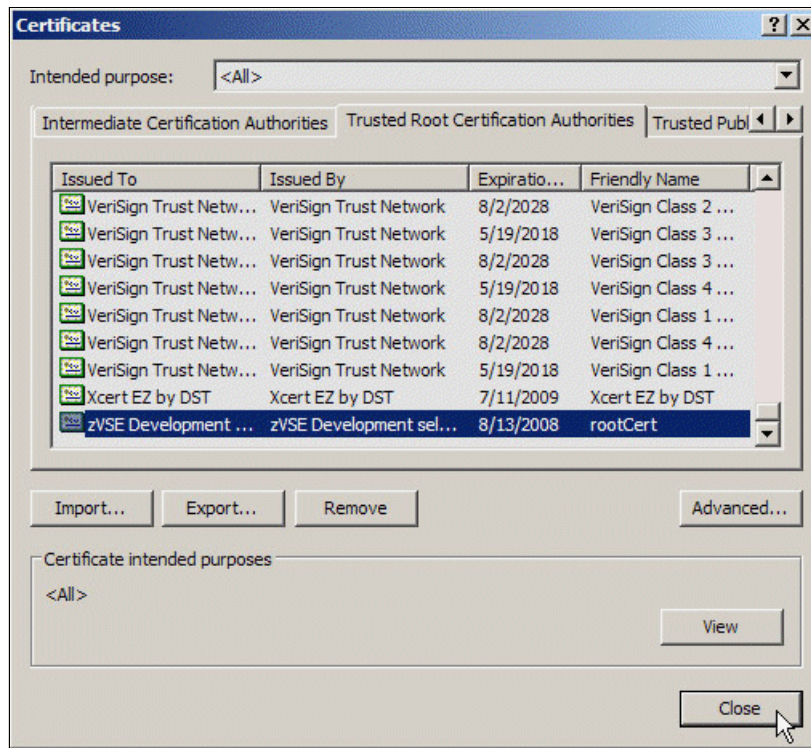


Figure 9-23 List of certificates

## 9.5.2 Attachmate EXTRA! session setup

Start a new Attachmate EXTRA! session and enter the IP address of your z/VSE system and the port number of the secure Telnet port.

For the Terminal/device type field, select IBM-3278 (see Figure 9-24 on page 313). Select the **Use Microsoft security implementation** option to access the Windows certificate store. Also, the level of encryption must be set to SSL V3.0.

**Note:** Make the following selections for unsecure connections:

- ▶ Change the port back to your unsecure Telnet port (normally 23).
- ▶ Set the Level of encryption to **None**.
- ▶ Deselect the **Use Microsoft security implementation** option.



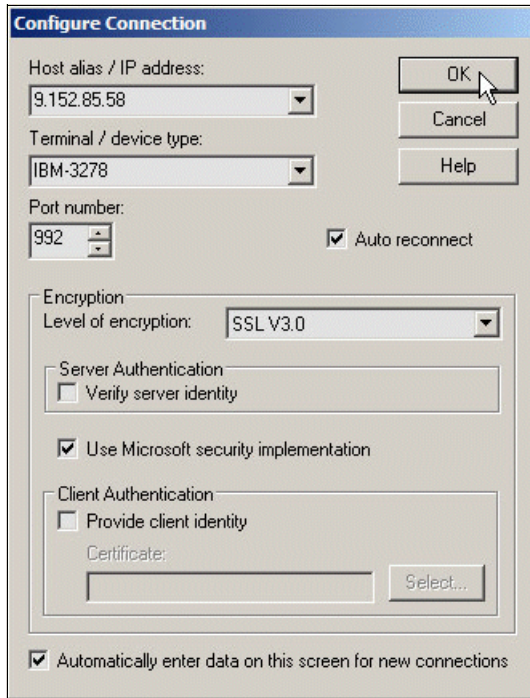


Figure 9-24 Configure connection

You can define another level of security by also selecting the **Verify server identity** option. In this case, the common name of your z/VSE server certificate must be identical to the z/VSE IP address.

Now, you can connect to z/VSE through secure Telnet to get the z/VSE sign-on window, as shown in Figure 9-25.

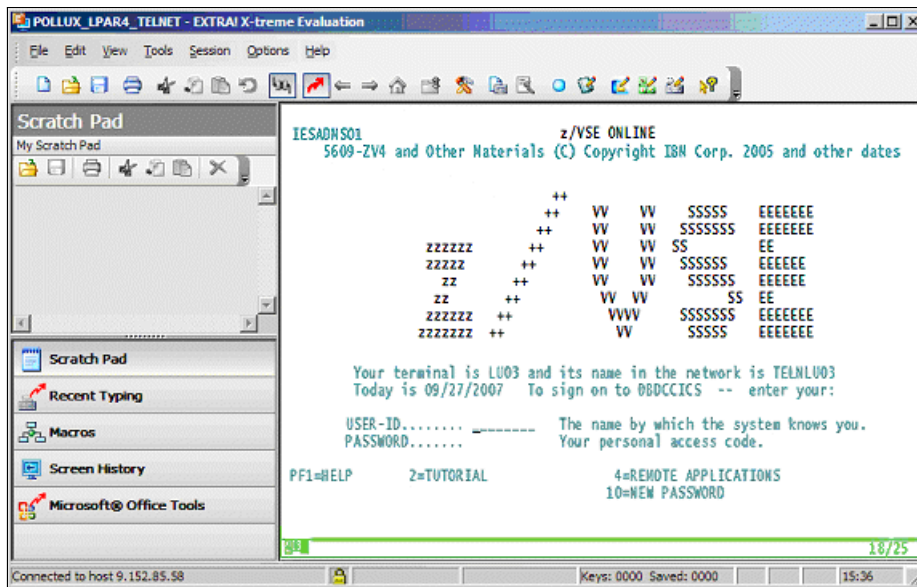


Figure 9-25 z/VSE sign-on panel

The closed lock icon that is shown in Figure 9-25 indicates that the session is now encrypted.

### 9.5.3 Viewing the log

To view the Attachmate event log, start application Status App in the Attachmate EXTRA! Program group. The log provides more information about the SSL handshaking process and the used cipher suite for this session.

During the test setup, we made the observations that are described in this section.

#### Problem with option Verify server identity

In the audit log, as shown in Figure 9-26, message Certificate signature does not verify from host 9.152.85.58 indicates that option Verify server identity was active, but the Common Name of the received server certificate was not identical to the server's IP address. The issue was resolved after new certificates were set up that included this precondition fulfilled.

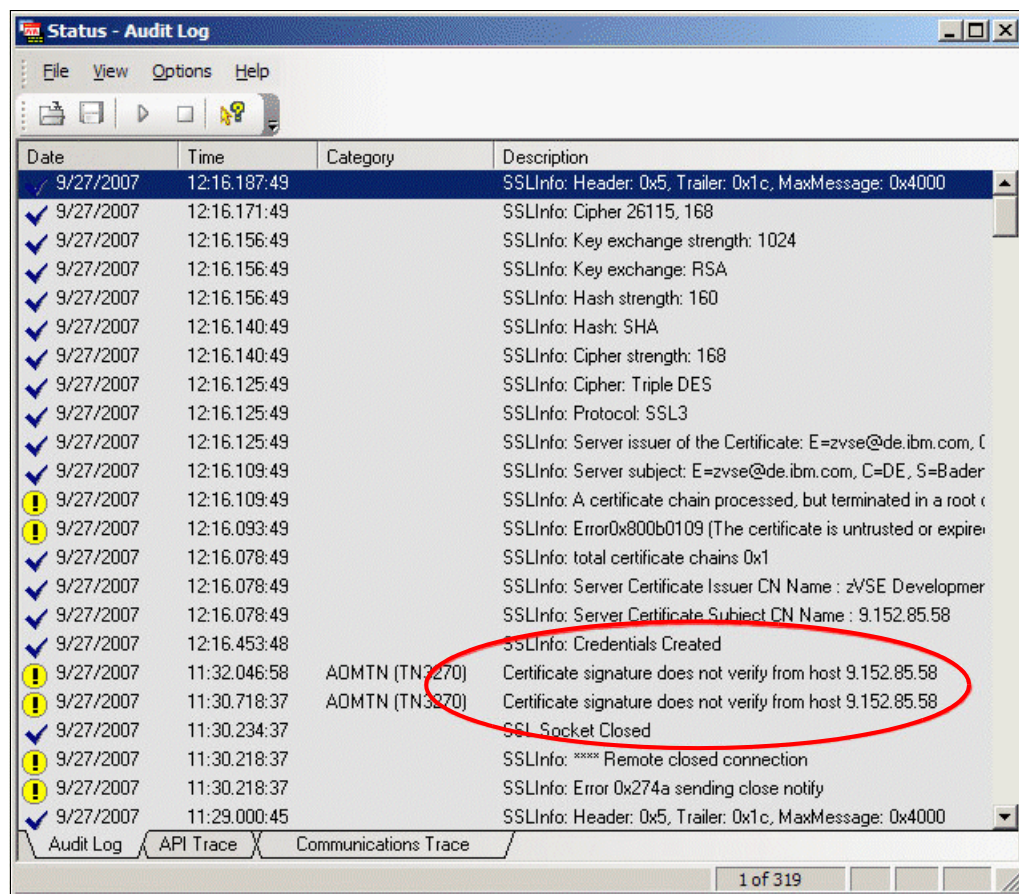


Figure 9-26 Audit log

#### Missing root certificate

The connection is established even when the z/VSE root certificate is not contained in the Windows certificate store. The following messages show that verifying the received server certificate was not possible, although the connection was established:

SSLInfo: A certificate chain processed, but terminated in a root certificate which is not trusted by the trust provider.

SSLInfo: Error0x800b0109 (The certificate is untrusted or expired!) returned by CertVerifyCertificateChainPolicy!

After importing the keyring file into the Windows certificate store, as described in 5.1.2, “Creating the z/VSE key and certificates” on page 200, these messages do not display.

## 9.5.4 Setting up for client authentication

To establish client authentication with Attachmate EXTRA!, you must perform the following tasks (assuming that the TLS D is still running in client authentication mode):

1. Change the TLS D definition on the z/VSE side to enable client authentication. This process is done through the TYPE parameter of the TLS D definition. For more information, see “Change TLS D for client authentication” on page 306.
2. Import the Client Certificate into the Windows key database.
3. Change the Attachmate session definition for client authentication.

### Importing the client certificate into the Windows key database

Open the Windows Control Panel and double-click **Internet Options**. In the Content tab, click **Certificates**. In the next window, click **Import** and follow the Certificate Import Wizard prompts.

Now, select the **Automatically select the certificates store based on the type of certificate** option to select the certificate store based on the type of the certificate, as shown in Figure 9-27.

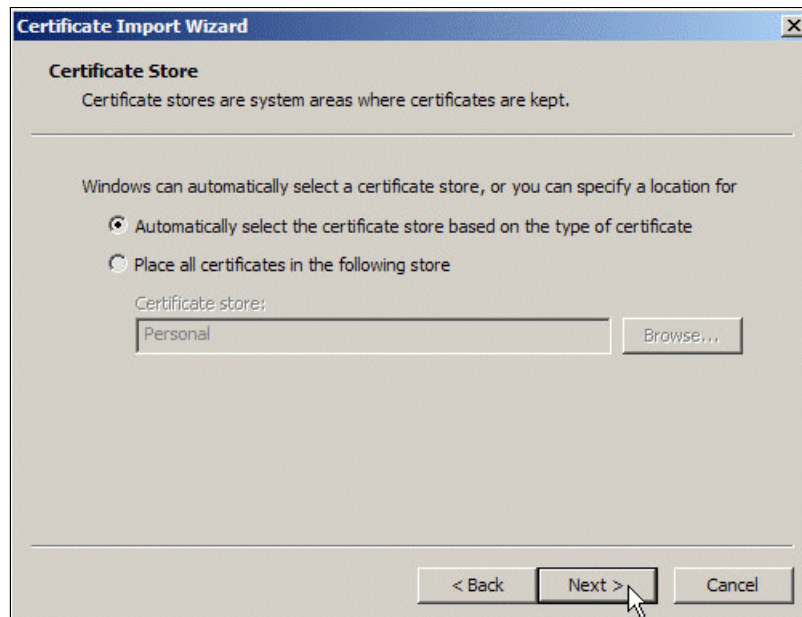


Figure 9-27 Certificate Import Wizard window

The certificates are now stored in the section Personal. Because we imported the root certificate, it now appears twice in the store. Remove it from the Personal store, as shown in Figure 9-28.

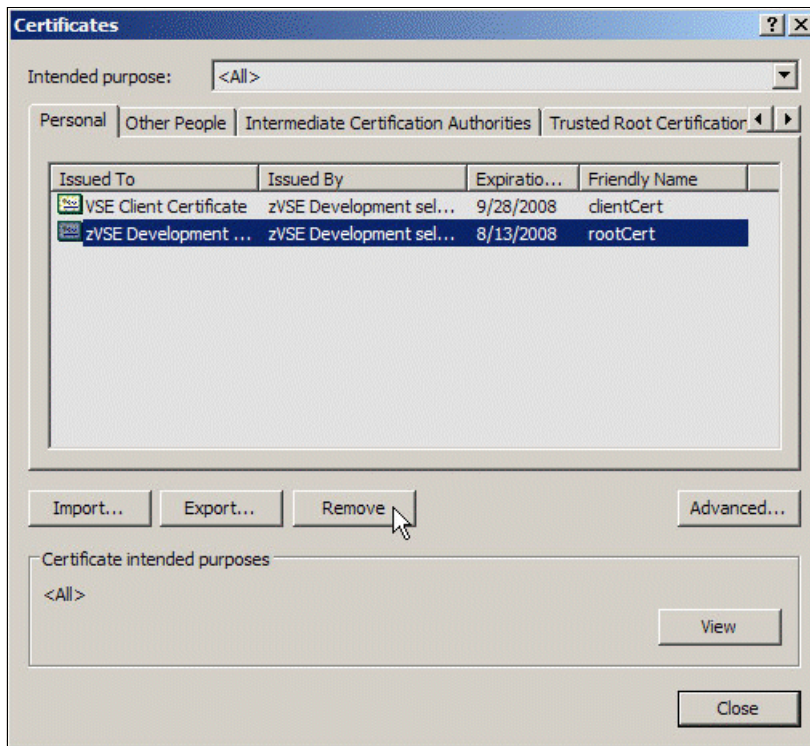


Figure 9-28 List of certificates



## Changing the Attachmate session for client authentication

In the Attachmate session setup window, select **Provide client identity** and click **Select** (see Figure 9-29).

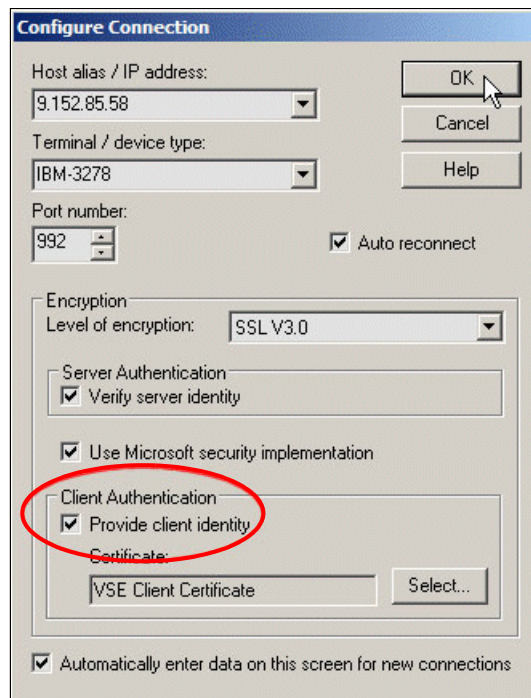


Figure 9-29 Configure connection

The available certificates are listed in the drop-down list. Select the z/VSE client certificate for use by this session, as shown in Figure 9-30.

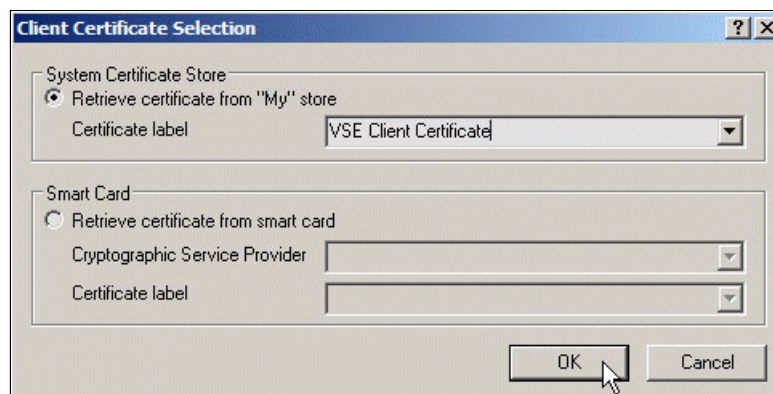


Figure 9-30 Client certificate selection

After you apply these changes, the session uses client authentication after restart.





# Secure File Transfer Protocol

File Transfer Protocol (FTP) is the standard protocol for exchanging and manipulating files over a TCP/IP network. Typically, an FTP client connects to an FTP server to send, receive, or manipulate files on that server.

The only security that is provided is the optional use of a user ID and a password during the start of the FTP connection. All commands and data, including the user ID and password, are passed across the network in the clear, which is a security issue.

The SecureFTP for VSE feature of TCP/IP for VSE/ESA provides user authentication, confidentiality, and data integrity by using digitally signed certificates, data encryption, and secure hash functions. These cryptographic functions provide authentication, privacy, and integrity for commands and data that is transmitted by using the FTP protocol by implementing SSL and TLS into FTP clients and servers that are running on the z/VSE platform. z/VSE can exchange data through secured FTP connections with other z/VSE systems, and with any other platforms that support SSL/TLS secured FTP.

In this chapter, we describe the setup of secure FTP in scenarios with z/VSE acting as server or as client. We also describe considerations for when firewalls are used.

This chapter includes the following topics:

- ▶ 10.1, “Introduction” on page 320
- ▶ 10.2, “z/VSE as FTP server” on page 320
- ▶ 10.3, “z/VSE as FTP client” on page 324
- ▶ 10.4, “Considerations for firewalls” on page 340
- ▶ 10.5, “Observations” on page 343

## 10.1 Introduction

Secure FTP requires RSA key pairs and digital certificates on the server and client sides. For more information about creating keys and certificates, see 5.1, “Generating the server key and certificates” on page 198.

In the examples, we use the following software:

- ▶ z/VSE V4R2
- ▶ TCP/IP for VSE/ESA 1.5F
- ▶ z/VSE Connector Server as part of z/VSE V4R2
- ▶ Java 1.6 from Sun Microsystems
- ▶ Keyman/VSE, update from 08/2007
- ▶ FileZilla server version 0.9.23 beta
- ▶ FileZilla client version 2.2.30
- ▶ Symantec Client Firewall Version 8.7.4.97
- ▶ OpenSSL Light 0.98

With TCP/IP for VSE/ESA 1.5 Service Pack F, the following fixes are necessary for secure FTP:

- ▶ 204, 206, 244, 247, 251, 252, 253 shipped with APAR PK77248
- ▶ 264, if you are using FileZilla client and TLS 1.0 for connecting to z/VSE

## 10.2 z/VSE as FTP server

Setting up secure FTP with z/VSE as the server is similar to setting up SSL for use with the z/VSE Connector Server or CICS Web Support. This process is described in Chapter 5, “Secure Sockets Layer with z/VSE” on page 197. In the following sections, we describe how secure FTP is set up by using z/VSE as the server.

### 10.2.1 Set up and start the z/VSE FTP server

The following types of FTP servers in z/VSE are available:

- ▶ External FTP daemons, which run in a separate z/VSE partition
- ▶ Internal FTP daemons, which run as a subtask in the TCP/IP partition

The JCL that is shown in Example 10-1 starts an external SSL-enabled FTP server on z/VSE.

*Example 10-1 Start external FTP server*

---

```
* $$ JOB JNM=FTPSERV,CLASS=8,DISP=D
* $$ LST CLASS=A
// JOB FTPSERV
// OPTION LOG,NOSYSDMP
// OPTION SYSPARM='00'
/* LIBDEF *,SEARCH=(PRD1.BASE)
// EXEC FTPBATCH,SIZE=FTPBATCH,PARM='UNIX=YES,FTPDPOR=990,SSL=SERVER'
SET DIAGNOSE ON
SET SSL PRIVATE CRYPTO.KEYRING.SFTP1024 NOCLAUTH
/*
/&
* $$ E0J
```

---



When running the FTP server, various console messages are issued, as shown in Example 10-2.

*Example 10-2 Messages from the FTP server*

---

```
F8 0008 // JOB FTPSERV
      DATE 08/07/2007, CLOCK 12/03/38
F8 0118 FTP302I Connected to TCP/IP Sysid 00 in F7 from F8
...
F8 0118 FTP900I FTP Daemon: FTPBSRVR listening on 9.152.84.147,990
F8 0118 FTP304I FTPX1000 subtask is running 005044E0
F8 0118 FTP314I Command connection SSL secured, data connection:PRIVATE
F8 0118 FTP306I Commands from SYSIPT COMPLETED
```

---

You can check the listening port here again. In this example, the server listens on port 990.

The following TCP/IP command starts an internal FTP server with the same properties:

```
DEFINE FTPD, ID=FTPDSL, PORT=990, COUNT=1, TIMEOUT=9000, UNIX=YES, -
      DRIVER=FTPDAEMN, SSL=YES, SSLKEY=CRYPTO.KEYRING.SFTP1024, -
      SSLVER=SSLV3, SSLCIPHER=ALL, SSLDATACONN=PRIVATE
```

**Note:** You define the FTP daemon with UNIX=YES or UNIX=BIN to make the z/VSE file system accessible for the FileZilla FTP client. Without UNIX mode, the z/VSE file system appears as one *<Directory>* entry, which cannot be accessed by the FTP client.

UNIX=BIN has the advantage that it also allows true binary transfers to be used. Without it, binary transfers might occur in ASCII mode. The UNIX=BIN allows binary transfers in UNIX emulation mode.

## 10.2.2 z/VM considerations

The z/VM FTP client encounters a problem when the FTPD is defined with UNIX mode. All transferred data records are truncated to 80 characters. Therefore, when you are also using z/VM to access z/VSE through FTP, you define a second FTPD on z/VSE with UNIX=NO. Messages similar to the messages that are shown in Example 10-3 appear on the z/VSE console when the problem occurs.

*Example 10-3 Error messages*

---

```
FTP928E Record 1 larger than max(80) for ptf/file UNIX=YES
IPN549E IPCCDROP error: Invalid ownership 01
```

---

### 10.2.3 Connect to z/VSE by using an FTP client

In the example that is described in this section, we use the FileZilla FTP client. After connecting to the z/VSE FTP server, the z/VSE directory listing is retrieved, as shown in Figure 10-1.

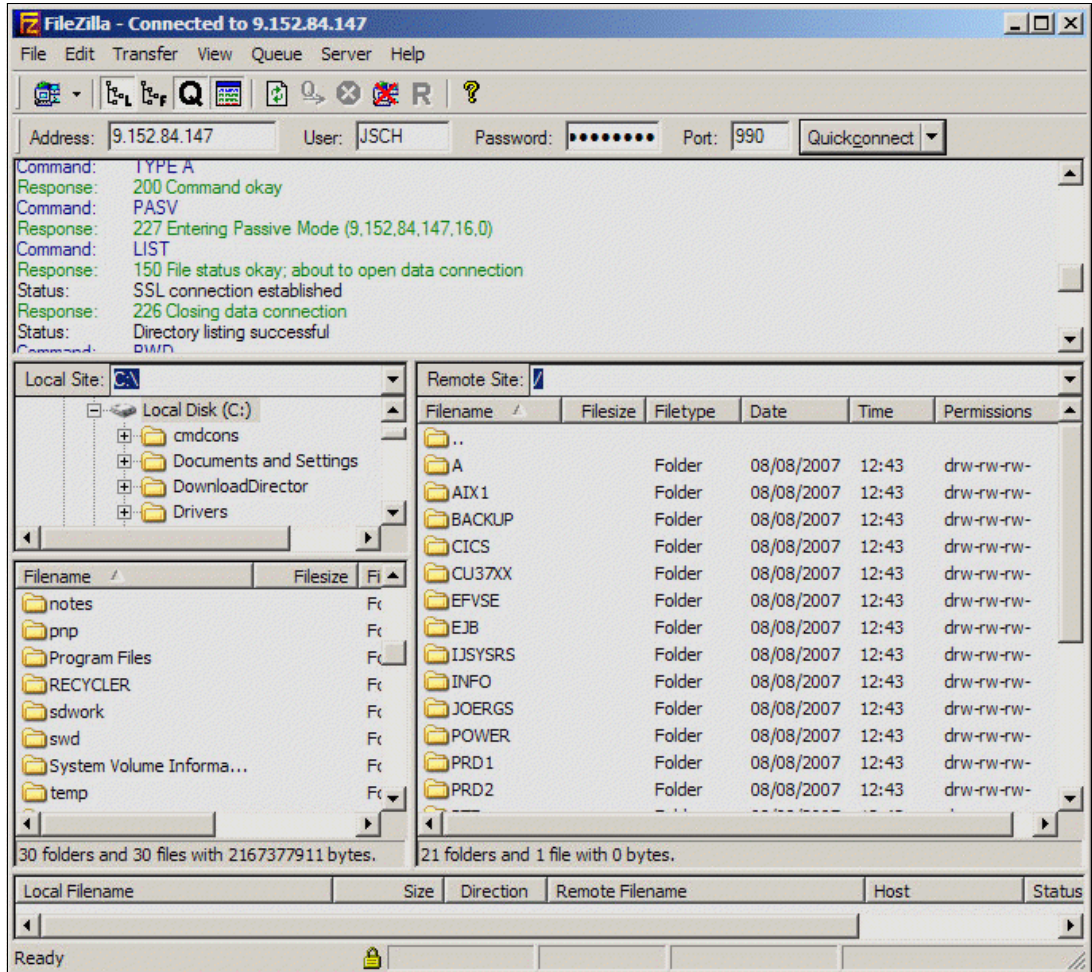


Figure 10-1 FileZilla FTP client

On the z/VSE side, the FTP daemon indicates which ciphers are used in the current connection.

Cipher 0035 in Example 10-4 means that transferred data is encrypted by using AES-256.

#### Example 10-4 Messages from FTP daemon

```
F7 0098 0005: FTP900I FTP Daemon: FTPDSSL listening on 9.152.84.147,990
F7 0098 0034: FTP922I Control connection using SSL TLSV1 Cipher=0035
(01670000)
F7 0098 0034: FTP909I JSCH in session with 9.152.216.58,1694
F7 0098 0034: FTP910I Data connection open 9.152.216.58,1695 (4101)
F7 0098 0034: FTP922I Data connection using SSL TLSV1 Cipher=0035 (0166F000)
```

## 10.2.4 Transferring the certificate to the client side

The FileZilla client allows importing the server certificate to its certificate store while opening a secure FTP session (transferring the certificate in advance is not necessary). Accept the server certificate permanently when the Accept certificate message appears on the client side.

For example, to permanently add this certificate to the server, select the **Always trust this certificate** option and click **Accept**, as shown in Figure 10-2. If you do not select the option, the message appears again when you connect to z/VSE again.

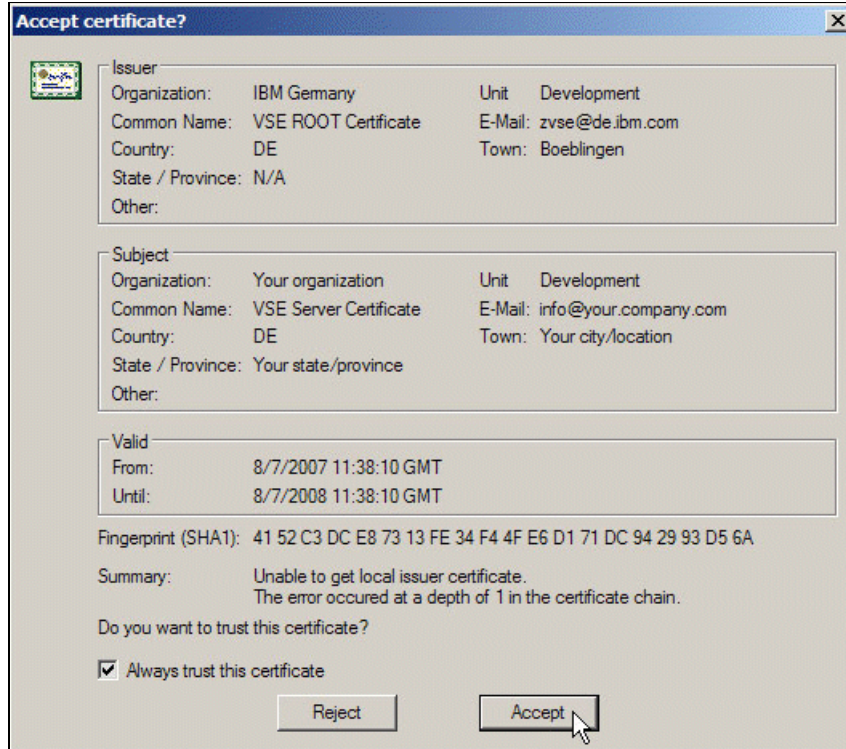


Figure 10-2 Accepting certificate window

## 10.3 z/VSE as FTP client

Setting up secure FTP with z/VSE as the client and a non-z/VSE system as server requires some configuration effort on an FTP server side.

In the following sections, we use the Open Source FTP server FileZilla and the vsftpd server on Linux as examples of non-z/VSE FTP servers.

### 10.3.1 Sample setup with FileZilla server

The FileZilla server is available for download from this website:

<http://filezilla-project.org/>

On Windows, the FileZilla server is installed as a Windows service and is started automatically when Windows is started. You might want to prevent the automatic start for security reasons. To configure the server for manual startup, open the Windows Control panel, select **Administrative Tools** → **Services**, and change the startup option for the FileZilla server to manual startup.

#### Generating the server key and certificate

This function is provided by the FileZilla server. Start the server and open the FileZilla server interface. Then, select **Edit** → **Settings**, as shown in Figure 10-3, to open the Settings window.

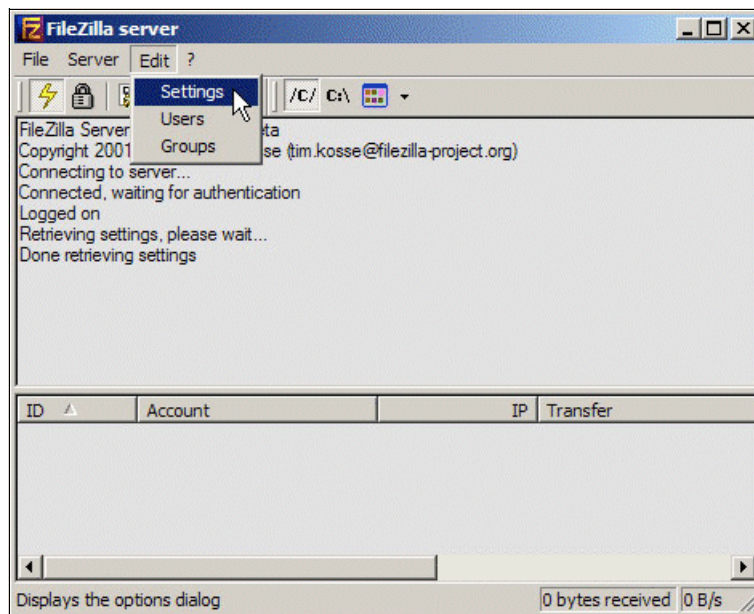


Figure 10-3 FileZilla server



On the FileZilla Server Options panel, select **SSL/TLS Settings** and specify the file name (or file names) for the server key and the certificate file, as shown in Figure 10-4. Both items can be stored into the same file. You also must specify a password for the key file. This password is required later when importing the key file into a web browser for further use; for example, in an SSL setup.

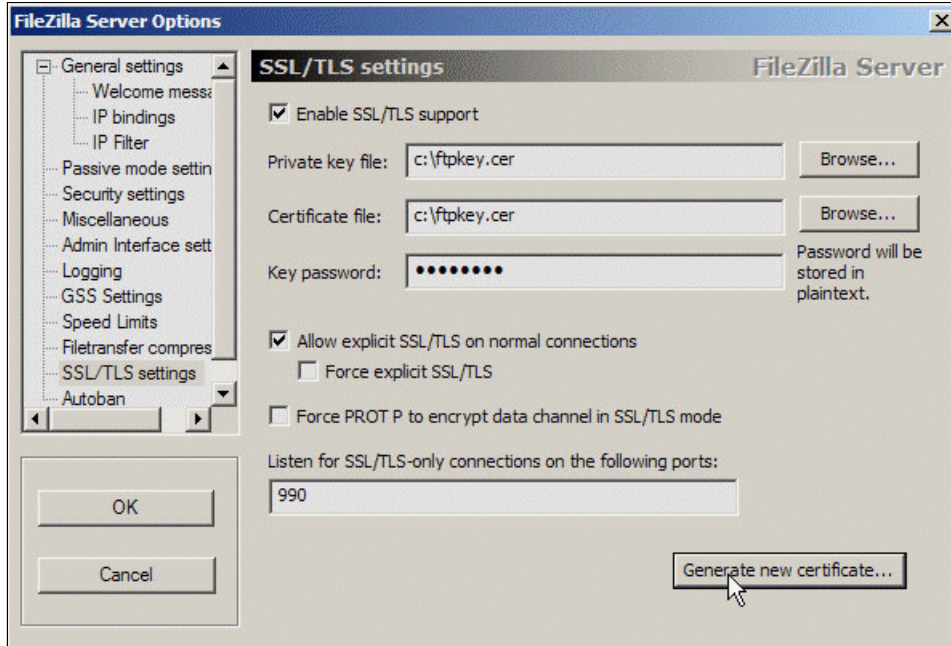


Figure 10-4 FileZilla Server Options

Then, click **Generate new certificate**.

In the next window, complete the required information and click **Generate certificate**, as shown in Figure 10-5.

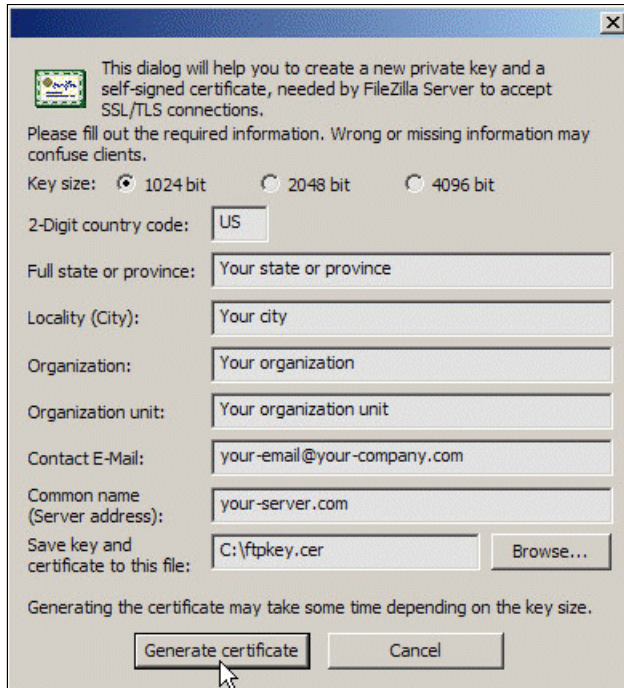


Figure 10-5 Generating certificate

**Note:** The created certificate features a validity period of one year. FileZilla does not allow specifying any other validity period. For more information, see “Using OpenSSL to create the server key and certificate” on page 328.

As shown in Figure 10-6, the certificate was created. Click **OK** to return to the Server Options window.

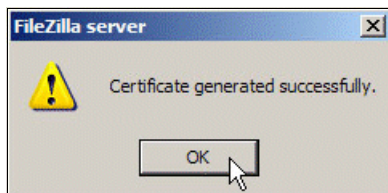


Figure 10-6 Certificate generated

In the FileZilla Server Options panel, click **OK**, as shown in Figure 10-7.

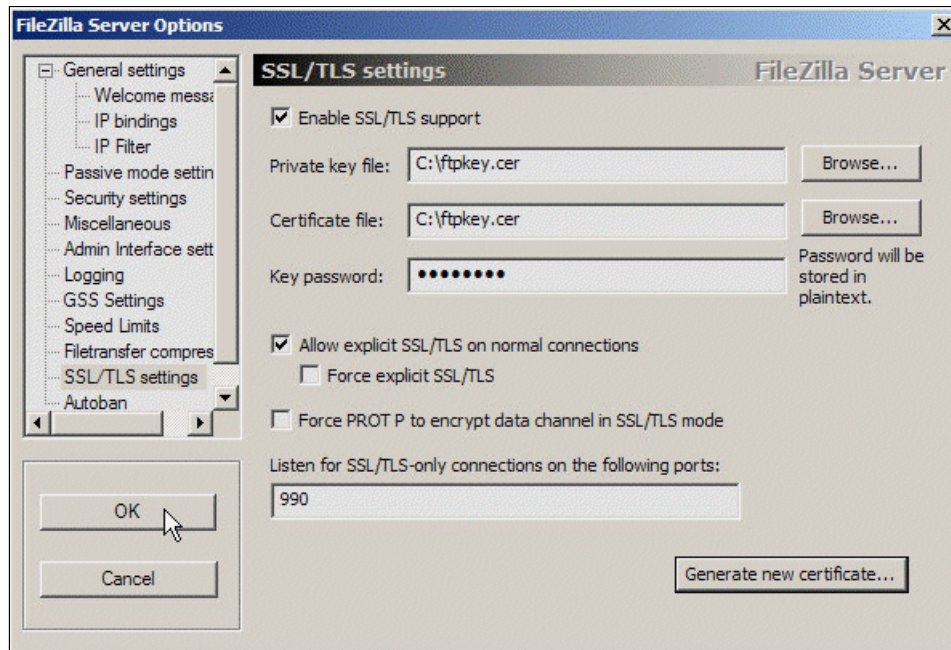


Figure 10-7 Complete FileZilla server options

The server stores the settings for further use. You are returned to the FileZilla Server main window.

The private key and the certificate are now stored in local file `c:\ftpkey.cer`. FileZilla stores the data in base64 text form. When you review the file contents with a text editor, it looks similar to the contents that is shown in Example 10-5 on page 328.

*Example 10-5 Private key and certificate in base64 format*

---

```
-----BEGIN RSA PRIVATE KEY-----
MIICXQIBAAKBgQDmYs6yuU/5Fq5vCHKIvwKMmpJuQEWUpOF7BJoY8Dt1Lfp+fER
4SLLrnFrXL6NBG735namX1Ed7Du3/9LIEIAIE6u0z0bGuie6699zenZwBUqAcaZL
RzgMSyEiTUUy4Pa9mvuKRAGGPP/8Wj0Ekzx5ieiUAGSTXpXtKzNEyWiwIDAQAB
AoGAD/aWteGLPgopSf4/TLDXf2CSdtszNnbeS/BAkkUfMBuGJssvfpoi85pg3sd
...
gcJ5Phq811pcxmrpzAcCQQCyGwtJTW1ceENAYJpGJKSzB7F0EjZL5N05vgwkFu1C
F651IQFfYCo3XRZXJSypF42RUCiMsJjipUQFpL0cKGEo
-----END RSA PRIVATE KEY-----
-----BEGIN CERTIFICATE-----
MIIDADCCAmgAwIBAgIBADANBgkqhkiG9w0BAQUFADCBxTEYMBYGA1UEAxMPeW91
ci1zZXJ2ZXIuY29tMQswCQYDVQQGEwJVUzEfMBOGA1UECBMWW91ciBzdGF0ZSBv
ciBwcm92aW5jZTESMBAGA1UEBxMJWW91ciBjaXR5MR0wGAYDVQQKExFzb3VyIG9y
...
EeEiy65xa8S+jQRu9+Z2p19RHew7t//SyBCAJR0rtM9Gxronuuvfc3p2cAVKgHGm
S0c4DEsmBIk1FMuD2vZr7ikQBhJz//FozhJM8eYno1ABkk016V7SszRM1osCAwEA
ATANBgkqhkiG9w0BAQUFAA0BgQCH1VcZJKVwCTHJCz0W7RHgrPgadMQTxNe6IKE/
Jce0fmA7aq0ruukSnG7NxAe2p3fWuKe+C8Vq2vE0hnG99AH4XIVr33Ri1p0UnyQj
cKVdX0/XCC9ta4N24QZW11GD6Nxp/sgoLsPbWbHKS4/CHNZKcmJjrTJSSAn2aBjv
ds10ig==
-----END CERTIFICATE-----
```

---

Your next step is to read the FTP server certificate from the local file and send it to z/VSE.

## Using OpenSSL to create the server key and certificate

This section describes how to create the server key and certificate by using OpenSSL to specify a different validity period than one year. (You can skip this section if you created the certificate with FileZilla as described in “Generating the server key and certificate” on page 324.)

At the time of this writing, this method is the only method of which we are aware. FileZilla does not accept PFX or JKS files as the key or certificate file. Only base64-encoded (PEM) files can be used and OpenSSL seems to be the only available application that creates the correct format.

Installing OpenSSL on Windows XP requires the following files:

- ▶ Visual C++ 2008 Redistributables. These runtime components are necessary for OpenSSL.
- ▶ Win32 OpenSSL v0.9.8i Light.

The files are available for download from:

<http://www.slproweb.com/products/Win32openssl.html>

First, download and install the Visual C++ Redistributables; then, install OpenSSL Light.

To create the key and certificate files, open a command prompt and go to the OpenSSL installation bin directory. Then, enter following command string:

```
openssl req -new -x509 -keyout ftpkey.pem -out ftpcert.pem -days 3650
```

In this example, the certificate is valid for 10 years (3650 days).



You are prompted to specify personal information, as shown in Example 10-6.

*Example 10-6 OpenSSL prompt for certificate information*

```
C:\OpenSSL\bin>openssl req -new -x509 -keyout ftpkey.pem -out ftpcert.pem -days
3650
Loading 'screen' into random state - done
Generating a 1024 bit RSA private key
.....+++++
.....+++++
writing new private key to 'ftpkey.pem'
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:DE
State or Province Name (full name) [Some-State]:BW
Locality Name (eg, city) []:Boeblingen
Organization Name (eg, company) [Internet Widgits Pty Ltd]:IBM Germany
Organizational Unit Name (eg, section) []:Development
Common Name (eg, YOUR name) []:9.152.222.125
Email Address []:zvse@de.ibm.com

C:\OpenSSL\bin>
```

Now, open the FileZilla SSL/TLS settings panel, as shown in Figure 10-8. Enter the file names for the key and certificate files, and click **OK**.

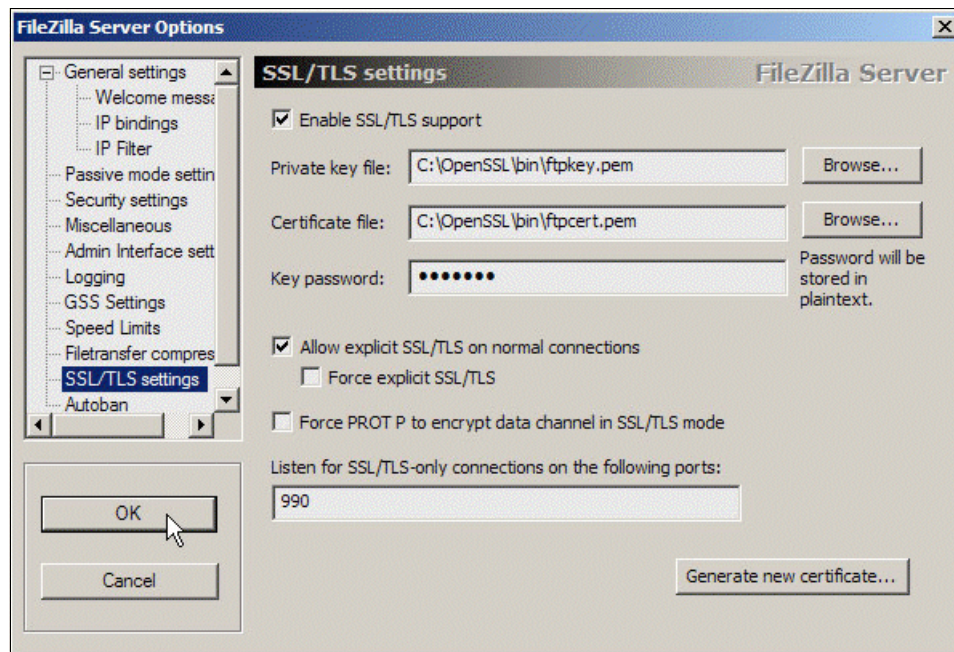


Figure 10-8 FileZilla server options with new SSL settings

For more information about uploading the certificate to z/VSE, see “Sending the server certificate to z/VSE” on page 330.

## **Sending the server certificate to z/VSE**

A certificate can be sent to z/VSE by using one of the following methods:

- ▶ Copy the text form of the certificate into a VSE/POWER job.
- ▶ Use the Keyman/VSE tool to upload the certificate to z/VSE.

### ***Use a VSE/POWER job***

Paste a copy of the text form of the certificate into a job that is similar to Example 10-7 and specify the member name of the z/VSE library member, which contains the certificate on your z/VSE system.

#### *Example 10-7 VSE/POWER job to catalog server certificate*

---

```
$$ JOB JNM=CIALROOT,CLASS=0,DISP=D
$$ LST CLASS=A
// JOB CIALROOT
// OPTION SYSPARM='00'           SysId of main TCP/IP partition
// EXEC CIALROOT,SIZE=CIALROOT,PARM='CRYPTO.KEYRING.MYCERT'
-----BEGIN CERTIFICATE-----
MIIDADCCAmgAwIBAgIBADANBgkqhkiG9w0BAQUFADCBxTEYMBYGA1UEAxMPeW91
ci1zZXJ2ZXIuY29tMQswCQYDVQQGEwJVUzEfMB0GA1UECBMWW91ciBzdGF0ZSBv
ciBwcm92aW5jZTESMBAGA1UEBxMjJW91ciBjaXR5MR0wGAYDVQQKExFzb3VyIG9y
Z2FuaXphdG1vb290MB0GA1UECXMWW91ciBvcmdhbm16YXRpb24gdW5pdDEqMCgG
...
S0c4DEsmBik1FMuD2vZr7ikQBhJz//FozhJM8eYno1ABkk016V7SszRM1osCAwEA
ATANBgkqhkiG9w0BAQUFAA0BgQCH1VcZJKVwCTHJCz0W7RHgrPgadMQTxNe6IKE/
Jce0fmA7aq0ruukSnG7NxAe2p3fWuKe+C8Vq2vE0hnG99AH4XIVr33Ri1p0UnyQj
cKVdX0/XCC9ta4N24QZW1lGD6Nxp/sgoLsPbWbhKS4/CHNZKcmJjrTJSSAn2aBJv
ds10ig==
-----END CERTIFICATE-----
/*
/&
$$ E0J
```

---

This job catalogs a new z/VSE library member MYCERT.ROOT in sublibrary CRYPTO.KEYRING.

### ***Using the Keyman/VSE tool***

Copy the certificate text, including the delimiter lines BEGIN CERTIFICATE and END CERTIFICATE, into the clipboard and read the clipboard contents with Keyman/VSE, as shown in Figure 10-9 on page 331.

As an alternative, you can read the file that is created by FileZilla or OpenSSL directly by selecting the option **Import certificate from file** with a Keyman/VSE version that has a build date of August 2007 or later.

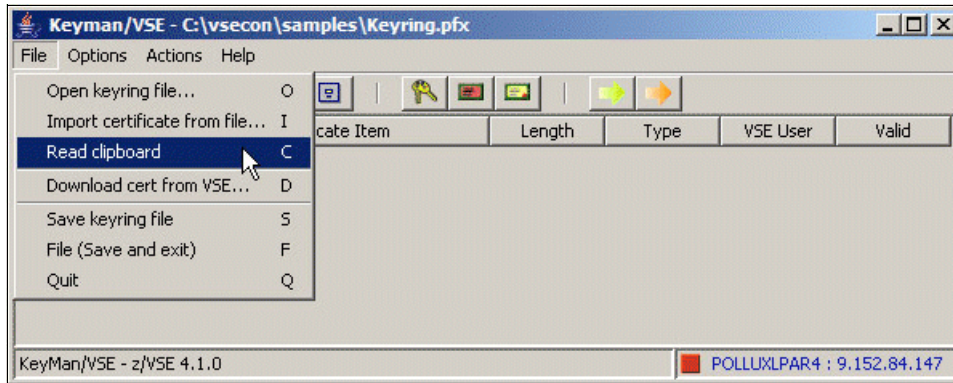


Figure 10-9 Keyman/VSE - read certificate

The certificate is now available for upload in Keyman/VSE. Before performing the upload, as shown in Figure 10-10, ensure that the correct z/VSE system is shown in the lower right corner of the Keyman/VSE main window (here, it is POLLUXLPAR4) and that the z/VSE Connector Server is running on z/VSE.

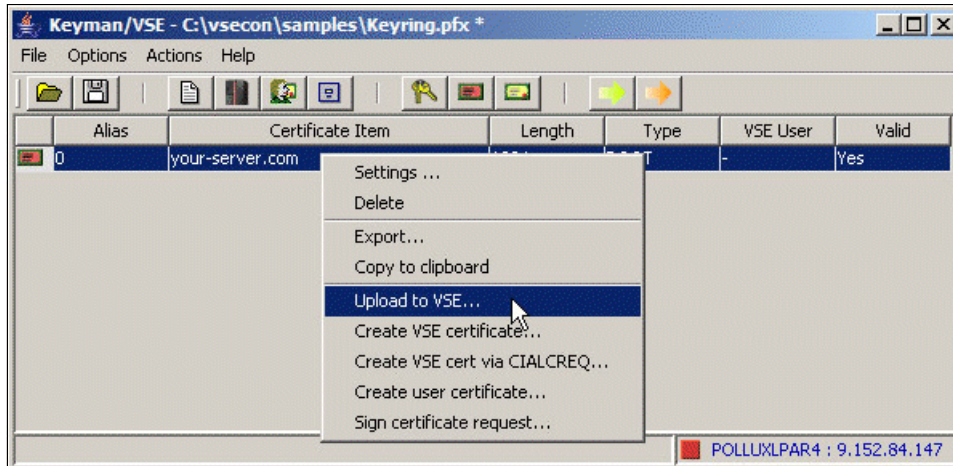


Figure 10-10 Upload certificate to z/VSE

If you encounter problems uploading the certificate, see 10.5.1, "Cannot submit a VSE/POWER job with Keyman/VSE" on page 343.

## Defining an FTP user in FileZilla

To set up an FTP user in the FileZilla server, complete the following steps:

1. Select **Edit** → **Users**, as shown in Figure 10-11.

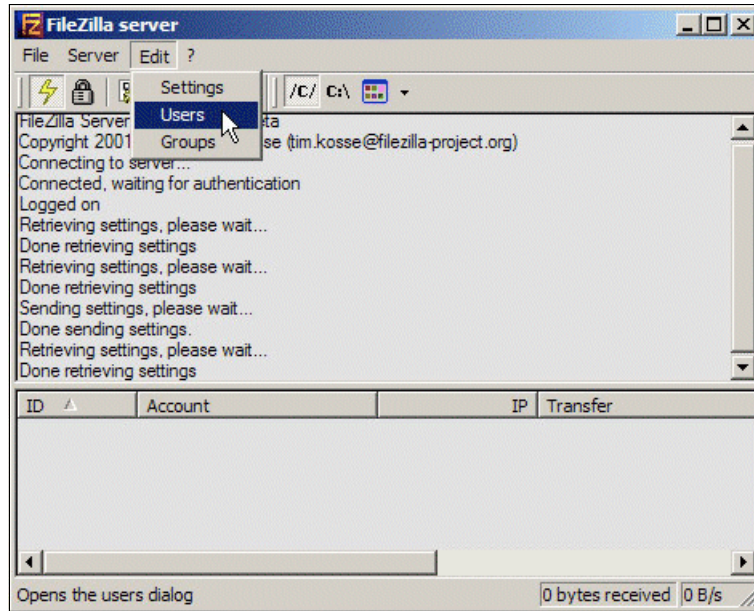


Figure 10-11 FileZilla server main window

2. In the Users window, click **Add**, as shown in Figure 10-12.

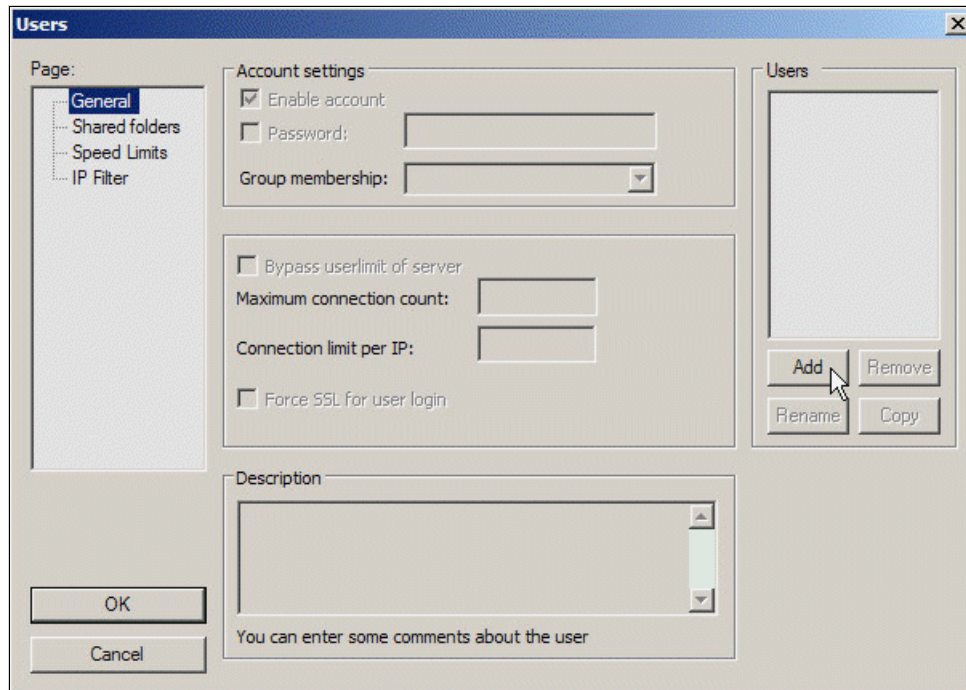


Figure 10-12 FileZilla Users window



3. Enter the name of your FTP user ID, as shown in Figure 10-13.



Figure 10-13 Add user

4. Specify a password for this user, as shown in Figure 10-14.

**Notes:** Consider the following points:

- ▶ Passwords are case-sensitive. When you specify the password in uppercase characters on the z/VSE side, you must also use uppercase letters here.
- ▶ Because user IDs are not case-sensitive, so you can use mixed-case characters when the user in FileZilla is defined. However, use uppercase characters on the z/VSE side.

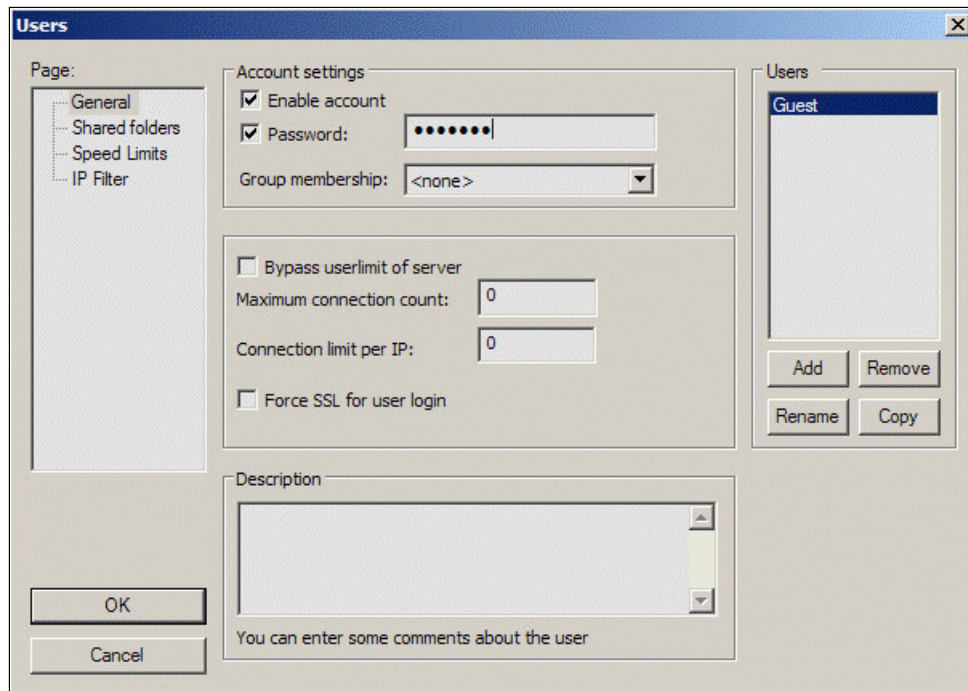


Figure 10-14 Specify password

5. You also must specify a home directory for this user. Otherwise, connections are not possible. In the next window (see Figure 10-15 on page 334), select **Shared folders** and click **Add**.

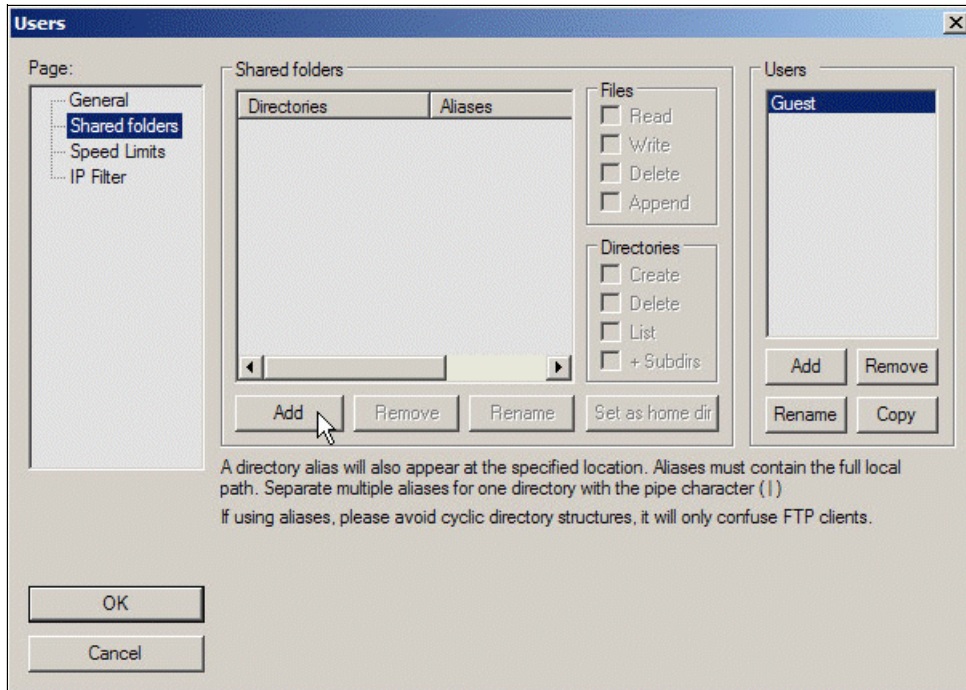


Figure 10-15 Add shared folders for this user

- Click **Set as home dir**, as shown in Figure 10-16. You also might want to customize the security settings in the Files and Directories group boxes. Click **OK**. We are now ready to connect to the server from a z/VSE system.

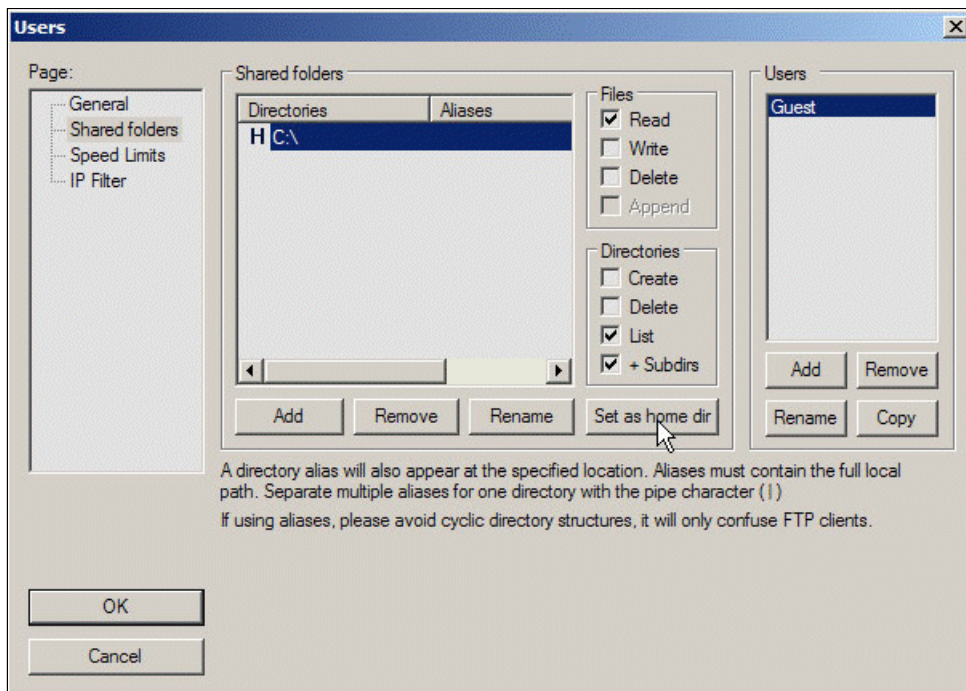


Figure 10-16 Set user's home directory

## Connecting to the server by using the z/VSE FTP client

You can now connect to the FTP server by using the z/VSE FTP client. The IP address that is shown in Example 10-8 belongs to the Windows workstation where the FileZilla server runs.

### Example 10-8 z/VSE FTP client job

---

```
* $$ JOB JNM=FTPBATC,CLASS=4,DISP=D
// JOB FTPBATC
// OPTION SYSPARM='00'
// LIBDEF PHASE,SEARCH=PRD1.BASE
// EXEC FTPBATC,SIZE=FTPBATC,PARM='SSL=CLIENT'
SET SSL PRIVATE CRYPTO.KEYRING.MYCERT NOCLAUTH ALL
LOPEN
LUSER JSCH
LPASS MYPASSW
LAUTH SSL
OPEN 9.152.216.58 990
AUTH SSL
PROT P
USER GUEST
PASS GUESTPW
DIR
CLOSE
LCLOSE
QUIT
/*
/&
* $$ E0J
```

---

Ensure that the certificate name matches the name that you specified when uploading the certificate to z/VSE. In this example, we used the member name MYCERT.

In the job, LUSER and LPASS specify a z/VSE user with its password. These parameters are necessary to enable the FTP client to access the z/VSE file system. The USER and PASS specify the remote FTP user and its password, as defined on the server side in “Defining an FTP user in FileZilla” on page 332. The remote password (PASS) is case-sensitive.

**Note:** The interactive FTP client (CICS FTP transaction) is not SSL-enabled.

## 10.3.2 Sample setup with vsftpd server on Linux

The following sections describe the SSL setup with vsftpd (very secure FTP server) on Linux.

Our test environment consisted of the following components:

- ▶ Linux RHEL 5.6 on Intel by way of PuTTY
- ▶ vsftpd 2.3.2
- ▶ z/VSE V4R3M0
- ▶ TCP/IP for VSE/ESA 1.5F

Because the FTPBATC client on z/VSE does not support SSL client authentication, we only tested SSL server authentication.

## Creating the key and certificate

We used OpenSSL to create the RSA key and the certificate for vsftpd. The following commands create a .pem file that contains a new RSA key and a self-signed certificate:

```
# cd /etc/vsftpd/  
# /usr/bin/openssl req -x509 -nodes -days 365 -newkey rsa:1024 -keyout vsftpd.pem  
-out vsftpd.pem
```

The -nodes parameter means “no DES encryption” and causes the .pem file to be *not* password-protected. We did not try password protection.

Now, follow the prompts on the shell that is shown in Example 10-9.

### *Example 10-9 Create RSA key and certificate on the shell*

---

```
Generating a 1024 bit RSA private key  
.....++++++  
.....++++++  
writing new private key to 'vsftpd.pem'  
-----  
You are about to be asked to enter information that will be incorporated  
into your certificate request.  
What you are about to enter is what is called a Distinguished Name or a DN.  
There are quite a few fields but you can leave some blank  
For some fields there will be a default value,  
If you enter '.', the field will be left blank.  
-----  
Country Name (2 letter code) [AU]:DE  
State or Province Name (full name) [Some-State]:  
Locality Name (eg, city) []:Boeblingen  
Organization Name (eg, company) [Internet Widgits Pty Ltd]:IBM Germany  
Organizational Unit Name (eg, section) []:Development  
Common Name (eg, YOUR name) []:LINR02  
Email Address []:zvse@de.ibm.com  
linr02:/etc/vsftpd #
```

---

## Transferring the certificate to z/VSE

To transfer the certificate to z/VSE, we displayed the .pem file in the shell, as shown in Example 10-10. Then, we copied the certificate portion to the clipboard.

### *Example 10-10 Display private key and certificate*

---

```
linr02:/etc/vsftpd # cat vsftpd.pem  
-----BEGIN RSA PRIVATE KEY-----  
MIICXQIBAAKBgQDDpKdOWHpP1RdoygV40vdsPmrzm/q72aBwi0qvPEiCdmce891i  
XBds4lhPOWChgVN+ZvnEGjmMd3b16FrdFRaE8DpHvwcY13kpfNe1HkchvU+2Fa/y  
srZBR90Mn4hXmtPdbcIk2yydm/LSYqe2YrTp1rP1Tj74Z6G4jq9Q8ErfXwIDAQAB  
AoGABfq7T3gAzPveikpaXnFW7Iz/0dA7mpAtMiNftf1esfYScf10E9qCsMY+MrpQ  
...  
gdr007z1kF8gxpeKnioNWj03Dowpsh9IM60/9TpYeyj6R5Va2fr+osu2HPECQQCv  
I/jpmvbn4fuDZxU99F2L/SNG+150UZ4xuYCGajNH2xaQdMGgv2EMFypzecPhOd+D  
lpviGFp+vTcRvGWQcMDTAKB53pEIT/f323CSbbP003kfU4QBsm7IrUt7qicxp6J  
IHQEE0R3/HENTpPyKSRZL1F56a5t8F6gh11ddsazXXed  
-----END RSA PRIVATE KEY-----  
-----BEGIN CERTIFICATE-----  
MIIDpTCCA6wAwIBAgIJAKXdxR/V2LxOMA0GCSqGSIb3DQEBBQUAMIGUMQswCQYD
```



```
VQQGEwJERTETMBEGA1UECBMKU29tZS1TdGF0ZTETMBEGA1UEBxMKQm91Ymtpbmd1
bjEUMBIGA1UEChLSUJNIEEd1cm1hbnkxZDASBgNVBAsTCOR1dmVsb3BtZW50MQ8w
DQYDVQQDEwZMSU5SMDIxHjAcBgkqhkiG9w0BCQEWd3p2c2VAZGUuaWJtLmNvbTAe
...
pd3FH9XYvE4wDAYDVROTBAAUwAwEB/zANBgkqhkiG9w0BAQUFAAOBgQAPYB7P+L66
eZHeUROiEHjWLENWC1XMnWmD0kQom5jFHG8LDFX16oB7GQwPecxiBnHCyIZNgp2b
F3RcOKSUCuYCGDoq/mSjhwv9pbeMHpG1gAXMUgKnFUV04BKPYYfaTUaibfVMMF
HzCrb3mJT1WDiaD1aL+L3hPRU/gS+vT6rw==
-----END CERTIFICATE-----
linr02:/etc/vsftpd #
```

---

Cataloging the certificate in z/VSE can be done by using one of the following methods:

- ▶ Using a CIALCERT job
- ▶ Using the Keyman/VSE tool

### ***Cataloging the certificate in z/VSE with a CIALCERT job***

Manually create a CIALCERT job and paste the certificate into z/VSE, as shown in Example 10-11.

#### *Example 10-11 CIALCERT job*

---

```
* $$ JOB JNM=CIALCERT,CLASS=0
// JOB CIALCERT
// EXEC CIALCERT,SIZE=CIALCERT,PARM='CRYPTO.KEYRING.VSFTPD'
-----BEGIN CERTIFICATE-----
MIICHTCCAe4CCQCYw0JIiyOHNTANBgkqhkiG9w0BAQUFADCBhjELMAkGA1UEBhMC
QQgxZAJBgNVBAGTA1pIMQ8wDQYDVQQHEwZadXJpY2gxDTALBgNVBAoTBEdFTUix
CzAJBgNVBAsTAK1UMRgwFgYDVQQDEw9teWNoZ3N0LmxvY2F5ZSMB4XDTEwMTEyMjE1
9w0BCQEWFHJvb3RABG9jYWxob3N0LmxvY2F5ZSMB4XDTEwMTEyMjE1QjE1S5W
y3vmFaMMRuXvSua6Trta2oC76gAWJYImgQLB8C+Bu2aQekG22CukFWAowW7dm9zx
wZgN4JuQZ+KQDeEuw+QH1M1RiLnkcqmK31bfjKSXnSL3AXiWp1hmku0MoSMjZksP
/zP78foU1Qzn16/WFnbS1Zojv/cKkz1Qjg==
-----END CERTIFICATE-----
/*
/&
* $$ E0J
```

---

Then, run this z/VSE job.

### ***Cataloging the certificate in z/VSE using Keyman/VSE***

Start the Keyman/VSE tool and select **File** → **Read clipboard**.

As shown in Figure 10-17 on page 338, the certificate is displayed as a root certificate because it is self-signed. That is, the issuer and subject name are equal. When you upload it to z/VSE, you must catalog it with member type .cert. You can select the member type in the upload window.

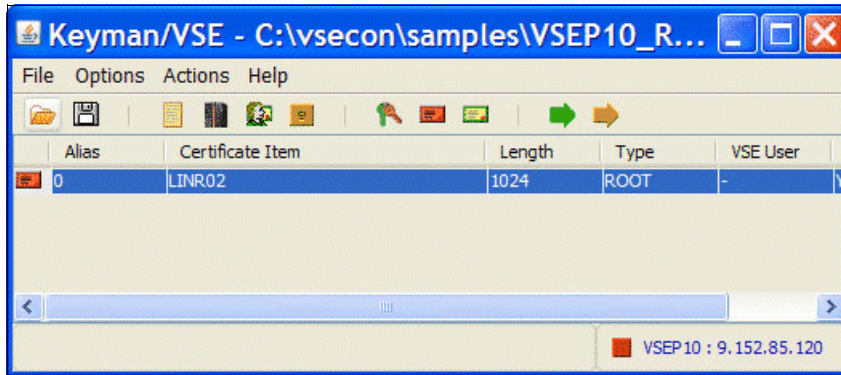


Figure 10-17 Certificate that is listed in Keyman/VSE

Right-click the certificate and select **Upload to VSE** to open the upload window.

Select **CERT**, as shown in Figure 10-18.

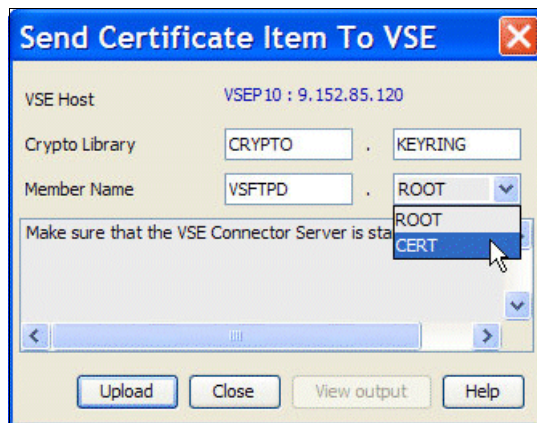


Figure 10-18 Selecting CERT

Then, click **Upload**.

### vsftpd configuration

The vsftpd.conf file is in the /etc directory. The following parameters are relevant for SSL:

```

ssl_enable=YES
force_local_data_ssl=NO
force_local_logins_ssl=YES
ssl_tlsv1=YES
ssl_sslv2=NO
ssl_sslv3=YES
rsa_cert_file=/etc/vsftpd/vsftpd.pem
listen_port=990
implicit_ssl=YES

```

The following parameter is important for z/VSE:

```

ssl_request_cert=NO

```

**Note:** On the vsftpd man page, you see get the following information:

ssl\_request\_cert

If enabled, vsftpd requests (but not necessarily require; see require\_cert) a certificate on incoming SSL connections. Normally, this issue does not cause problems, but IBM z/OS seems to have issues (new in v2.0.7).

The default is YES.

With z/VSE, this parameter *must* be set to NO. Apparently this setting affects the TLS protocol version that is used. Although z/VSE supports SSL up to TLS 1.0, vsftpd uses TLS 1.2 by default.

## Starting the vsftpd

The vsftpd starts as a service:

```
linr02:/etc/vsftpd # service vsftpd start
Starting vsftpd
linr02:/etc/vsftpd #
```

For more information about how to operate the server, maintain users, and so on, see the various vsftpd documentation that is available on the internet.

## z/VSE JCL

On z/VSE, use some JCL that is similar to the JCL that is shown in Example 10-12 to connect to vsftpd.

*Example 10-12 z/VSE job to connect to vsftpd*

---

```
* $$ JOB JNM=FTPSSL,DISP=D,PRI=3,CLASS=S
// JOB FTPSSL
// OPTION LOG,NOSYSDMP
// OPTION SYSPARM='00'
// LIBDEF *,SEARCH=(PRD2.TCP15F,PRD1.BASE)
// EXEC FTPBATCH,SIZE=FTPBATCH,PARM='SSL=CLIENT'
SET SSL PRIVATE CRYPTO.KEYRING.VSFTPD NOCLAUTH ALL
SET DIAGNOSE EVENT
SET DIAGNOSE ON
lopen
luser JSCH
lpass mypasswd
OPEN 9.152.84.171 990
user anonymous
pass blah
passive
dir
close
quit
/*
/&
* $$ E0J
```

---

## 10.4 Considerations for firewalls

Typically for company intranets, network access is controlled by firewalls. This control implies that often all port numbers are blocked, except for a few ports that are open for use by selected intranet applications.

When the Keyman/VSE tool is used to upload a generated private key to z/VSE in a firewall-controlled network, port 6045 must be open. This port must be open because it is the default port that is used by the CIALSRVR program, which receives the key material on z/VSE.

For FTP, port 21 often is used for unsecured FTP connections, and port 990 is for secure FTP connections. In addition, the FTP protocol requires one or more data connections, which are opened dynamically during an FTP session when transferring data such as files or even directory lists. Which port numbers are selected in a particular case depends on the FTP server and client implementations. To overcome this situation, most FTP servers provide the possibility to use either active or passive FTP mode.

### 10.4.1 Passive versus active FTP mode

Active FTP mode means that the FTP client tells the FTP server which data port to use for the transfer of a specific file. Now, when the server connects to this port, the client's firewall sees this connection as an incoming connection from a remote system.

Passive mode is started by the FTP client and tells the server to specify the data port. This process implies that the server must be configured to have certain port numbers open for use to connect to passive FTP clients.

For more information about active versus passive FTP mode, see this web page:

<http://slacksite.com/other/ftp.html>

## 10.4.2 Restricting the port range on the server side

The FileZilla server allows restricting the range of used data ports. In the FileZilla Server Options window, select **Passive Mode Settings** and specify the range of open ports in your company intranet, as shown in Figure 10-19.

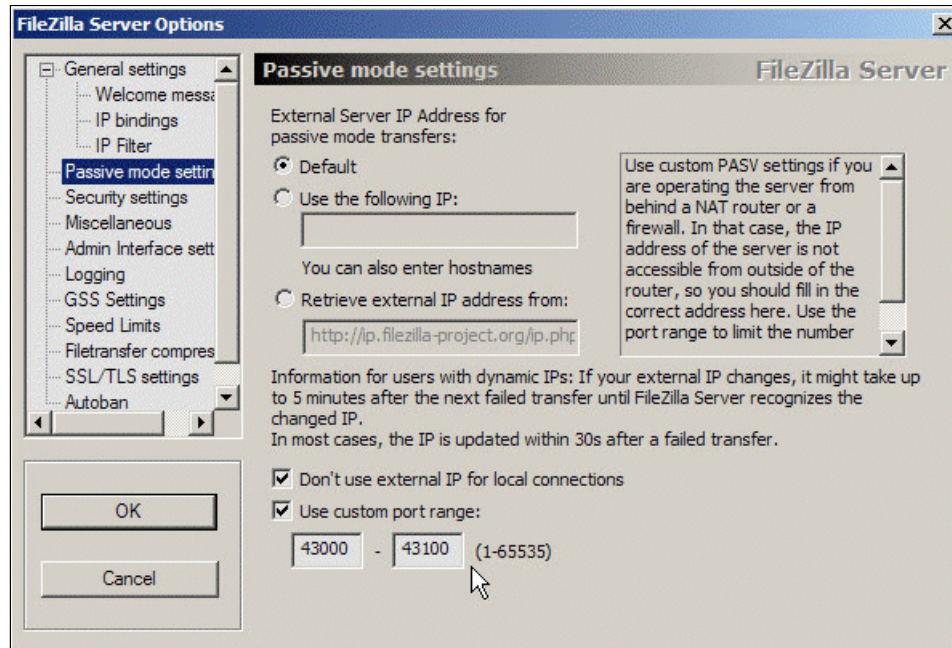


Figure 10-19 FileZilla server options

TCP/IP for VSE/ESA 1.5E provides the PORTRANGE command to manage firewall issues. The command is described in *TCP/IP for VSE V1R5.0 Command Reference*, SC33-6764. The command can be entered at the operator console or be specified in your IPINIT member and applies to all FTP servers and clients that are running on this z/VSE system.

The command features the following syntax:

```
PORTRange ,HIgh=num ,LOW=num
```

The following example uses the command:

```
F7-0104 IPN300I Enter TCP/IP Command
104 PORTRANGE, LOW=4096, HIGH=65535
F7 0098 IPN127E Port range changed to 4096 65535
```

The values 4096 and 65535 are the defaults if nothing else is specified. Any range with a difference of at least 4096 can be used and applies to all free port requests.

**Note:** Because PORTRANGE is a TCP/IP command and not an FTPBATCH or FTPD parameter, the values that are defined here apply to the entire stack, which means to all FTP servers and clients.

### 10.4.3 Restricting the port range on the client side

The FileZilla client allows restricting the port range in a similar way as the server (see Figure 10-20).

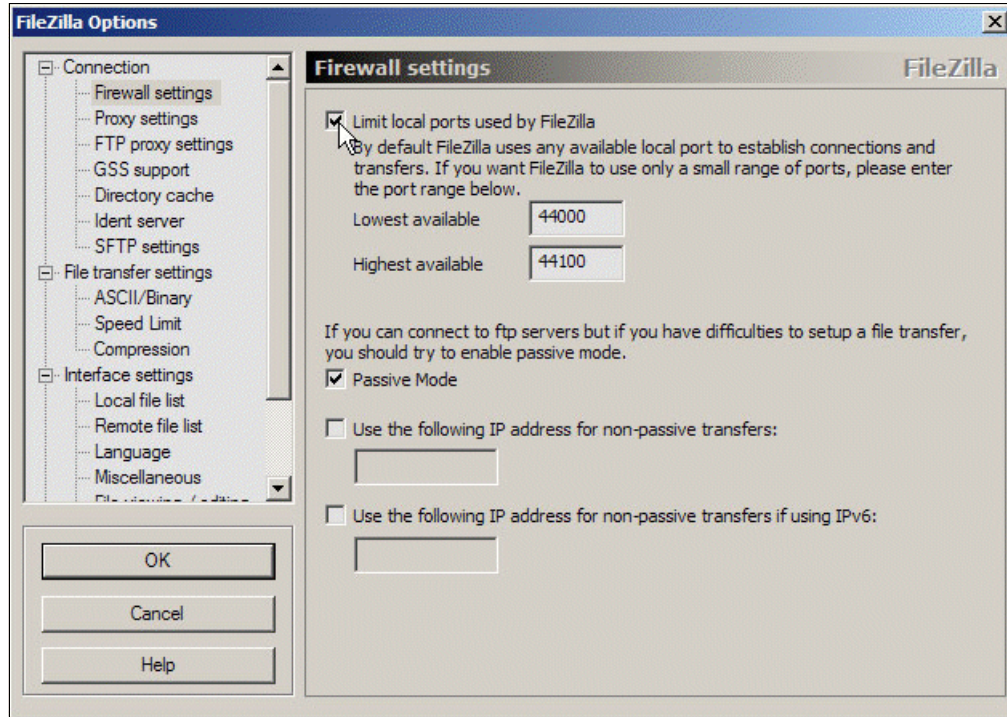


Figure 10-20 FileZilla options - port range

If z/VSE acts as the client, PORTRANGE applies in the same way as for the server.

### 10.4.4 Considerations on the DATAPORT parameter

The z/VSE FTPBATCH application includes the following optional parameter to specify a data port:

```
EXEC FTPBATCH,PARM='xxx,DATAPORT=43000'
```

We found that this data port is used for transferring files only, but it is not used when issuing a DIR command to the remote platform. The DIR list is transferred through another randomly selected port. With each subsequent DIR command, the port number is increased, which means that the firewall administrator is forced to open more ports to get it to work.

**Note:** Connectivity Systems, Inc. recommends not using this parameter unless conditions are exceptional. The parameter forces the FTPBATCH job to use one specific data port, which can cause other problems. For example, when many incoming connections must be handled by one DATAPORT, the port opens and closes in short time intervals, which can block further connections.

For more information about z/VSE FTPBATCH parameters, see the following publications:

- ▶ *TCP/IP for VSE V1R5.0 User Guide, SC33-6763*
- ▶ *TCP/IP for VSE V1R5.0 Optional Features, SC33-6767*



## 10.4.5 Firewall configuration

One important firewall setting is the permission or blocking of the different IP protocols: TCP, UDP, and ICMP. At least the TCP protocol must be permitted so that FTP can work.

Figure 10-21 shows how the Symantec Client Firewall handles these definitions. Depending on the firewall product that is used, the user interface might be different.

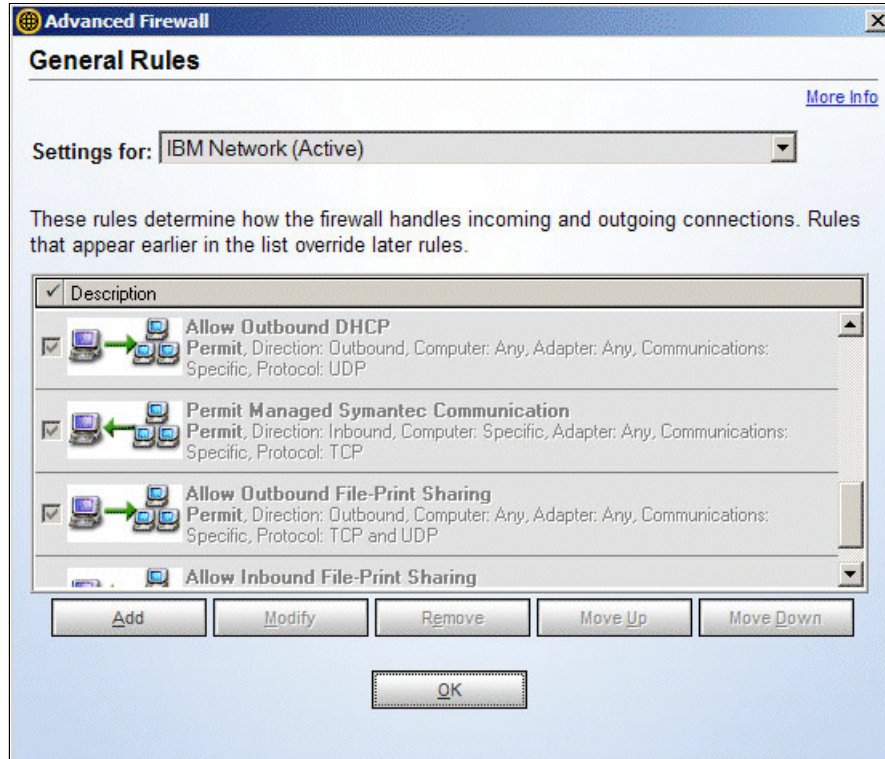


Figure 10-21 Firewall configuration

## 10.5 Observations

This section describes several observations that we made during our tests.

### 10.5.1 Cannot submit a VSE/POWER job with Keyman/VSE

The symptom and possible reason are described.

#### Symptom

When uploading the server certificate to z/VSE through Keyman/VSE, the CIALROOT job never appears in the z/VSE reader queue, although displaying the contents of the z/VSE keyring library is possible. The connection indicator at the lower right corner of the Keyman/VSE main window is green, which shows that the IP connection to the z/VSE Connector Server is established.

### Possible reason

You are using an external security manager (ESM), such as CA-Top Secret from Computer Associates, or Alert from Connectivity Systems, Inc. (CSI), and the ESM is not correctly configured so that your z/VSE user can submit VSE/POWER jobs through the z/VSE Connector Server.

When CA-Top Secret is used, you can use following command to give your z/VSE user full authorization, or contact CA for the related information:

```
TSS ADD(user-ID) IESINIT(IESEADM) IESTYPE(USERTYPE1,NEW,SELECT)
IESFL1(BAT,PSL,COD,VSAM) IESFL2(BQA,ESC,COU,CMD,OLPD,XRM)
```

For Alert, contact CSI to obtain the related information or disable security in the z/VSE Connector Server by completing the following tasks:

- ▶ Modify job skeleton SKVCSCFG in ICCF library 59, as shown in Example 10-13.
- ▶ Catalog the configuration member by using job skeleton SKVCSCAT and restart the connector server.

#### *Example 10-13 Stop security*

---

```
; *****
; SECURITY CONFIGURATION
; - SECURITY: FULL      - LOGON, RESOURCE AND USER TYPE CHECKING
;       RESOURCE - LOGON AND RESOURCE, BUT NO USER TYPE
;                   CHECKING.
;       LOGON      - LOGON, BUT NO RESOURCE AND USER TYPE
;                   CHECKING
;       NO         - NO LOGON, RESOURCE AND USER TYPE CHECKING
; *****
; SECURITY = NO
```

---

## 10.5.2 SSL handshaking fails

The symptom and possible reason are described.

### Symptom

SSL handshaking fails when connecting from FTPBATCH on z/VSE to a FileZilla server. FileZilla shows the following error messages in the FileZilla server interface window:

```
150 Opening data channel for file transfer.
Data connection SSL warning: SSL3 alert write: fatal: handshake failure
Data connection SSL warning: SSL_accept: error in SSLv3 read client hello C
```

### Possible reason

You are using TCP/IP for VSE/ESA 1.5F and some fixes are missing. Check for the following 15F zaps:

- ▶ 244
- ▶ 247
- ▶ 251
- ▶ 252
- ▶ 253





## WebSphere MQ with SSL

In this chapter, the term *MQ* is used to refer to two products: MQSeries for VSE/ESA and WebSphere MQ for z/VSE. MQ provides application programming services that enable application programs to communicate with each other by using message queues.

In the context of online applications, messaging and queuing can be understood as follows:

- ▶ Messaging means that programs communicate by sending each other messages (data) rather than by calling each other directly.
- ▶ Queuing means that the messages are placed in queues in storage so that programs can run as follows:
  - Independently of each other
  - At different speeds and times
  - In different locations
  - Without having a logical connection between them.

In this chapter, we first describe the basic setup of an MQ environment on z/VSE and Windows. The basic setup is then enhanced to support SSL.

This chapter includes the following topics:

- ▶ 11.1, “Introduction” on page 346
- ▶ 11.2, “Installing WebSphere MQ” on page 346
- ▶ 11.3, “Configuring WebSphere MQ” on page 356
- ▶ 11.4, “Configuring for SSL” on page 378
- ▶ 11.5, “Implementing SSL client authentication” on page 393
- ▶ 11.6, “Using SSL peer attributes” on page 393
- ▶ 11.7, “Configuring a z/VSE queue manager remotely” on page 396
- ▶ 11.8, “Observations” on page 409

## 11.1 Introduction

For our examples, we set up two z/VSE test systems: one with MQSeries for VSE V2R1M2 and the other with WebSphere MQ for z/VSE V3R0. The setup panels that are shown in the following sections were taken from WebSphere MQ for z/VSE V3R0 only; the setup is no different compared to MQSeries for VSE V2R1M2.

In addition, we use the following software in our examples:

- ▶ z/VSE V4R2
- ▶ TCP/IP for VSE/ESA 1.5F
- ▶ VSE Connector Server as part of z/VSE V4R2
- ▶ Java 1.6.0 from Sun Microsystems
- ▶ WebSphere MQ for Windows V7.0 including MQ Explorer V7.0

## 11.2 Installing WebSphere MQ

The MQ security issues RACROUTE requests for MQ-specific resource classes. With z/VSE V4R3 and WebSphere MQ V3R0, these resource classes are supported by the Basic Security Manager (BSM). If you want to activate this MQ security for older versions of MQ, you must have an ESM, such as CA-Top Secret.

In this chapter, we focus on only a secure connection between MQ on z/VSE and MQ on Windows. Therefore, the BSM support is sufficient.

### 11.2.1 MQ installation on z/VSE

A detailed description of the installation of MQSeries on z/VSE is beyond the scope of this document. We used *Using MQSeries for VSE*, SG24-5647 and followed the steps as described in Chapter 1 "Installation." There, we used the following resources:

- ▶ Sublibrary member PRD2.MQSERIES, which contains all members that were restored from tape
- ▶ User catalog MQ.USER.CATALOG with name MQMCAT on volume SYSWK2

After creating all necessary resources and completing all of the definitions, transaction MQSU must be started to initialize the MQSeries configuration file. But, before any MQ transaction is started, you must define a generic security profile to the BSM. For more information about this process, see 11.2.2, "Maintaining security profiles" on page 346.

### 11.2.2 Maintaining security profiles

This section describes how all MQ transactions, starting with the two letters MQ, are defined to the BSM through a generic security profile. Other transactions, such as the TST2 transaction that is used to send some test messages, must be defined in the same way.

Open the dialog that is named Maintain Transaction Profiles by using fast path 2811 and specify option 1 to add a security profile, as shown in Figure 11-1 on page 347.

```

IESADMBLSLE                MAINTAIN SECURITY PROFILES
BSM RESOURCE CLASS:      TCICSTRN                ACTIVE
START...                  (CASE SENSITIVE)
OPTIONS:    1 = ADD        2 = CHANGE        5 = DELETE        6 = ACCESS LIST

      OPT      PROFILE NAME                DESCRIPTION                UNIVERSAL AUDIT
                                      ACCESS VALUE

      1      ftp                IBM SUPPLIED                22
      -      iccf                IBM SUPPLIED                12
      -      lpr                IBM SUPPLIED                12

PF1=HELP                    3=END
PF7=BACKWARD    8=FORWARD    9=PRINT

```

Figure 11-1 Add new profile

Define a generic profile MQ, as shown in Figure 11-2.

```

IESADMBSAE                MAINTAIN SECURITY PROFILES
BSM RESOURCE CLASS:      TCICSTRN

Add Profile:

PREFIX.....                CICS region

RESOURCE NAME..... MQ                Maximum length is 4 characters.
                                      (1=yes, 2=no)

GENERIC..... 1

UNIVERSAL ACCESS... _                (_=None, 2=Read, 3=Update, 4=Alter)

AUDIT-LEVEL 1 ..... 1                (_=None, 1=Failure, 2=Success, 3=All)
ACCESS-LEVEL 1 ..... 2                (2=Read, 3=Update, 4=Alter, _=default)

AUDIT-LEVEL 2 .....                (_=None, 1=Failure, 2=Success, 3=All)
ACCESS-LEVEL 2 .....                (2=Read, 3=Update, 4=Alter, _=default)
DESCRIPTION..... IBM SUPPLIED        Optional remark
PF1=HELP                    3=END                    5=UPDATE

```

Figure 11-2 Define generic profile for MQ

Create an access list for the generic profile. Specify option 6, as shown in Figure 11-3. Press Enter.

```

IESADMBLSLE                MAINTAIN SECURITY PROFILES
BSM RESOURCE CLASS:      TCICSTRN                ACTIVE
START... M                  (CASE SENSITIVE)
OPTIONS:    1 = ADD        2 = CHANGE        5 = DELETE        6 = ACCESS LIST

      OPT      PROFILE NAME                DESCRIPTION                UNIVERSAL AUDIT
                                      ACCESS VALUE

      6      *MQ                IBM SUPPLIED                12
      -      NEWC                IBM SUPPLIED                12
      -      OLPD                IBM SUPPLIED                12

PF1=HELP                    3=END
PF7=BACKWARD    8=FORWARD    9=PRINT

```

Figure 11-3 Create an access for profile MQ

Enter option 1 for ADD and press Enter, as shown in Figure 11-4 on page 348.

```

IESADMBSLA                MAINTAIN ACCESS LIST
BSM    CLASS: TCICSTRN    PROFILE: MQ
START...
OPTIONS:  1 = ADD          2 = CHANGE          5 = DELETE
          NUMBER OF ENTRIES ON LIST:  00000

      OPT    NAME    ACC

      1

PF1=HELP                    3=END
PF7=BACKWARD                8=FORWARD

```

Figure 11-4 Define a new entry in the access list

Now add GROUP01 to the access list and give it read access, as shown in Figure 11-5.

```

IESADMBSAA                MAINTAIN ACCESS LIST
BSM    CLASS: TCICSTRN    PROFILE: MQ

Add Userid or Groupid:

NAME..... GROUP01        Userid or Groupid
ACCESS..... 2             (_=None,
                          2=Read, 3=Update, 4=Alter)

PF1=HELP                    3=END                    5=UPDATE

```

Figure 11-5 Add GROUP01 to access list

Then, press PF5 for UPDATE. To activate your changes, you must rebuild the security tables through dialog 283.

Now the MQSU transaction can be started. Output similar to the following example is displayed:

```
MQSU: MQSeries install completed, 6457 input records read.
```

### 11.2.3 MQ installation on Windows

We used a 90-day trial version of WebSphere MQ V7.0, which is available at this web page:

[http://www.ibm.com/developerworks/downloads/ws/wmq/?S\\_TACT=105AGX28&S\\_CMP=TRIALS](http://www.ibm.com/developerworks/downloads/ws/wmq/?S_TACT=105AGX28&S_CMP=TRIALS)

The installation file, WMQv700Trial-x86\_nt.zip, is approximately 569 MB.

Starting setup.exe opens an installation window in which the following prerequisites are checked:

- ▶ Windows XP + SP2
- ▶ WebSphere Eclipse Platform V3.3

On the Network Configuration tab, we selected **NO**.

**Note:** If WebSphere Eclipse Platform is not installed on your PC, you can install it from the downloaded WebSphere MQ package. “WebSphere Eclipse” should not be confused with the standard “Eclipse” IDE, as downloaded from this website:  
<http://www.eclipse.org>

Enter directory Prereqs/IES and start the setup (setup.exe).

After you install WebSphere Eclipse, you can now install WebSphere MQ 7.0. After copying files, WebSphere MQ performs network configuration tasks.

We again answered **NO** to the question about the existence of any domain controllers running Windows 2000.

Figure 11-6 shows the default configuration window. Thus far, no default configuration is set up. Therefore, click **Set up Default Configuration**.

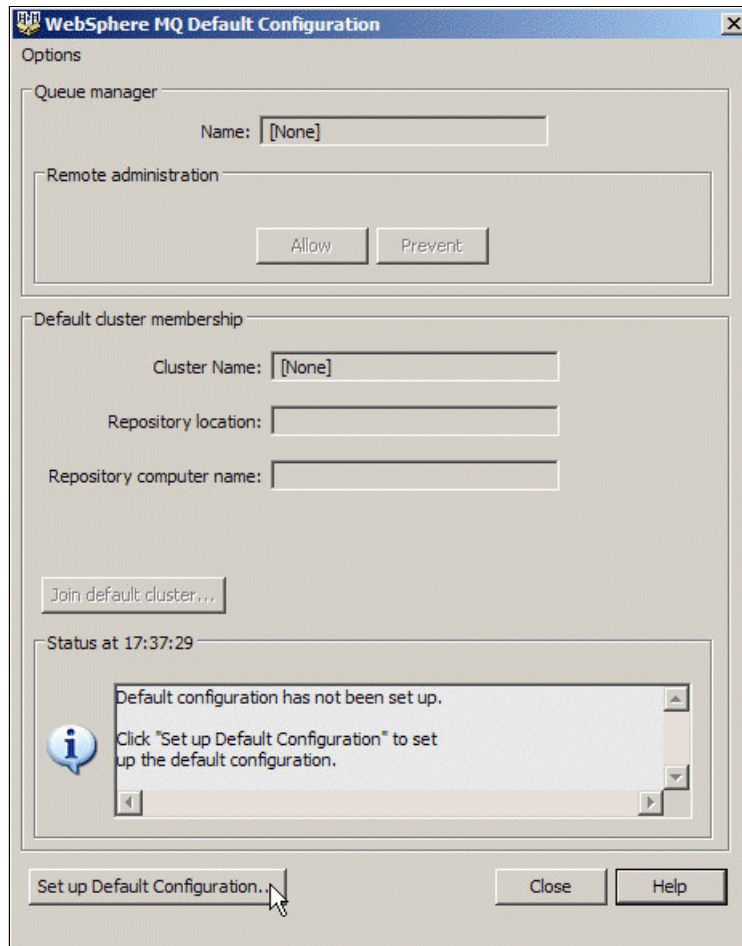


Figure 11-6 WebSphere MQ default configuration

Figure 11-7 on page 350 shows the configuration hints. Click **Next** to start default configuration.

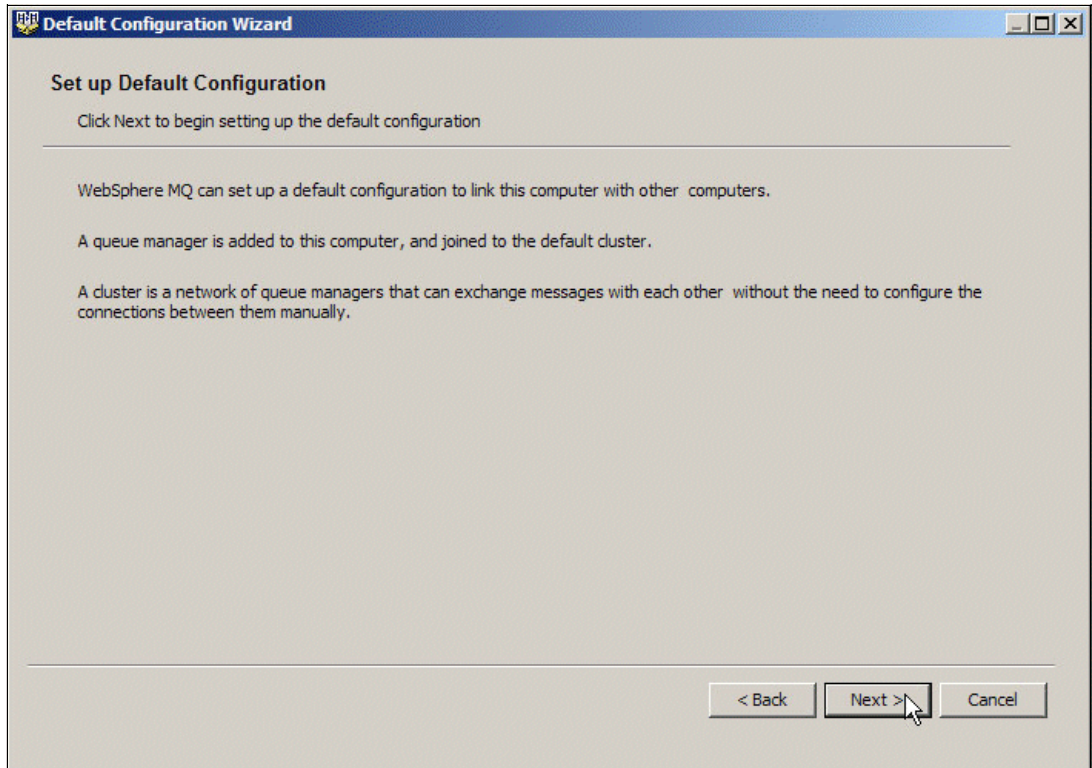


Figure 11-7 Configuration hints

As shown in Figure 11-8, specify a queue manager name and default cluster name. Click **Next**.

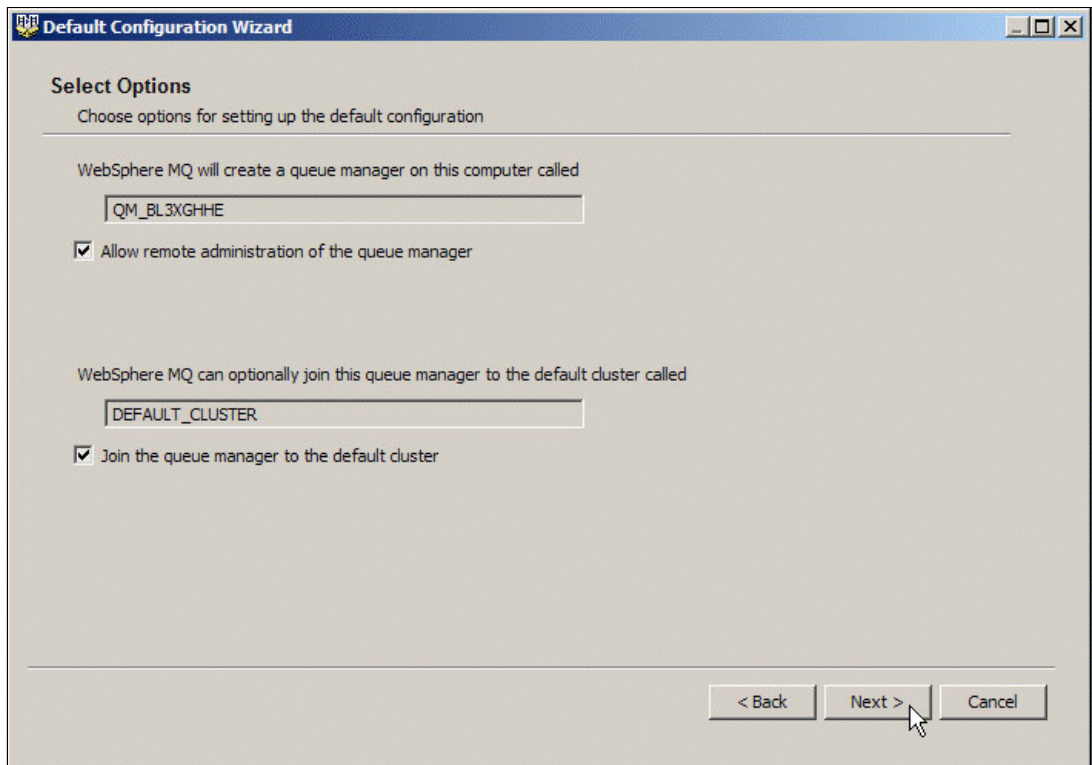


Figure 11-8 Select options for default configuration



As shown in Figure 11-9, confirm that this computer will hold the cluster repository. Click **Next**.

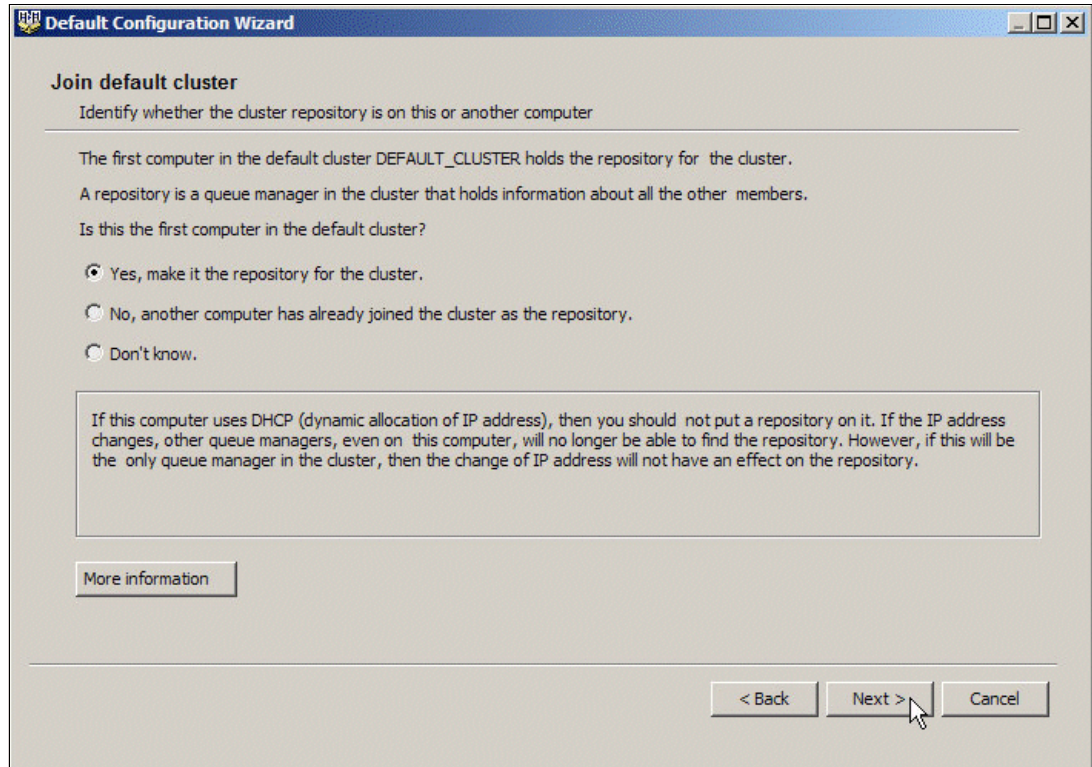


Figure 11-9 Specify computer for cluster repository

No other computer includes a fixed IP address. Click **Next**, as shown in Figure 11-10 on page 352.

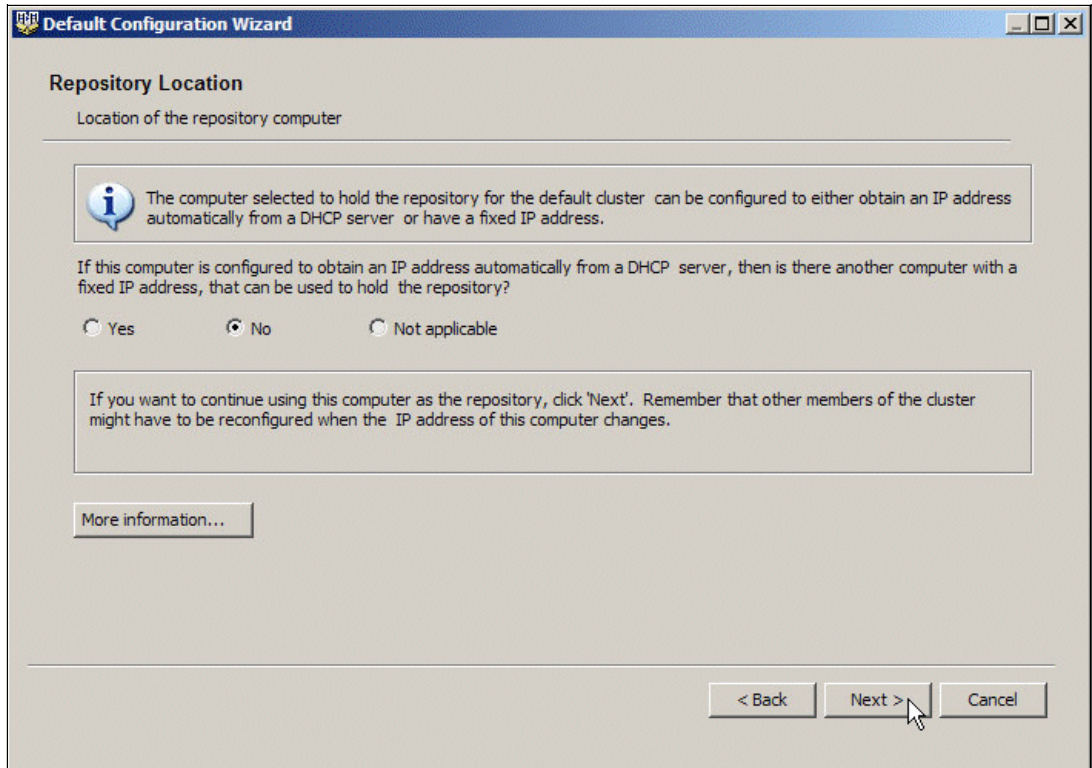


Figure 11-10 Repository location

Figure 11-11 shows the specified computer name. Click **Next** to continue.

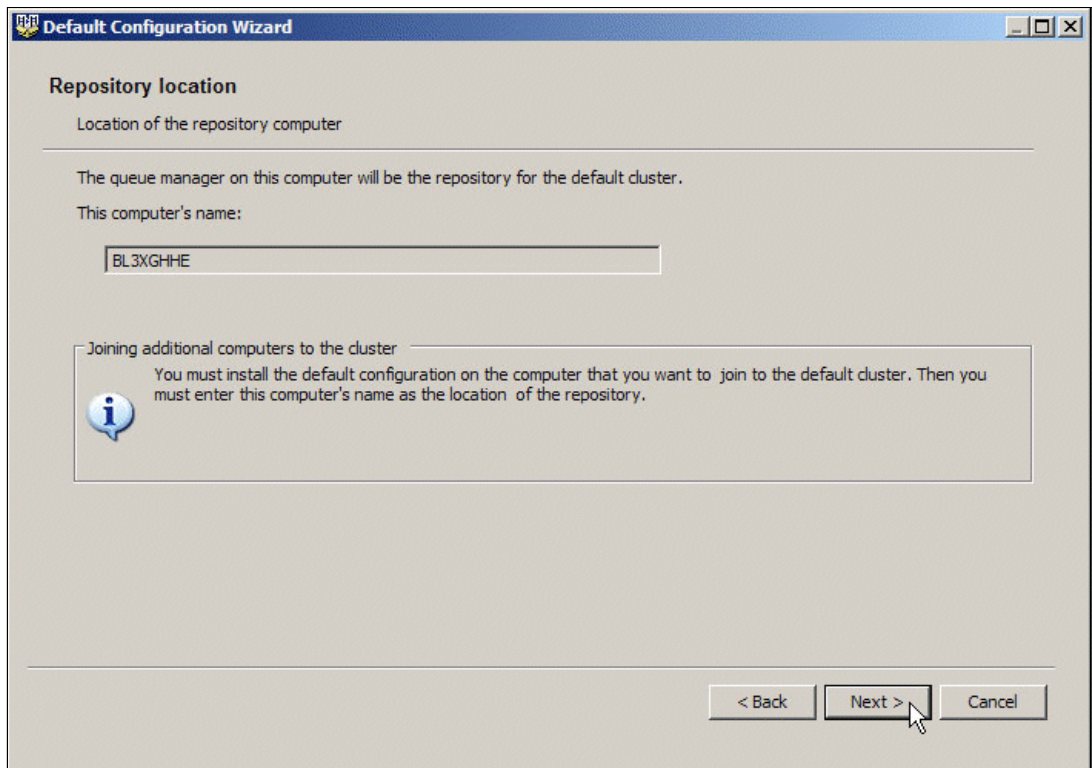


Figure 11-11 Computer name for repository



Figure 11-12 shows the default configuration. If it is correct, click **Finish** to continue.

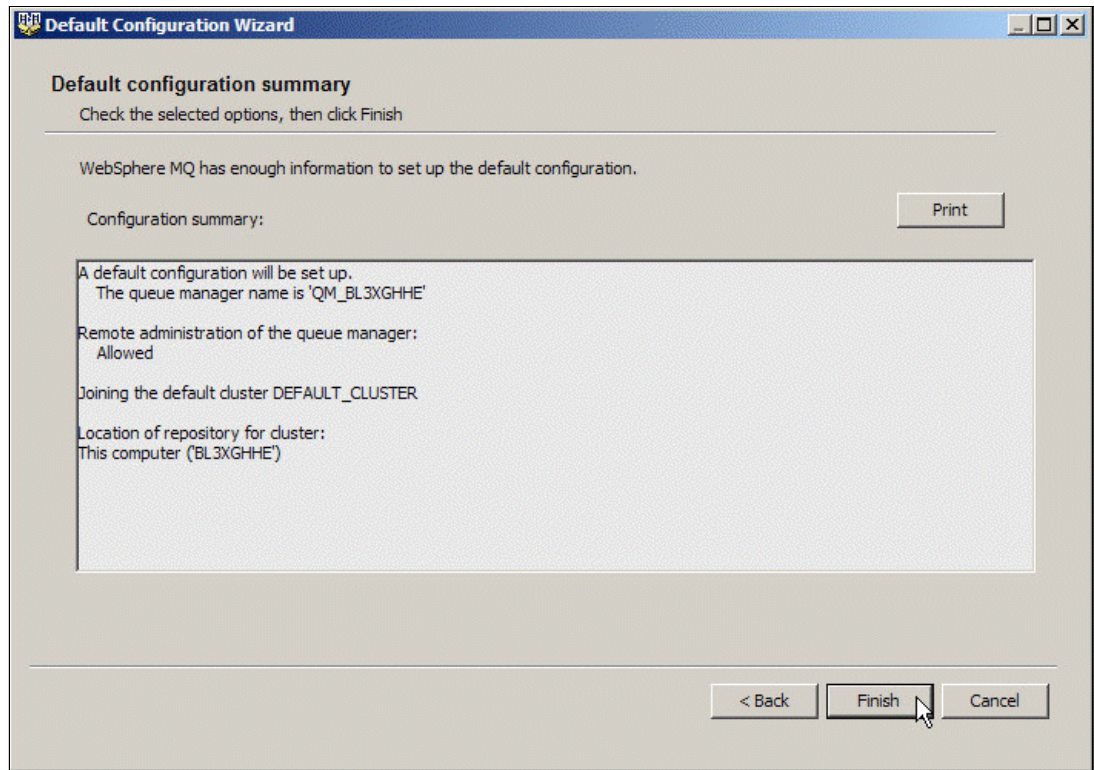


Figure 11-12 Default configuration summary

Figure 11-13 shows that the default configuration is complete. Click **Close**.

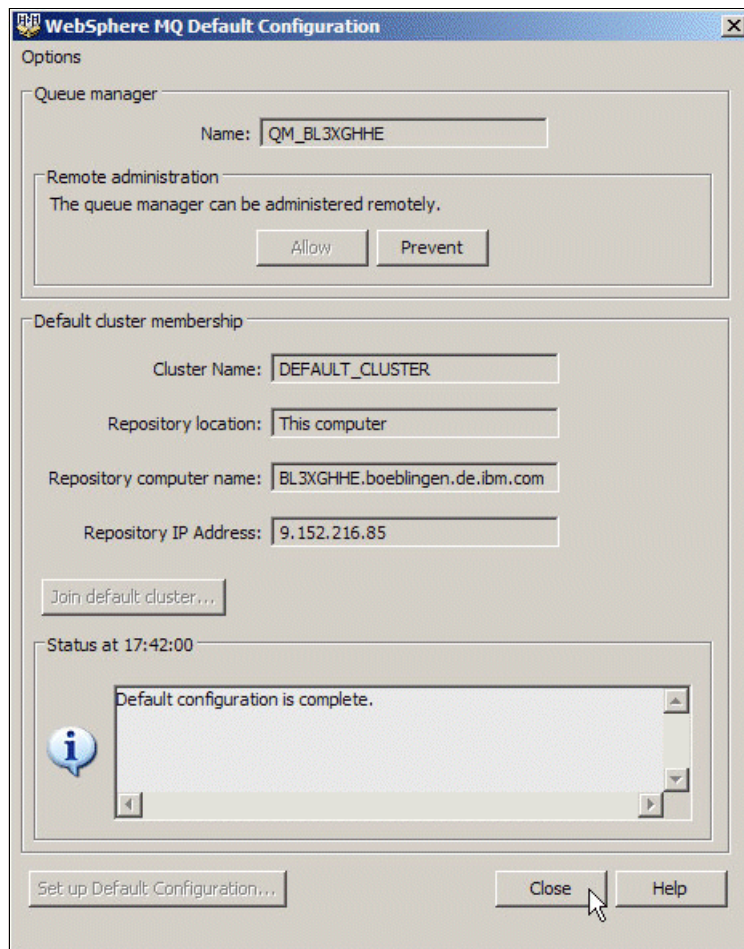


Figure 11-13 Default configuration completed

Figure 11-14 on page 355 shows the MQ Wizard window. Click **Finish**.



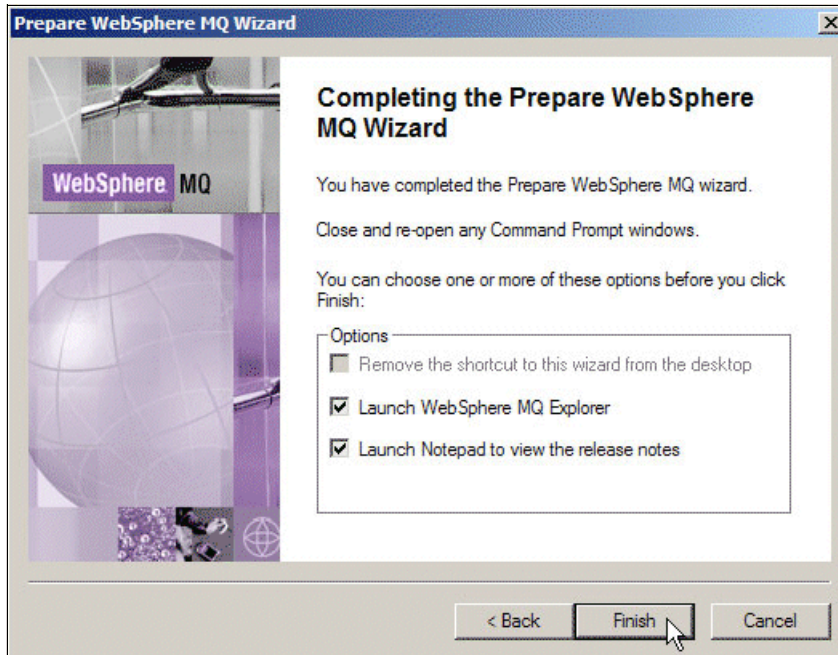


Figure 11-14 MQ Wizard

The WebSphere MQ Explorer is started, as shown in Figure 11-15.

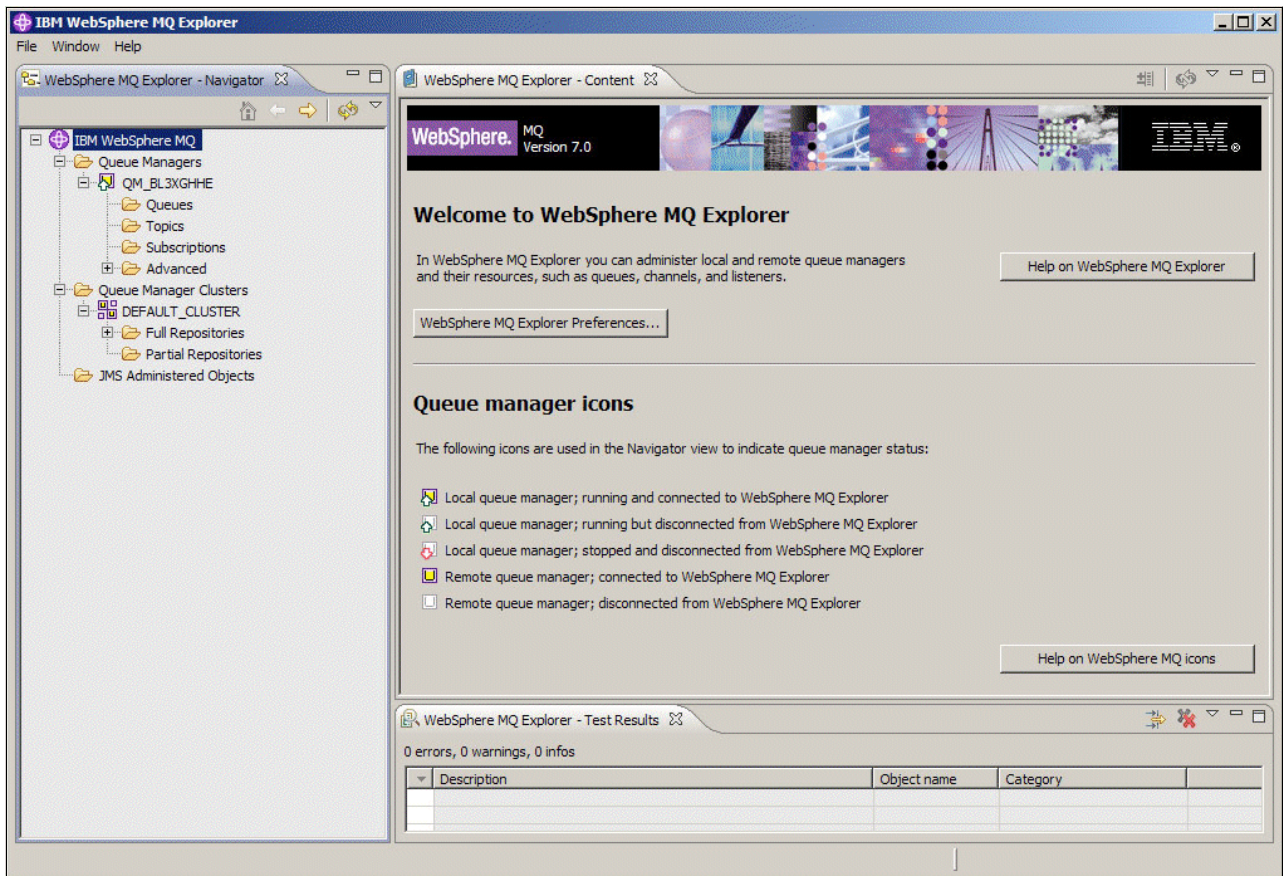


Figure 11-15 WebSphere MQ Explorer

## 11.3 Configuring WebSphere MQ

This section describes how to configure WebSphere MQ on both sides. Figure 11-16 shows the relationships of the various queues and channels.

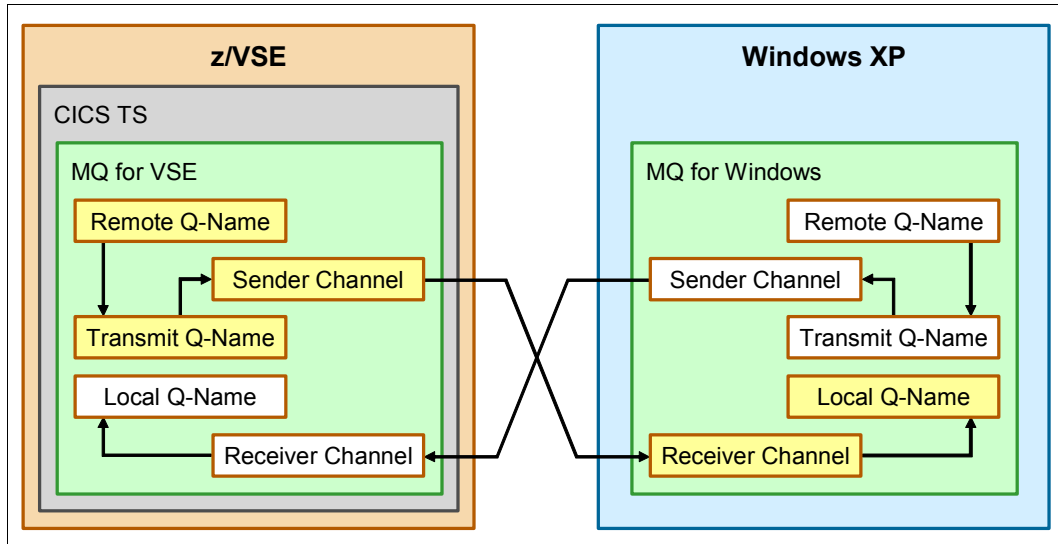


Figure 11-16 MQ queues and channels

Table 11-1 lists the names for the queues and channels that we used in our setup.

Table 11-1 MQ queue and channel names

Queues and channels	z/VSE	Windows
Queue Manager	QMGR.VSE	QM_BL3XGHHE
Local Q-Name	VSE42	WINXP
Remote Q-Name	WINXP	VSE42
Transmit Q-Name	XMT.WINXP	XMT.VSE42
Sender channel	VSE.TO.WIN	WIN.TO.VSE
Receiver channel	WIN.TO.VSE	VSE.TO.WIN

### 11.3.1 MQ configuration on z/VSE

As described in the Initial configuration section of *Using MQSeries for VSE, SG24-5647*, you must define the following information:

- ▶ z/VSE queue manager
- ▶ System queues
- ▶ Sender and receiver channels

First, you must establish the MQSeries environment in CICS through the MQSE transaction. At this point, the necessary security definitions should be made as described in 11.2.1, “MQ installation on z/VSE” on page 346.

## Defining the queue manager

After running the MQSE transaction, you can start the MQ master terminal (MQMT) transaction to define the z/VSE queue manager. Select option **1** (Configuration), and again select option **1** (Global System Definition).

The Global System Definition panel is displayed, as shown in Figure 11-17.

```
12/16/2008      IBM WebSphere MQ for z/VSE Version 3.0.0      DBDCCICS
11:17:41              Global System Definition              CIC1
MQWMSYS              Queue Manager Information              A000
Queue Manager . . . . . : QMGR.VSE
Description Line 1. . . . . :
Description Line 2. . . . . :
                    Queue System Values
Maximum Connection Handles.: 00000100      System Wait Interval : 00000030
Maximum Concurrent Queues .: 00000100      Max. Recovery Tasks  : 0000
Allow TDQ Write on Errors  : Y   CSMT      Local Code Page     . : 01047
Allow Internal Dump       . . . . . : Y       Subsystem id      . . . . . : MQV1
                    Queue Maximum Values
Maximum Q Depth . . . . . : 00100000      Maximum Global Locks.: 00001000
Maximum Message Size. . . . . : 00002048      Maximum Local Locks .: 00001000
Maximum Single Q Access . . . . . : 00000100
                    Global QUEUE /File Names
Configuration File. . . . . : MQFCNFG
LOG Queue Name. . . . . : SYSTEM.LOG
Dead Letter Name. . . . . : SYSTEM.DEAD.LETTER.QUEUE
Monitor Queue Name. . . . . : SYSTEM.MONITOR

Requested record displayed.
PF2=Return PF3=Quit PF4/Enter=Read      PF9=Com PF10=Log PF11=Evt PF12=Ext
```

Figure 11-17 Global system definition

At this point, we define the Queue Manager name and keep all other defaults.

Press PF6 to permanently update your queue manager definition.

## Defining the local queue

Define the local queue through transaction MQMT, and select option **1.2**. Then, press PF5 to continue (see Figure 11-18).

```
12/16/2008      IBM WebSphere MQ for z/VSE Version 3.0.0      DBDCCICS
11:19:05              Queue Main Options                    CIC1
MQWMQUE              SYSTEM IS ACTIVE                      A000

                    Default Q Manager. . . . . : QMGR.VSE

                    Object Type. . . . . :
                    L = Local Queue
                    M = Model Queue
                    R = Remote Queue
                    AQ = Alias Queue
                    AM = Alias Queue Manager
                    AR = Alias Reply Queue

                    Object Name. . . . . : VSE42

PF2=Return PF3=Quit PF4/Enter=Read PF5=Add PF6=Update
                    PF9=List PF12=Delete
```

Figure 11-18 Queue main options

The Local Queue Definition opens (see Figure 11-19).

```

12/16/2008      IBM WebSphere MQ for z/VSE Version 3.0.0      DBDCCICS
11:18:44        Queue Definition Record                    CIC1
MQWMQUE         QM - QMGR.VSE                               A000

                Local Queue Definition

Object Name. . . . . : VSE42
Description line 1 . . . . :
Description line 2 . . . . :

Put Enabled . . . . . : Y   Y=Yes, N=No
Get Enabled . . . . . : Y   Y=Yes, N=No

Default Inbound status . . : A   A=Active, I=Inactive
      Outbound status. . . : A   A=Active, I=Inactive

Dual Update Queue. . . . . :

Automatic Reorganize (Y/N) : N   Start Time: 0000 Interval: 0000
VSAM Catalog . . . . . :

PF2=Return  PF3=Quit  PF4/Enter=Read  PF5=Add  PF6=Update
                        PF9=List  PF10=Queue  PF12=Delete

```

Figure 11-19 Local queue definition

Press PF5 to open the Queue Extended Definition panel, as shown in Figure 11-20.

```

12/16/2008      IBM WebSphere MQ for z/VSE Version 3.0.0      DBDCCICS
11:20:55        Queue Extended Definition                  CIC1
MQWMQUE         A000

Object Name: VSE42

General          Maximums          Events
Type . . . : Local      Max. Q depth . . : 00100000  Service int. event: N
File name . . : MQFI001  Max. msg length: 00002048  Service interval . : 00000000
Usage . . . : N          Max. Q users . . : 00000100  Max. depth event . : N
Shareable . . : Y        Max. gbl locks . : 00001000  High depth event . : N
Dist.Lists . . : Y       Max. lcl locks . : 00001000  High depth limit . : 000
                                           Low depth event . . : N
                                           Low depth limit . . : 000

Triggering
Enabled . . . : N        Transaction id.:
Type . . . . :          Program id . . . :
Max. starts: 0001      Terminal id . . :
Restart . . . : N       Channel name . . :
User data . . . :
:

PF2=Return  PF3=Quit  PF4/Enter=Read  PF5=Add  PF6=Update
                        PF9=List  PF10=Queue

```

Figure 11-20 Queue extended definition

Enter the name of a VSAM file to be used for this queue (for example, MQFI001) and press PF5 again to finish.

## Defining the transmission queue

Specify the name of the transmission queue, as shown in Figure 11-21.

```
12/16/2008      IBM WebSphere MQ for z/VSE Version 3.0.0      DBDCCICS
09:52:02              Queue Definition Record          CIC1
MQWMQUE        QM - QMGR.VSE                          A000

                Local Queue Definition

Object Name. . . . . : XMT.WINXP
Description line 1 . . . . . :
Description line 2 . . . . . :

Put Enabled . . . . . : Y      Y=Yes, N=No
Get Enabled . . . . . : Y      Y=Yes, N=No

Default Inbound status . . . : A      A=Active, I=Inactive
      Outbound status. . . : A      A=Active, I=Inactive

Dual Update Queue. . . . . :

Automatic Reorganize (Y/N) : N      Start Time: 0000 Interval: 0000
VSAM Catalog . . . . . :

PF2=Return  PF3=Quit  PF4/Enter=Read  PF5=Add  PF6=Update
                        PF9=List  PF10=Queue  PF12=Delete
```

Figure 11-21 Queue definition record for transmission queue

Press PF10 (Queue) to open the Queue Extended Definition panel, as shown in Figure 11-22.

```
12/16/2008      IBM WebSphere MQ for z/VSE Version 3.0.0      DBDCCICS
09:49:51              Queue Extended Definition          CIC1
MQWMQUE        A000

Object Name: XMT.WINXP

General                      Maximums                      Events
Type . . . : Local          Max. Q depth . . : 00100000  Service int. event: N
File name . : MQFO001       Max. msg length: 00002048  Service interval . : 00000000
Usage . . . : T             Max. Q users . . : 00000100  Max. depth event . : N
Shareable . : Y             Max. gbl locks . : 00001000  High depth event . : N
Dist.Lists . : Y           Max. lcl locks . : 00001000  High depth limit . : 000
                                                Low depth event . . : N
                                                Low depth limit . . : 000

Triggering
Enabled . . : Y             Transaction id.:
Type . . . : E             Program id . . : MQPSEND
Max. starts: 0001         Terminal id . . :
Restart . . : N           Channel name . : VSE.TO.WIN
User data . . :
:
PF2=Return  PF3=Quit  PF4/Enter=Read  PF5=Add  PF6=Update
                        PF9=List  PF10=Queue
```

Figure 11-22 Queue extended definition for transmission queue

Change the Usage parameter to T (for transmission) as shown, and then, press PF5 to finish.

## Defining the remote queue

Define the remote queue for processing outgoing messages to Windows. Use MQMT path 1.2 to define a remote queue with name WINXP. Then, press PF5 to open the panel, as shown in Figure 11-23 on page 360.

```

12/16/2008      IBM WebSphere MQ for z/VSE Version 3.0.0      DBDCCICS
11:22:41              Queue Definition Record          CIC1
MQWMQUE        QM - QMGR.VSE                          A000

                Remote Queue Definition

Object Name. . . . . : WINXP
Description line 1 . . . . . :
Description line 2 . . . . . :

Put Enabled . . . . . : Y    Y=Yes, N=No
Get Enabled . . . . . : Y    Y=Yes, N=No

Remote Queue Name. . . . . : WINXP
Remote Queue Manager Name. : QM_BL3XGHHE
Transmission Queue Name. . : XMT.WINXP

PF2=Return  PF3=Quit  PF4/Enter=Read  PF5=Add  PF6=Update
                PF9=List  PF12=Delete

```

Figure 11-23 Remote queue definition

Enter the system-specific parameters for Remote Queue Name, Remote Queue Manager Name, and Transmission Queue Name.

Press PF5 to add the remote queue.

### Defining the sender channel

Channels are defined through transaction MQMT, options 1 (Configuration) and 3 (Channel Definitions). Set parameter Type to **S** (sender) and press PF5 to add (see Figure 11-24).

```

12/16/2008      IBM WebSphere MQ for z/VSE Version 3.0.0      DBDCCICS
11:24:03              Channel Record                    CIC1
MQWMCHN          A000
Channel : VSE.TO.WIN
Desc. . : SENDER CHANNEL TO WINXP
Protocol: T (L/T)  Type : S (S=Snd/R=Rcv/V=Srv/Q=Req/C=svrConn)  Enabled : Y

Sender/Server
Remote TCP/IP port . . . . . : 01414      Short/Long retry count . . : 000000000
Get retry number . . . . . : 00000000    Short retry interval . . . : 000000000
Get retry delay (secs) . . . . . : 00000000  Long retry interval . . . : 000000000
Convert msgs(Y/N). . . . . : N           Batch interval . . . . . : 000000000
Transmission queue name. . : XMT.WINXP
TP name. . :

Sender/Receiver/Server/Requester
Connection : 9.152.222.125
Max Messages per Batch . . : 000001      Message Sequence Wrap . . . : 000999999
Max Message Size . . . . . : 0002048      Dead letter store(Y/N) . . : N
Max Transmission Size . . . : 032766      Split Msg(Y/N) . . . . . : N
Max TCP/IP Wait . . . . . : 000000

F2=Return  PF3=Quit  PF4=Read  PF5=Add  PF6=Upd  PF9=List  PF10=SSL  PF11=Ext  PF12=Del

```

Figure 11-24 Channel record definition for sender

The parameter Connection specifies the IP address of the Windows PC. It should include a static IP address. When DHCP is used, ensure that you always include the same IP address. The parameter Message Sequence Wrap must match with the corresponding definition in Windows. For more information, see “Defining the receiver channel” on page 372.



## Defining the receiver channel

Use transaction MQMT, options 1 (Configuration) and 3 (Channel Definitions) again to define the receiver channel. Set parameter Type to **R** (receiver) (see Figure 11-25).

```
12/16/2008      IBM WebSphere MQ for z/VSE Version 3.0.0      DBDCCICS
11:24:42              Channel Record                      CIC1
MQWMCHN                          A000
Channel   : WIN.TO.VSE
Desc.    : RECEIVER CHANNEL FROM WINXP TO VSE
Protocol: T (L/T)  Type : R (S=Snd/R=Rcv/V=Srv/Q=Req/C=svrConn) Enabled : Y

Sender/Server
Remote TCP/IP port . . . . . : 01414      Short/Long retry count . . : 000000000
Get retry number . . . . . : 00000002    Short retry interval . . . : 000000000
Get retry delay (secs) . . . : 00000010  Long retry interval . . . : 000000000
Convert msgs(Y/N) . . . . . : N          Batch interval . . . . . : 000000000
Transmission queue name. . . :
TP name. . . :

Sender/Receiver/Server/Requester
Connection :
Max Messages per Batch . . : 000050      Message Sequence Wrap . . : 000999999
Max Message Size . . . . . : 0002048    Dead letter store(Y/N) . . : N
Max Transmission Size . . : 032766      Split Msg(Y/N) . . . . . : N
Max TCP/IP Wait . . . . . : 000000

F2=Return PF3=Quit PF4=Read PF5=Add PF6=Upd PF9=List PF10=SSL PF11=Ext PF12=Del
```

Figure 11-25 Channel record definition for receiver

Press PF5 to add the new definition. The parameter Message Sequence Wrap must match with the related value in the sender channel on Windows. For more information, see “Defining the sender channel” on page 370.

## Defining batch communications

Batch communications are necessary when you want to access MQ from batch.

Start MQMT and enter options 1 (Configuration) and 1 (Global System Definition). Then, press PF9 (Comms). Specify **Y** for Batch Int. auto-start. Then, press PF6 to update the definition (see Figure 11-26).

```
12/16/2008      IBM WebSphere MQ for z/VSE Version 3.0.0      DBDCCICS
11:25:32              Global System Definition            CIC1
MQWMSYS                          Communications Settings      A000

TCP/IP settings                      Batch Interface settings
TCP/IP listener port : 01414          Batch Int. identifier: MQBISERV
Licensed clients . . : 00000          Batch Int. auto-start: Y
Adopt MCA . . . . . : N
Adopt MCA Check . . : N

Channel Auto-Definition
Auto-definition . . . : Y
Auto-definition exit :

SSL parameters
Key-ring sublibrary : CRYPTO.KEYRING
Key-ring member . . : MQ02

PCF parameters
System command queue : SYSTEM.ADMIN.COMMAND.QUEUE
System reply queue . : SYSTEM.ADMIN.REPLY.QUEUE
Cmd Server auto-start: N
Cmd Server convert . : N
Cmd Server DLQ store : N

PF2=Queue Manager details PF3=Quit PF4/Enter=Read PF6=Update
```

Figure 11-26 Communication settings

## Defining log settings

Sometimes having all MQ messages on the operator console is convenient. To define the log settings, start MQMT, path 1.1 and press PF10. As shown in Figure 11-27, column C specifies whether MQ messages also are written to the console.

```
12/16/2008      IBM WebSphere MQ for z/VSE Version 3.0.0      DBDCCICS
11:26:03              Global System Definition          CIC1
MQWMSYS              Log and Trace Settings              A000

      Log Settings      Q C      Trace Settings

Informational . . . : Y Y      MQI calls . . . . . : N
Warning . . . . . : Y Y      Communication . . . . : N
Error . . . . . : Y Y      Reorganization . . . . : N
Critical . . . . . : Y Y      Data conversion . . . . : N
                        System . . . . . : N

      - and/or -

Communication . . . : Y Y
Reorganization . . : Y Y
System . . . . . : Y Y

PF2=Queue Manager details  PF3=Quit  PF4/Enter=Read  PF6=Update
```

Figure 11-27 Log and trace settings

## Starting MQ on z/VSE

Now, you can start MQ on z/VSE through MQMT, option 2 (Operations) and 4 (Initialization/Shutdown of System). The console displays the following messages:

```
MQI0030I - WMQ for z/VSE system starting
MQI0035I - WMQ for z/VSE licensed support for      0000 clients
MQI0040I - WMQ for z/VSE system started
MQI0200I - MQI000000I Queue manager started
MQI0200I - MQI006041I TCP/IP listener started
MQI0100I - WMQ Batch Interface (MQBISERV) started
```

Check the CICS job output for any security violations that are caused by transactions that are not defined to BSM.

## 11.3.2 MQ configuration on Windows

Basic network configuration was done during the WebSphere MQ installation. Start the MQ Explorer to continue with the z/VSE specific definitions.

### Defining the local queue

Select **Queues** → **New** → **Local Queue**, as shown in Figure 11-28 on page 363.

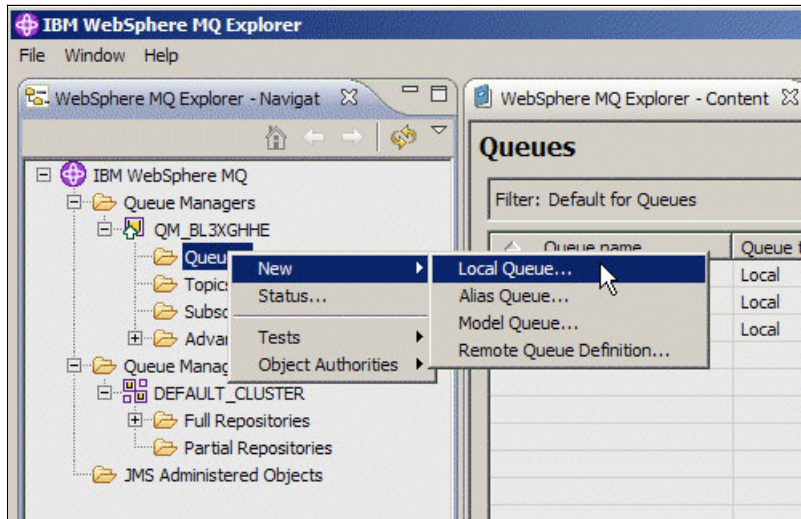


Figure 11-28 Select new local queue

Enter the local queue name WINXP, as shown in Figure 11-29.

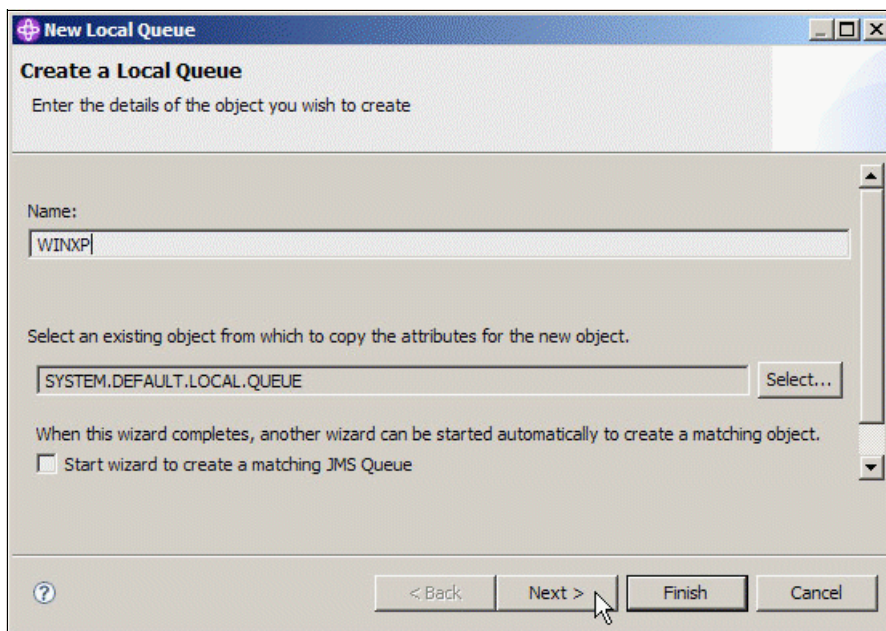


Figure 11-29 New local queue

Click **Next**. The Change properties panel opens, as shown in Figure 11-30 on page 364.

Select **Triggering**.

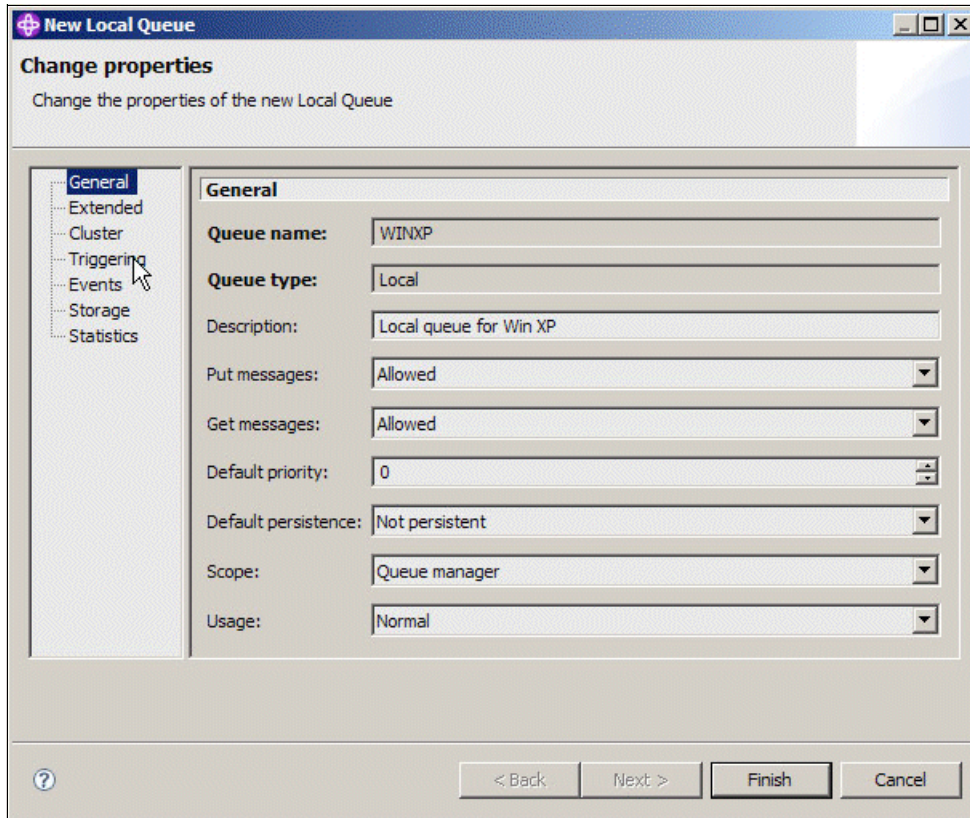


Figure 11-30 Change properties of local queue

As shown in Figure 11-31, change the triggering properties. Select Trigger control **On** and Trigger type **Every**. Then, click **Finish**.

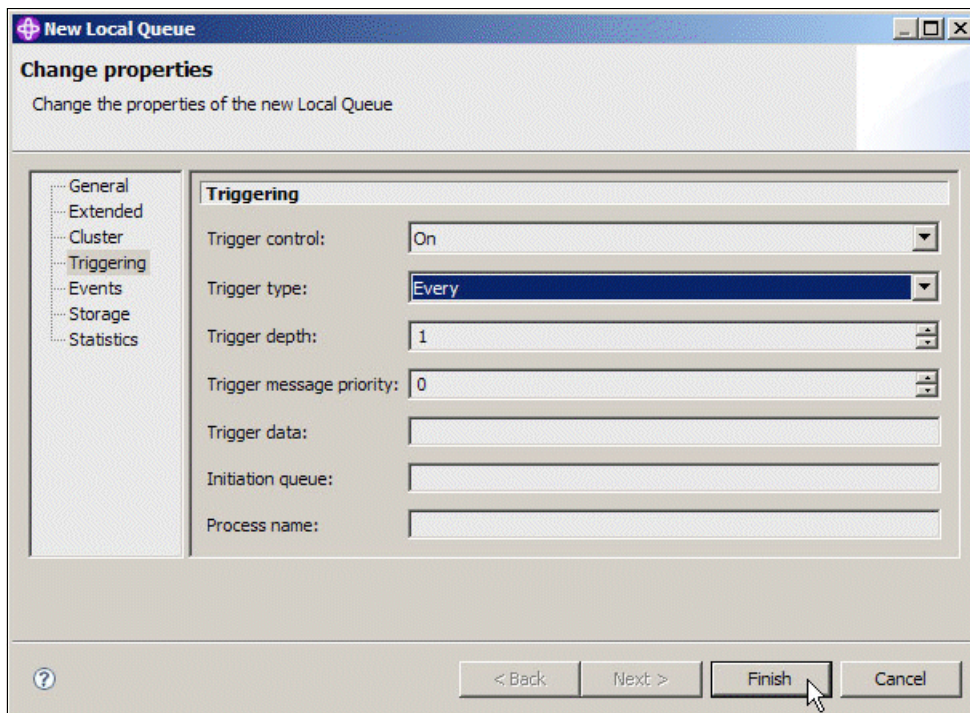


Figure 11-31 Change triggering properties



## Defining the transmit queue

Select **Queues** → **New** → **Local queue** to open the New Local Queue dialog, as shown in Figure 11-32.

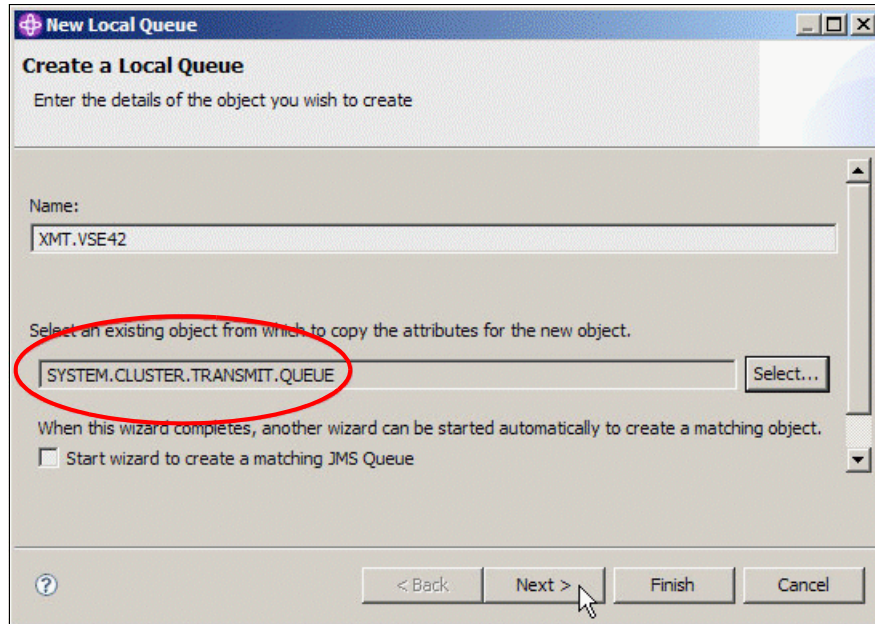


Figure 11-32 Create a local queue

Click **Select** and choose **SYSTEM.CLUSTER.TRANSMIT.QUEUE** as the model for the queue. In the Name field, enter the name of the transmit queue, `XMT.VSE42`. Then, click **Next**.

In the Change properties panel, the field Usage should display Transmission, as shown in Figure 11-33. Select **Triggering**.

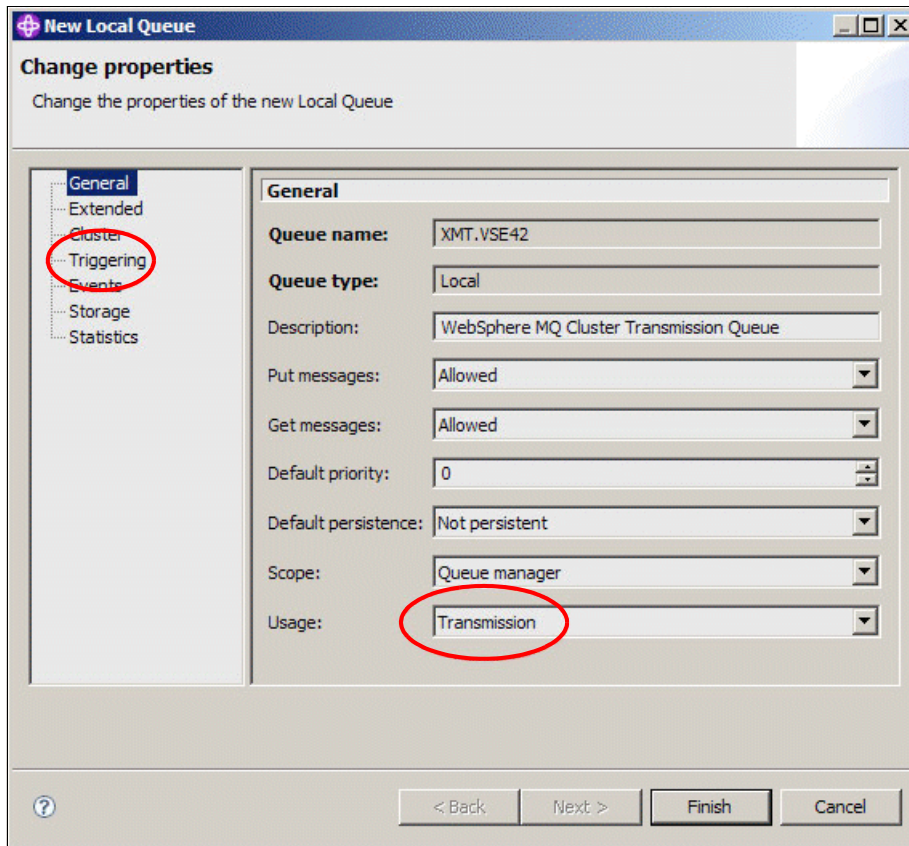


Figure 11-33 Change local queue properties

For triggering, specify trigger control **On** and trigger type **Every**, as shown in Figure 11-34. Click **Finish**.

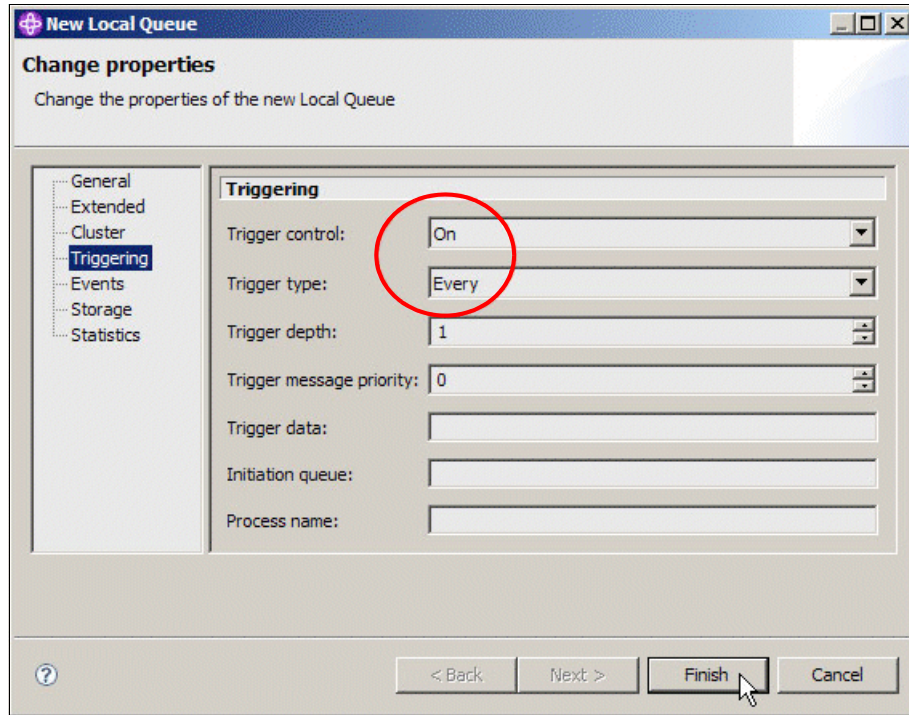


Figure 11-34 Change triggering properties

## Defining the remote queue

Select **Queues** → **New** → **Remote Queue Definition**, as shown in Figure 11-35.

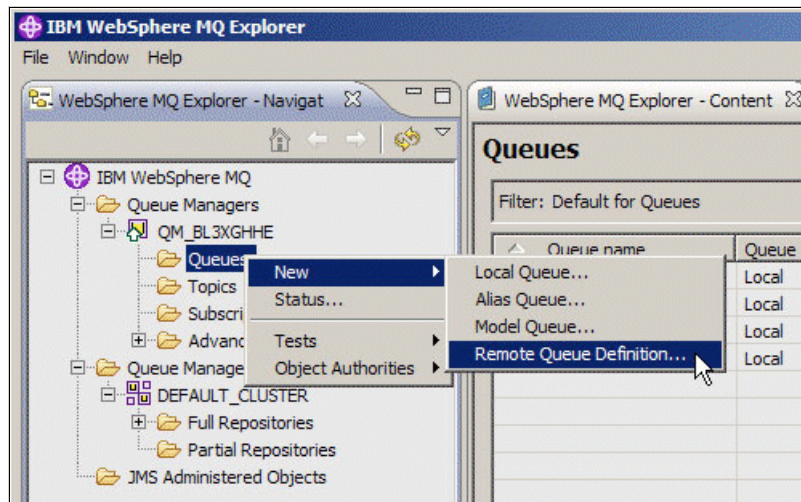


Figure 11-35 Select new remote queue definition

Enter the name of the remote queue, VSE42, as shown in Figure 11-36.

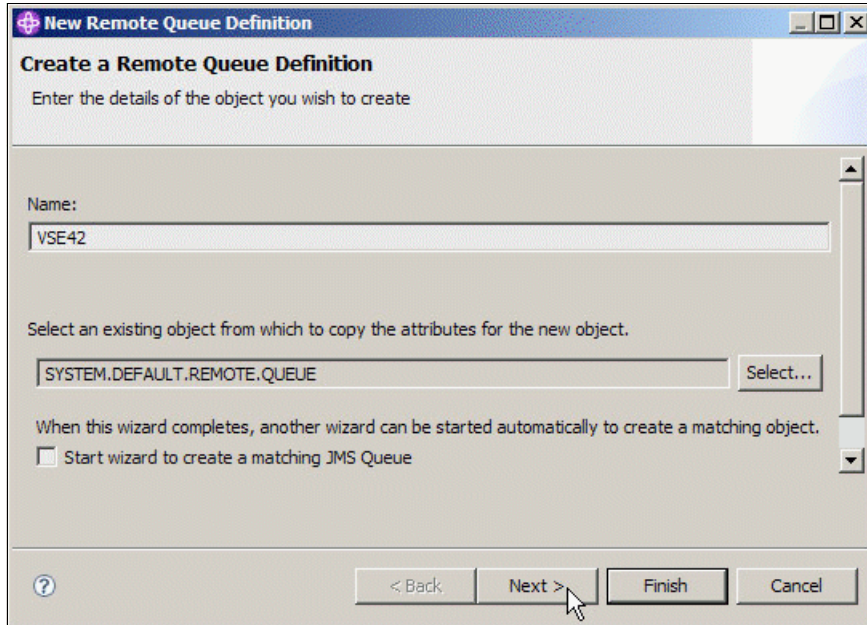


Figure 11-36 Create the remote queue definition

Click **Next**.



In the window that is shown in Figure 11-37, enter the remote queue name (VSE42), the remote queue manager (QMGR.VSE), and the transmission queue name (XMT.VSE42).

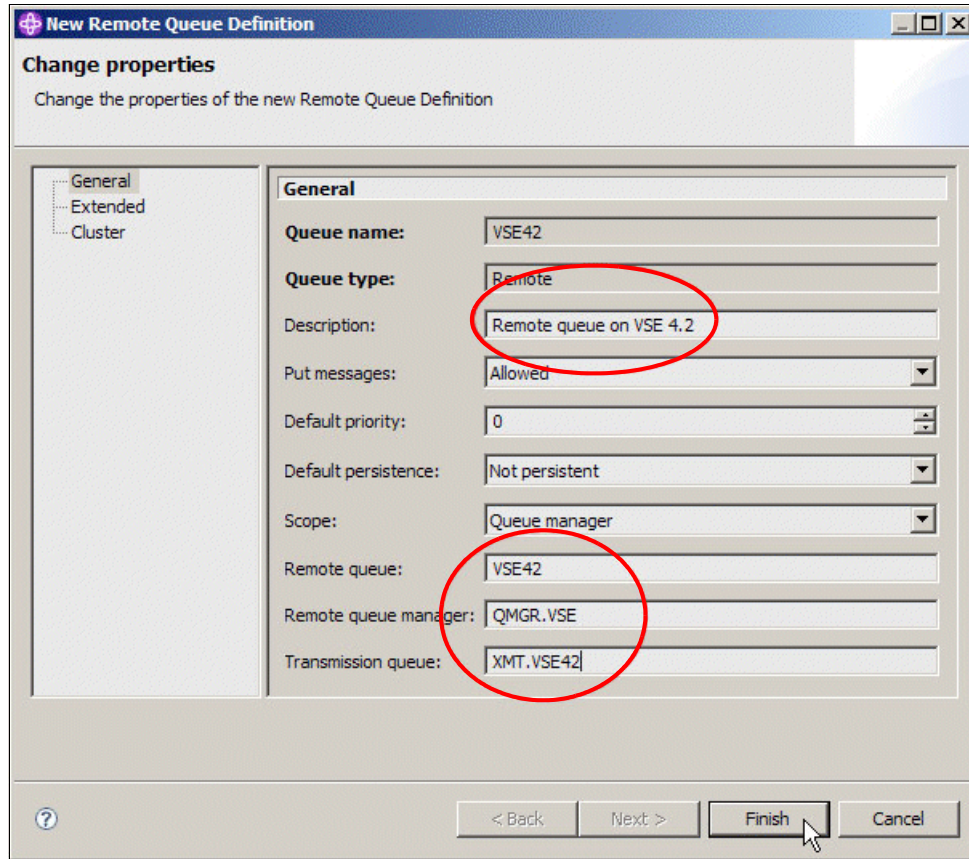


Figure 11-37 Change the properties of the remote queue definition

Click **Finish**.

Figure 11-38 shows the available queues.

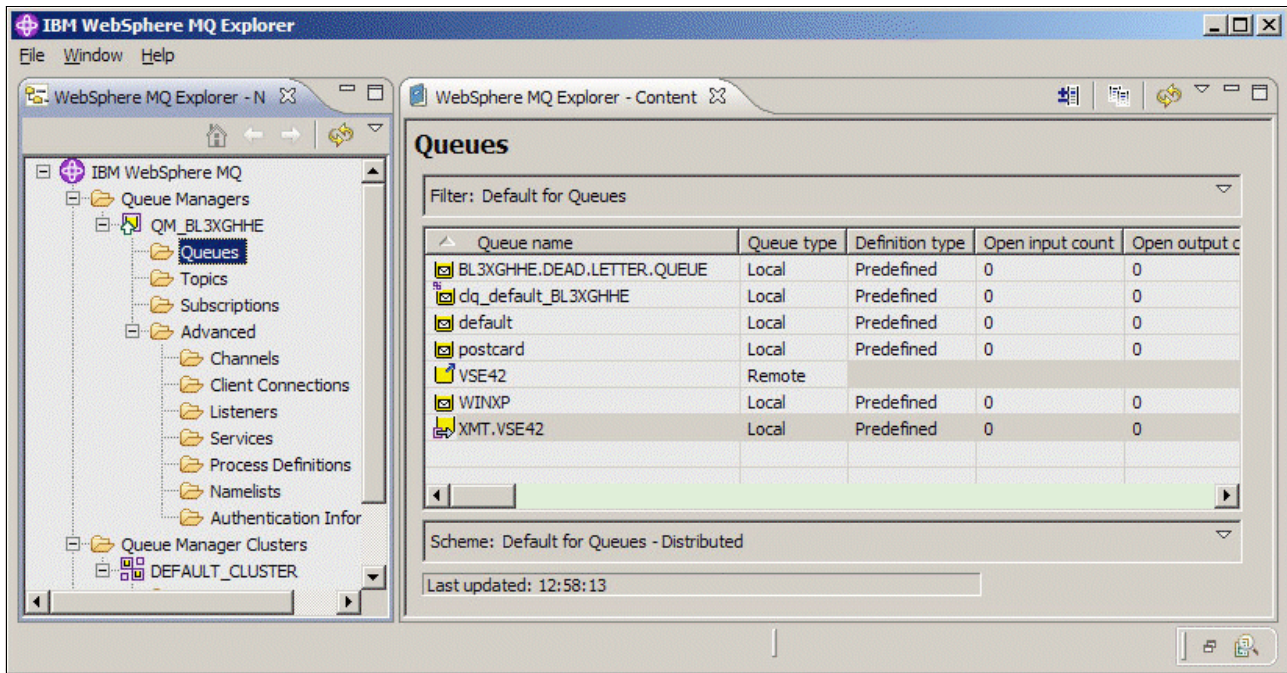


Figure 11-38 Queues

### Defining the sender channel

Select **Channels** → **New** → **Sender Channel**, as shown in Figure 11-39.

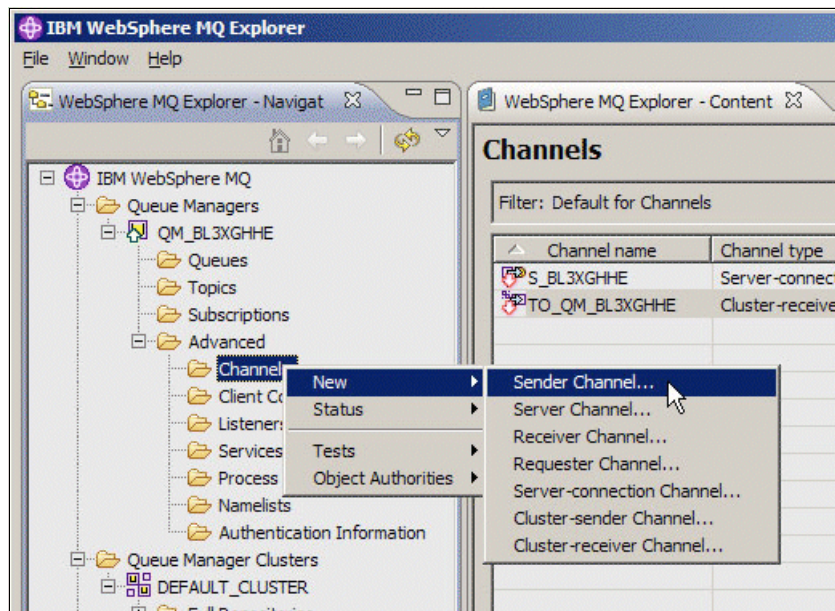


Figure 11-39 Define new sender channel

Enter the name of the sender channel (WIN.TO.VSE), as shown in Figure 11-40 on page 371.



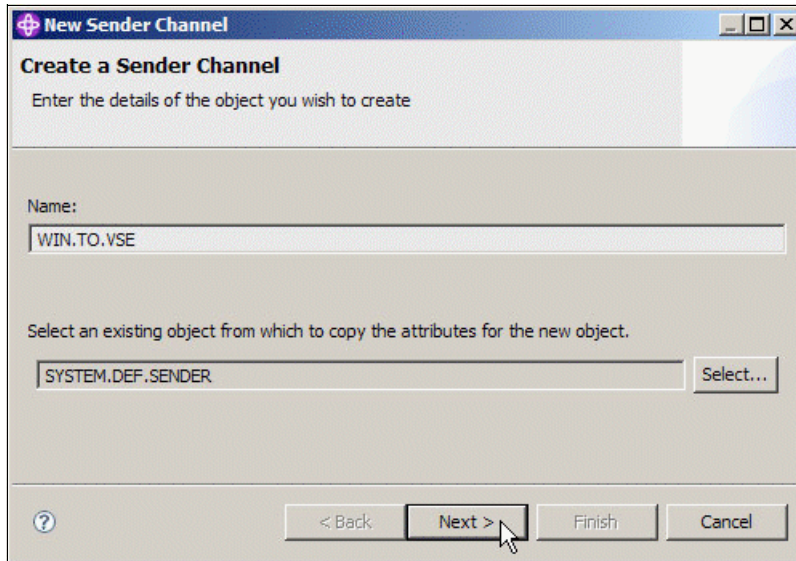


Figure 11-40 Create a sender channel

Click **Next**.

As shown in Figure 11-41, enter the IP address of your z/VSE system and the related transmission queue name.

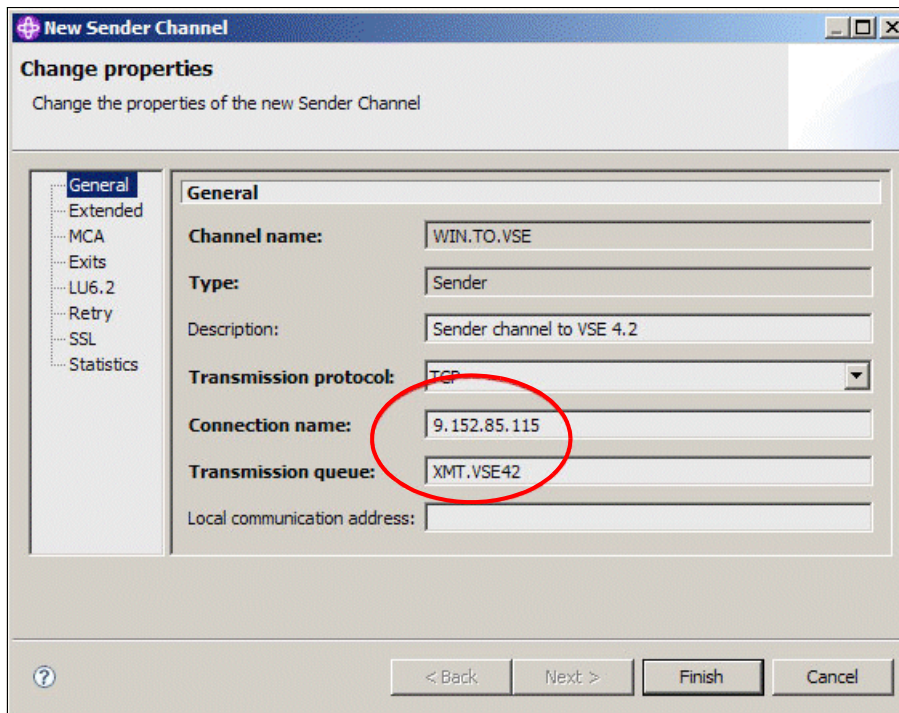


Figure 11-41 Change the general properties of the new sender channel

Select **Extended**.

Figure 11-42 shows the values for maximum message length and sequence number wrap. These values must match the values of the corresponding receiver channel on the z/VSE side. For more information, see “Defining the receiver channel” on page 361.

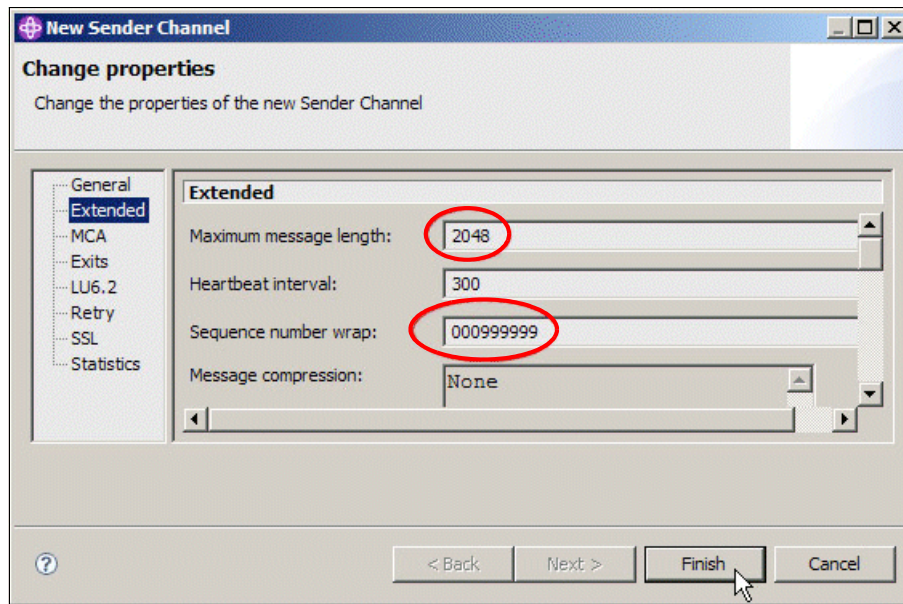


Figure 11-42 Change the property extended

## Defining the receiver channel

Select **Channels** → **New** → **Receiver Channel**, as shown in Figure 11-43.

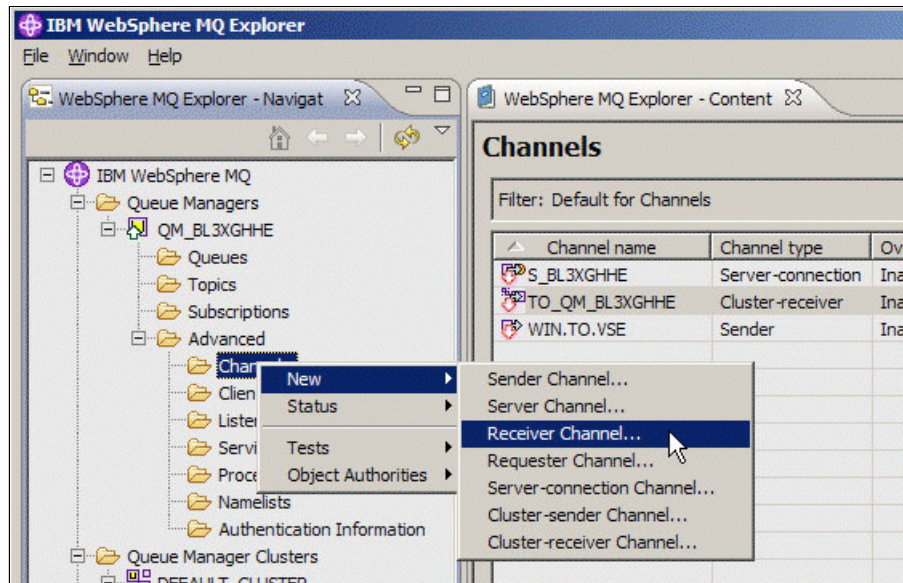


Figure 11-43 Define a new receiver channel



Enter the name of the receiver channel (VSE.TO.WIN), as shown in Figure 11-44.

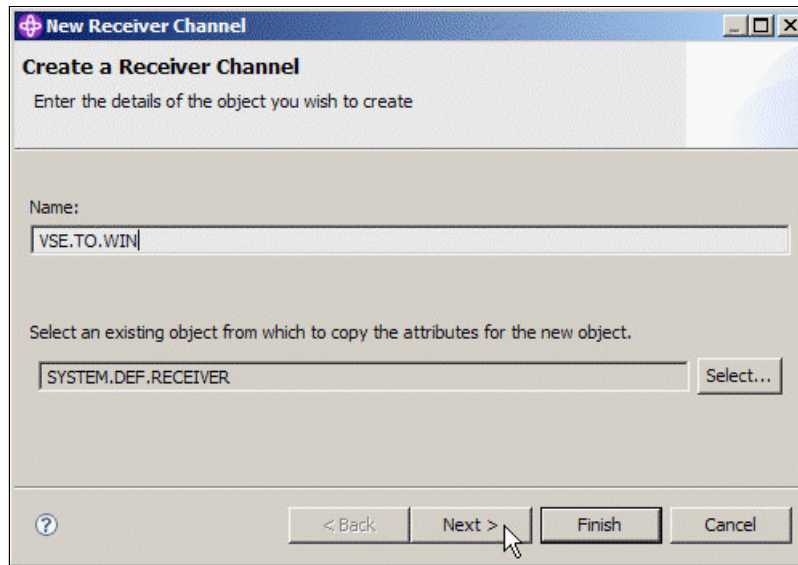


Figure 11-44 Create a receiver channel

Click **Next**.

In the next window, select **Extended**, as shown in Figure 11-45. The values for Maximum message length and Sequence number wrap must match the values of the corresponding sender channel on the z/VSE side. For more information, see “Defining the sender channel” on page 360.

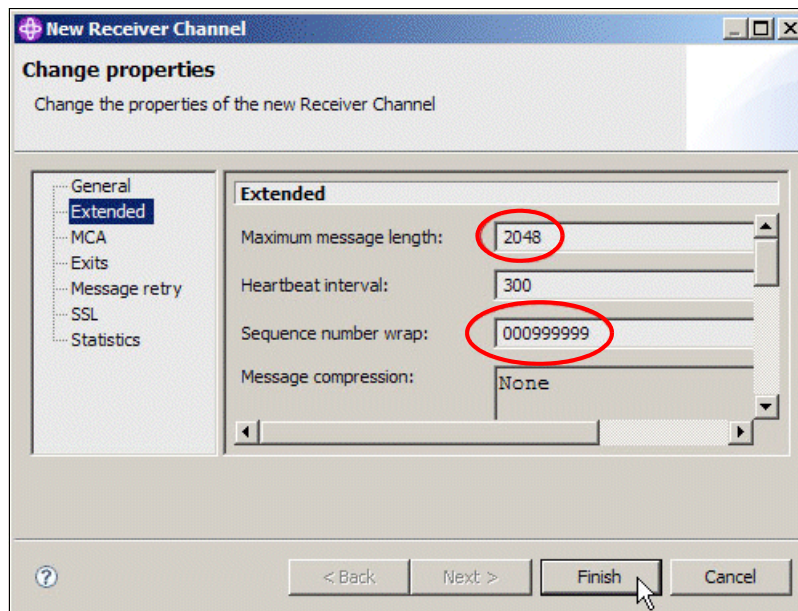


Figure 11-45 Change the properties of the receiver channel

Figure 11-46 shows your channels.

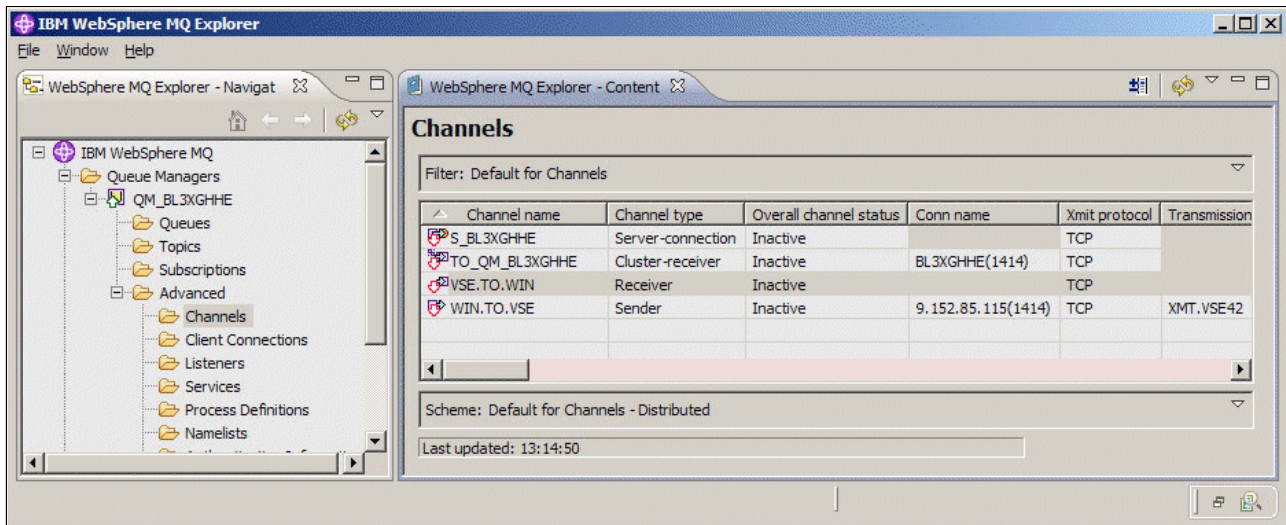


Figure 11-46 List of channels

Before you can send any message, start the sender channel, as shown in Figure 11-47.

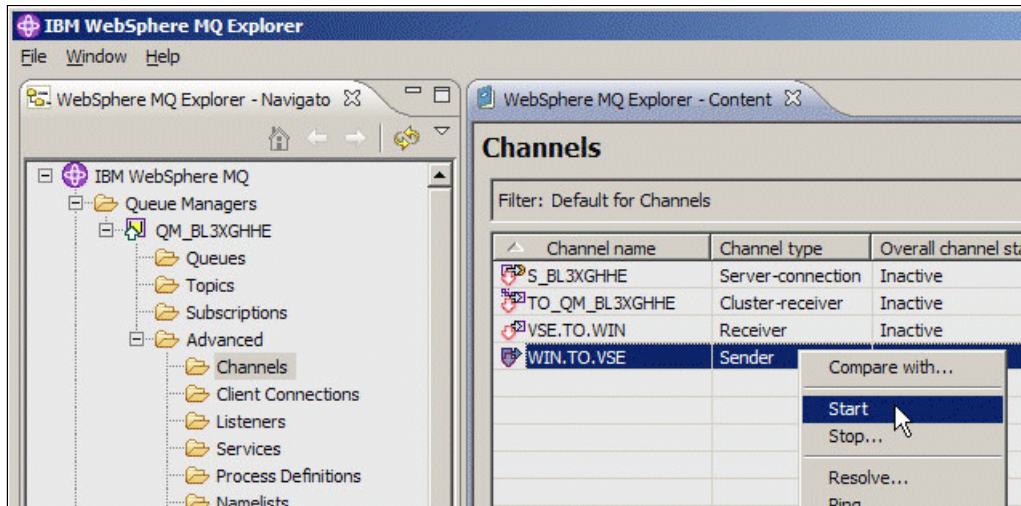


Figure 11-47 Start sender channel

The new overall channel status should now be running for the sender channel.



### 11.3.3 Testing the setup

Test your setup by sending several test messages from Windows to z/VSE and vice versa.

#### Sending a test message to z/VSE

First, start transaction MQMT on z/VSE and enter **3** (Monitoring) and **1** (Monitor queue). Then, select the local queue VSE42. Currently, no messages are in the queue. QDEPTH value is 0 (zero), as shown in Figure 11-48.

```
12/16/2008          IBM WebSphere MQ for z/VSE Version 3.0.0          DBDCCICS
11:30:38           Monitor Queues                                CIC1
MQWMMOQ                                                    A000

                    QUEUING SYSTEM IS ACTIVE
                    DETAIL QUEUE INFORMATION

VSE42
INBOUND:  STATUS B  ENABLED Y  OPEN Q          1
OUTBOUND: STATUS I  ENABLED Y  OPEN Q          0

BOTH:     FIQ      0    LIQ      0    GETS      0    QDEPTH    0

Enter=Refresh  PF2=Return  PF3=Exit  PF10=List
```

Figure 11-48 Detail queue information

In Windows, start the sender channel if it is not started.

Then, right-click the remote queue **VSE42** and select **Put Test Message**, as shown in Figure 11-49.

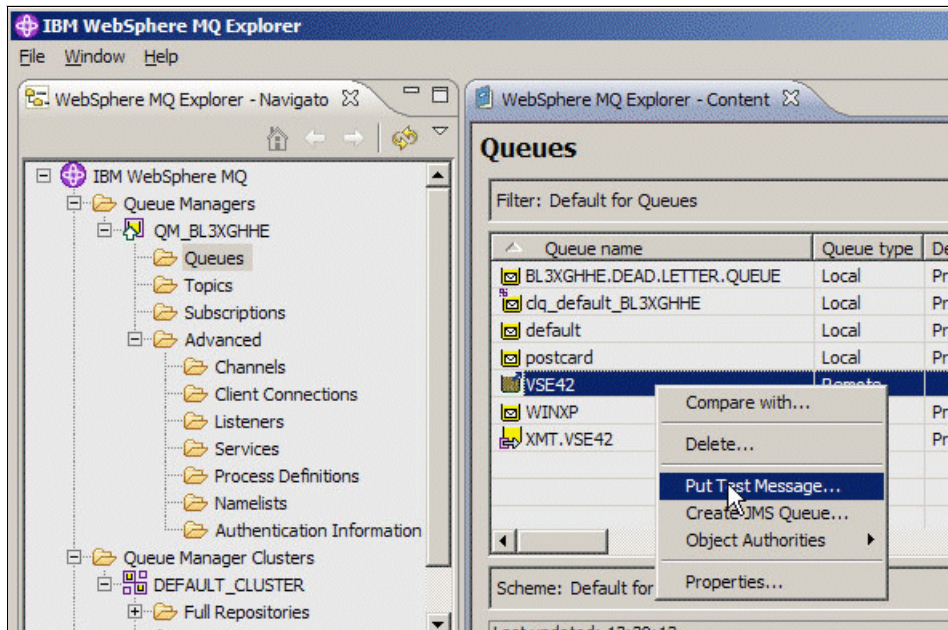


Figure 11-49 MQ Explorer list of queues



In the next window, enter text for the message (see Figure 11-50).

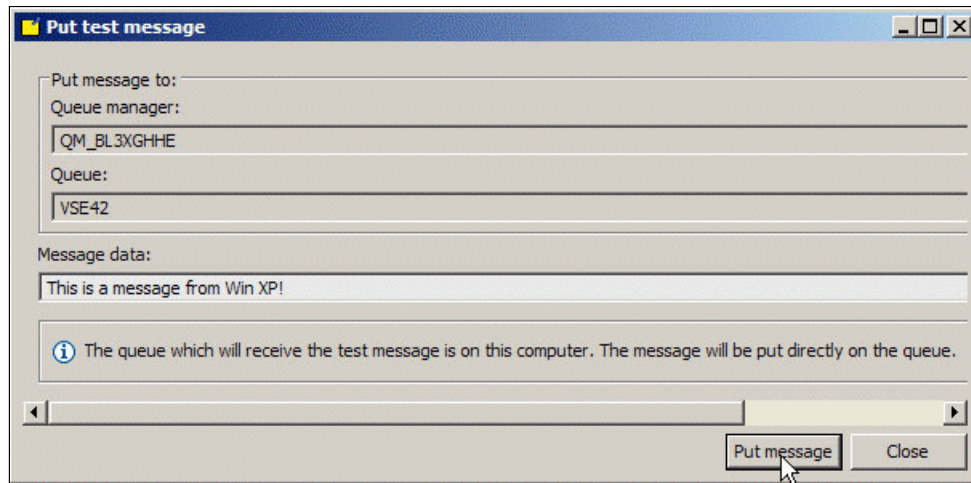


Figure 11-50 Put a test message

Click **Put message** to send this message.

On the z/VSE side, view the message by pressing Enter to refresh the panel.

Figure 11-51 shows that the QDEPTH value increased by one. You can now browse the queue to see the message.

```

12/16/2008          IBM WebSphere MQ for z/VSE Version 3.0.0          DBDCCICS
11:30:38           Monitor Queues                                   CIC1
MQWMMOQ                                                    A000

                    QUEUING SYSTEM IS ACTIVE
                    DETAIL QUEUE INFORMATION

VSE42
INBOUND:  STATUS B  ENABLED Y  OPEN Q          1
OUTBOUND: STATUS I  ENABLED Y  OPEN Q          0

BOTH:     FIQ       0    LIQ       1  GETS       0    QDEPTH     1
Enter=Refresh PF2=Return PF3=Exit PF10=List

```

Figure 11-51 Detail queue information

Press PF2 twice to return to the main MQMT panel and select **4** (Browse Queue Records). Enter the queue name to browse (VSE42).



You now successfully defined the MQ environment between z/VSE and Windows. For more information about enhancing this environment to use SSL for secured data transmission, see 11.4, “Configuring for SSL” on page 378.

## 11.4 Configuring for SSL

SSL is available in the following types:

- ▶ SSL server authentication
- ▶ SSL client authentication

The type to use is configured at the server side (receiver channel). With MQSeries, both sides can be server or client at the same time when messages are exchanged. When sending an MQ message, the sender is the client; when receiving an MQ message, the receiver is the server. Consider this information when you implement SSL with MQSeries.

In the following setup, first a set of keyring members (PRVK, ROOT, and CERT) is created with Keyman/VSE and uploaded to z/VSE. Then, the two certificates are stored in an MQ key database file on Windows XP.

With the setup that is shown in Figure 11-55, both sides can be SSL server or SSL client. When z/VSE is the server (receiver), the certificate that is contained in the CERT member is sent to the client (sender). When Windows is the server, the user certificate in the key database is sent to z/VSE.

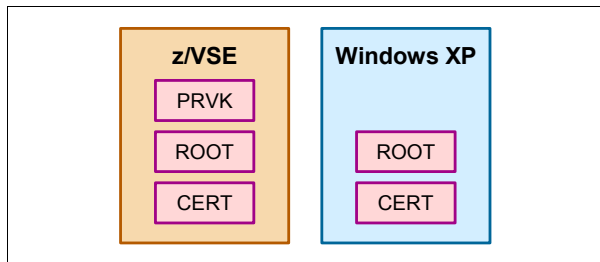


Figure 11-55 keyring setup on z/VSE and Windows

### 11.4.1 Creating the keys and certificates

Start the Keyman/VSE tool and create an RSA key pair, as shown in Figure 11-56.

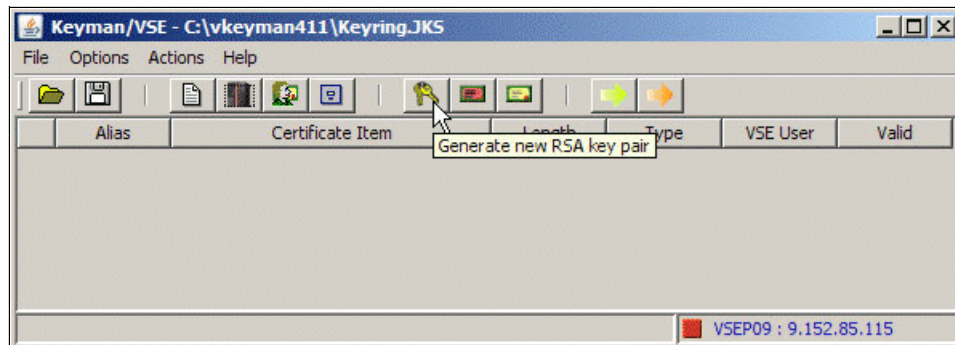


Figure 11-56 Creating an RSA key pair



Specify the key length of the RSA key pair as shown in Figure 11-57. A PCIXCC or Crypto Express2 card is required for processing 2048-bit keys on z/VSE.



Figure 11-57 Specify alias and key length of the new RSA key

Create a self-signed root certificate, as shown in Figure 11-58.

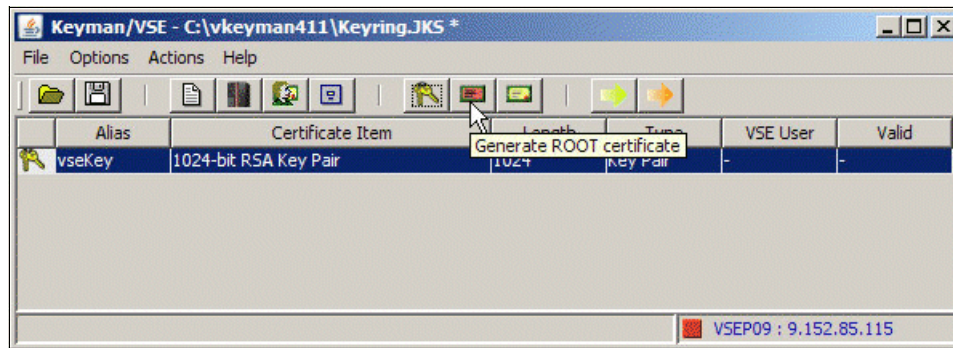


Figure 11-58 Creating a ROOT certificate

Enter the personal information for the ROOT certificate, as shown in Figure 11-59.

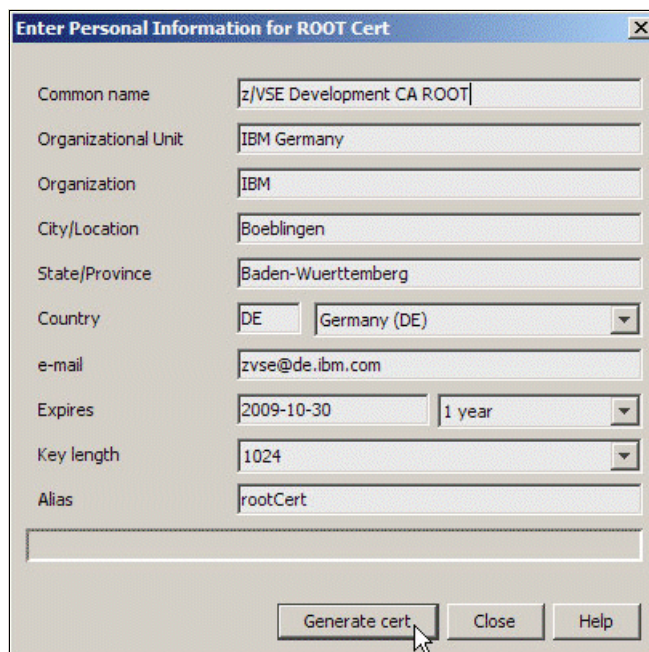


Figure 11-59 Personal information for a ROOT certificate

Click **Generate cert.**

Then, create a certificate request for the z/VSE server certificate, as shown in Figure 11-60.

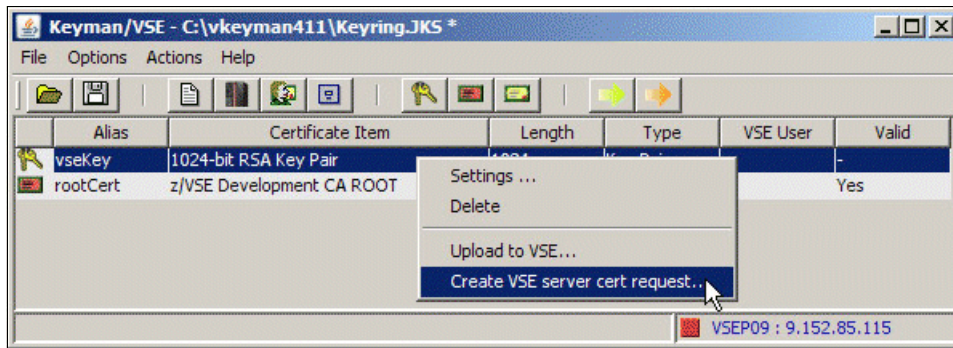


Figure 11-60 Create z/VSE server cert request

The server certificate is created by signing the certificate request with your root certificate. In the window that is shown in Figure 11-61, enter information to identify the z/VSE server certificate.



Figure 11-61 Generate certificate request

Click **Generate** to create the certificate request.

Copy the certificate request to the clipboard. As shown in Figure 11-62, right-click the certificate request and select **Copy to clipboard**.

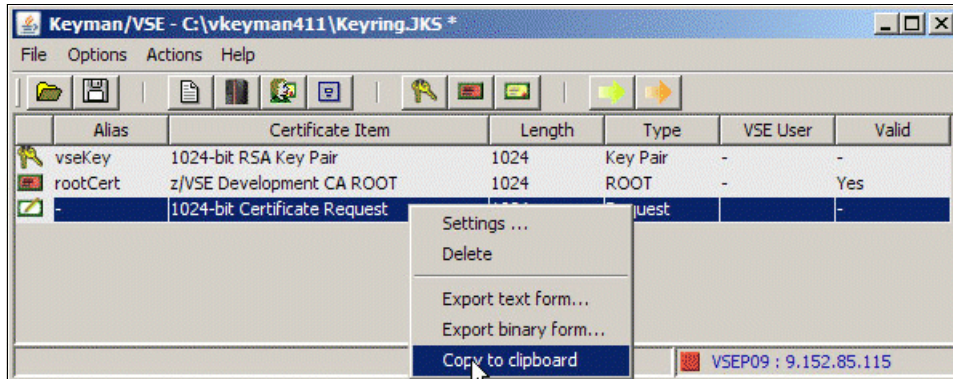


Figure 11-62 Copy certificate request to clipboard

Right-click the ROOT certificate and select **Sign certificate request**, as shown in Figure 11-63.

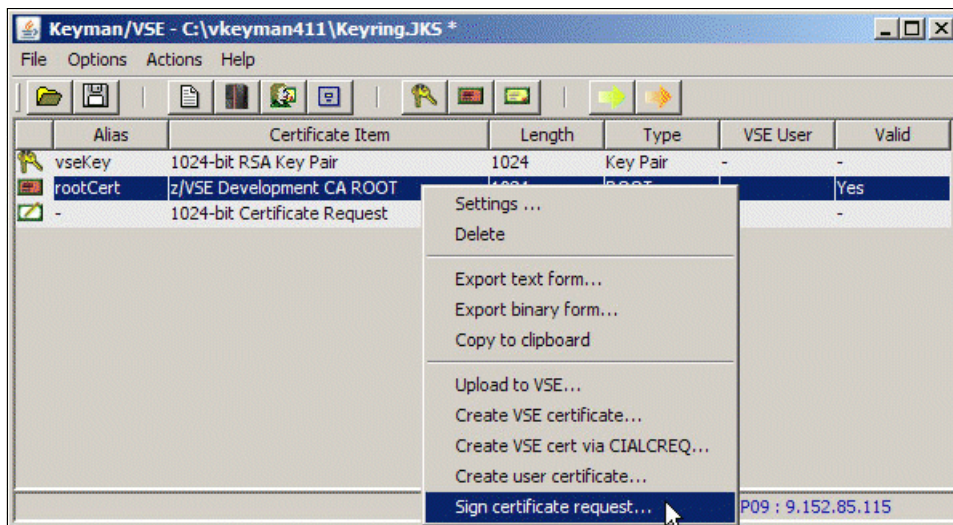


Figure 11-63 Sign the certificate request



Then, paste the clipboard content into the text area of the next window, as shown in Figure 11-64.

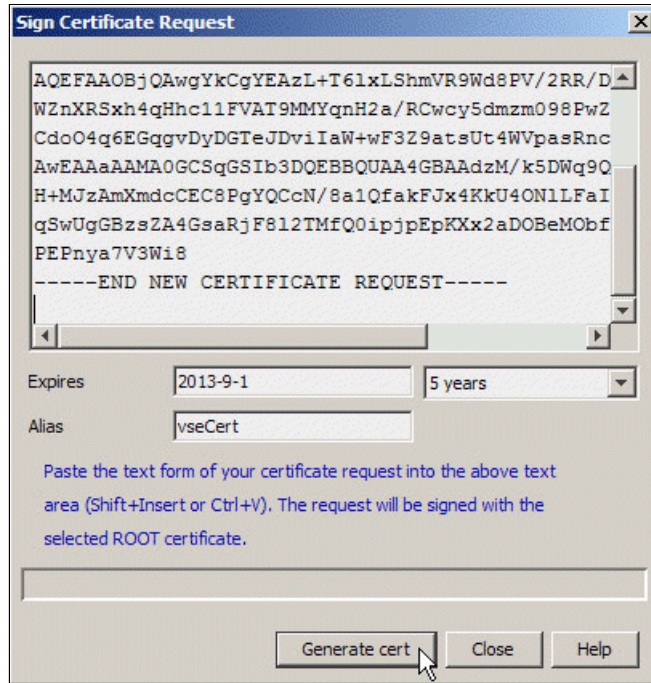


Figure 11-64 Generate certificate

Click **Generate cert** to create the z/VSE server certificate. The certificate request can be deleted now.

You should now have the three items in Keyman/VSE, as shown in Figure 11-65.

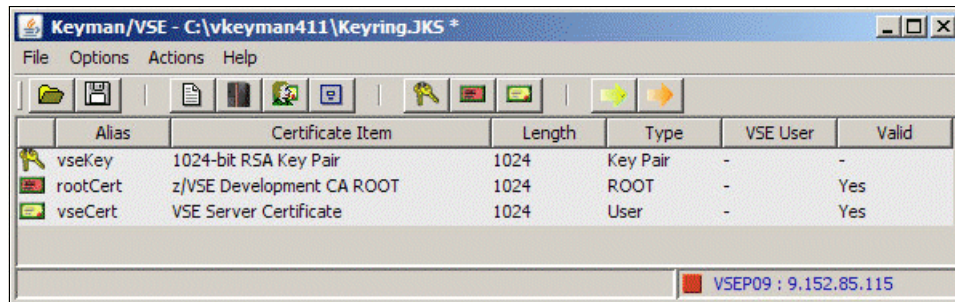


Figure 11-65 z/VSE keyring items

The next step is to upload these items to z/VSE.

### Uploading the certificate items to z/VSE

The three Keyman/VSE items are uploaded to the z/VSE sublibrary CRYPTO.KEYRING as three members of types CERT, PRVK, and ROOT with the name of the z/VSE keyring.

In the Keyman/VSE main window, open the VSE Host - Properties window (as shown in Figure 11-67 on page 383) and enter a name for the z/VSE library members that are uploaded. This name is the name of the z/VSE keyring.



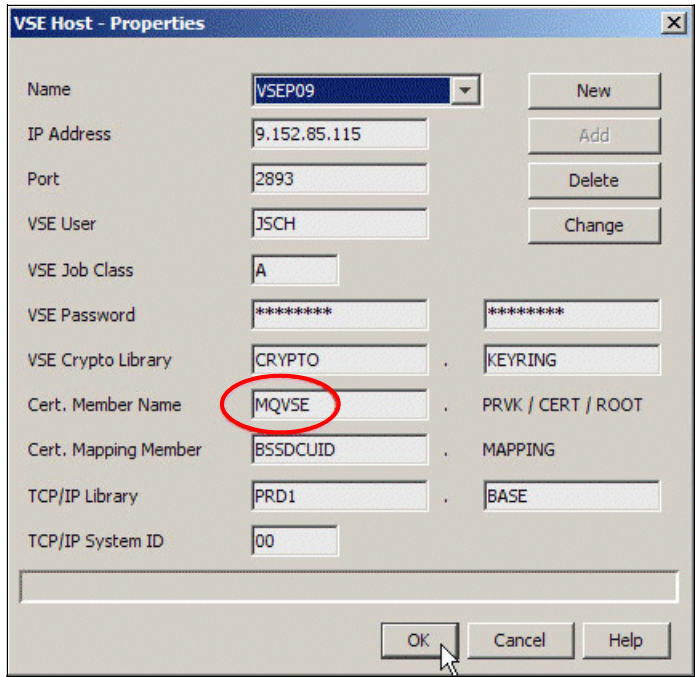


Figure 11-66 z/VSE host properties

Click **OK** to return to the Keyman/VSE main window.

Now, upload all three items to z/VSE by right-clicking an item and selecting **Upload to VSE**, as shown in Figure 11-67.

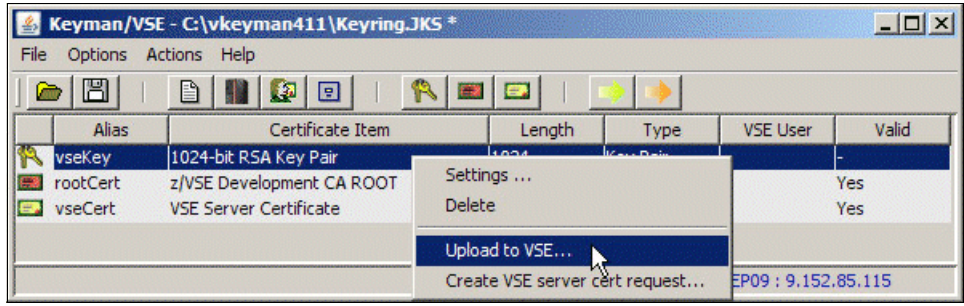


Figure 11-67 Upload to VSE

Repeat this step for all three items in the list.

Finally, three library members are cataloged in the z/VSE keyring library, as shown in Figure 11-68. Do not close the Keyman/VSE tool for now.

```

DIRECTORY DISPLAY      SUBLIBRARY=CRYPTO.KEYRING      DATE: 2008-11-05
                                                                    TIME: 11:38
-----
 M E M B E R      CREATION   LAST      BYTES   LIBR  CONT  SVA  A-  R-
 NAME            TYPE       DATE      UPDATE  RECS  BLKS  STOR ELIG MODE
-----
MQVSE           CERT      08-11-05  - -     707 B    1 YES  - - -
MQVSE           PRVK      08-11-05  - -    2048 B    3 YES  - - -
MQVSE           ROOT      08-11-05  - -     710 B    1 YES  - - -
L113I RETURN CODE OF LISTDIR IS 0
  
```

Figure 11-68 Crypto.Keyring contents

## Specifying the correct certificate label for MQ

After uploading the keyring members to z/VSE, the two certificates must be stored in a local keyring file. This file is imported into the MQ key database later.

Before storing the two certificates in the local keyring file, you must change the certificate label of the z/VSE certificate. WebSphere MQ requires the following naming convention (in lowercase, concatenated with the MQ queue manager name, also in lowercase):

`ibmwebspheremq`

In our example, the label is `ibmwebspheremqmq_b13xghhe`.

In the Keyman/VSE tool, select **Settings**, as shown in Figure 11-69.

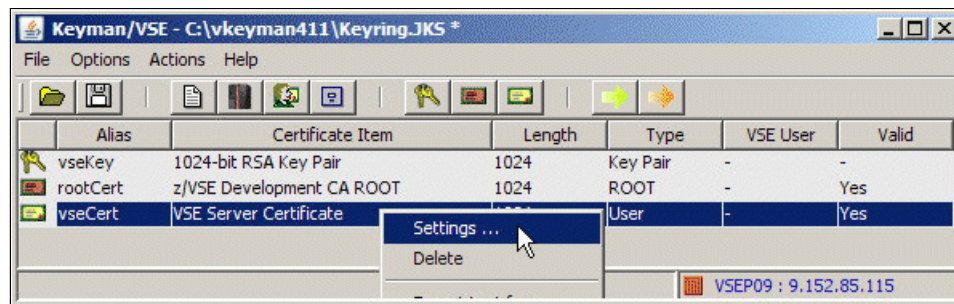


Figure 11-69 Open Settings dialog

The settings window of the z/VSE certificate opens, as shown in Figure 11-70. Change the certificate alias name (label) to the applicable string in your installation.

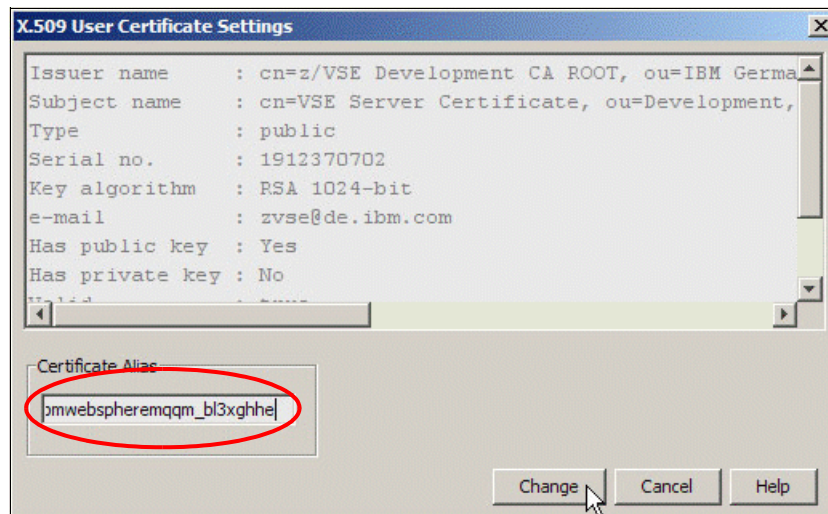


Figure 11-70 Specify certificate alias

Click **Change** to continue.

You can now delete the RSA key, and then save the keyring file, as shown in Figure 11-71.

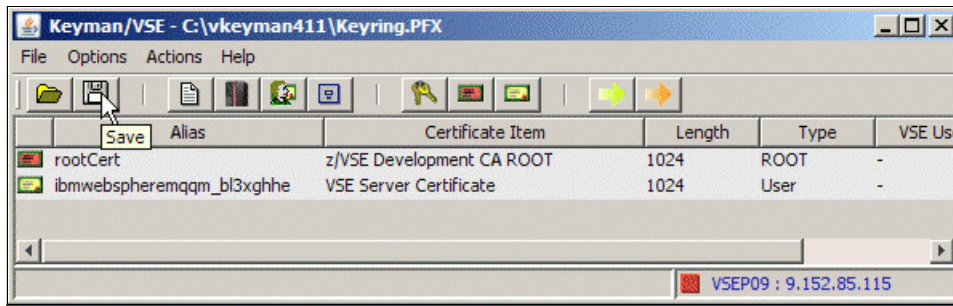


Figure 11-71 Save the keyring file

In the next window, select PFX and enter a keyring file password, as shown in Figure 11-72.

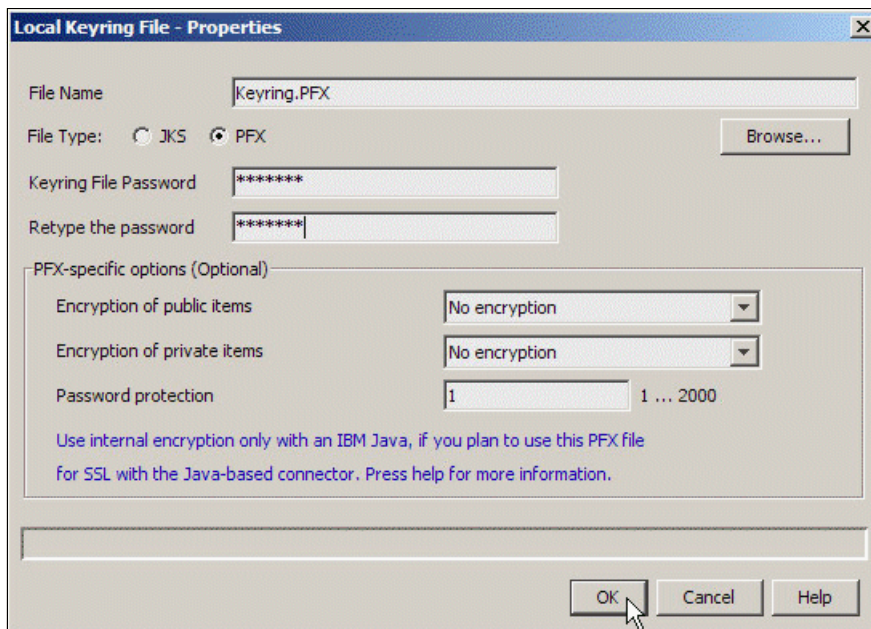


Figure 11-72 Set the properties of the keyring file

Click **OK** to save it.

The process that is used to import this keyring file into the MQ key database is described next.

**Important:** Make sure to import the complete PFX file into the key database so that the private key of the self-signed root certificate is not lost.



## Creating an MQ key database

In the MQ Explorer, select **IBM WebSphere MQ** → **Manage SSL Certificates**, as shown in Figure 11-73.

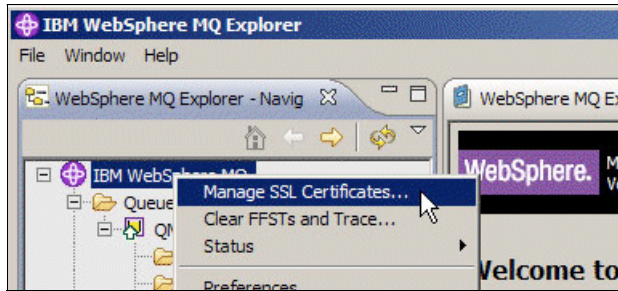


Figure 11-73 Select Manage SSL Certificates

The IBM Key Management GUI opens. In the IBM Key Management GUI, select **Key Database File** → **New**.

Then, enter the file name and location of the database and click **OK**, as shown in Figure 11-74.

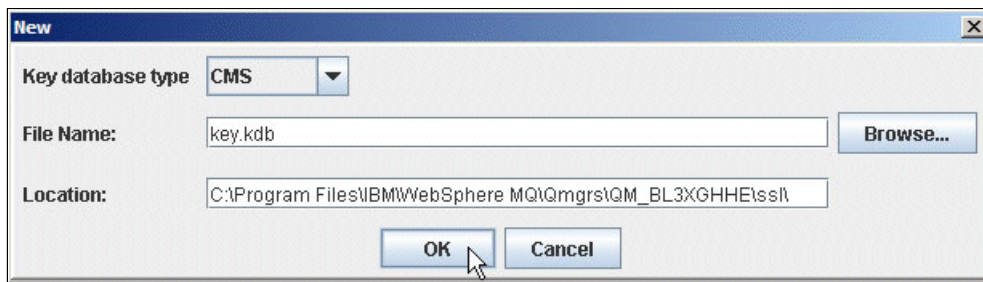


Figure 11-74 Specify file name and location

In the IBM Key Management GUI, select **Key Database File** → **Change Password** to specify the key database password, as shown in Figure 11-75.

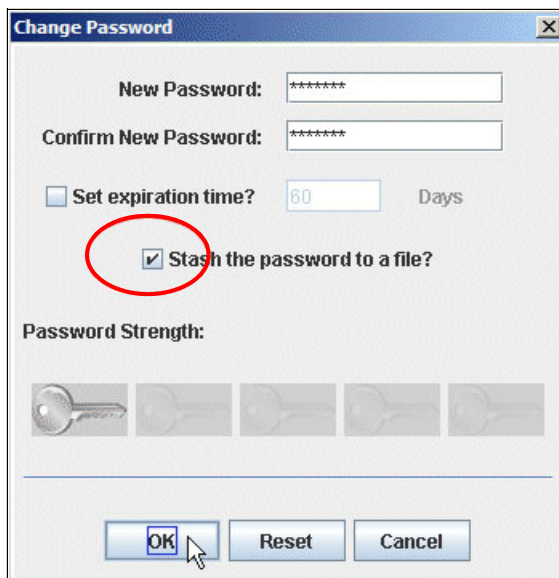


Figure 11-75 Change the password

**Important:** You must create a password *stash* file. When opening an SSL session, MQ must access the key database; therefore, it requires the password. The stash file contains an encrypted copy of the password and is accessed by MQ at runtime. It is in the same directory as the key repository, with the same file name, but with a *.sth* extension.

## Storing the certificate items in the MQ key database

In the IBM Key Management GUI, select **Personal Certificates** from the drop-down list and click **Import**, as shown in Figure 11-76.

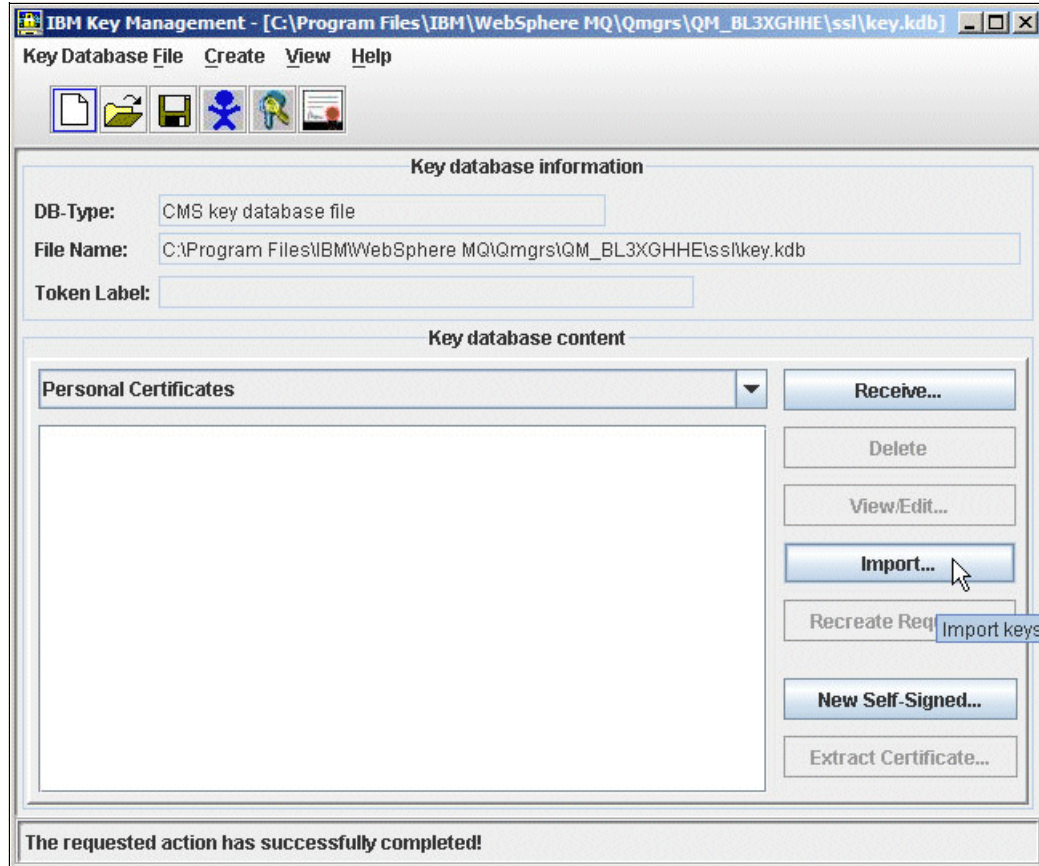


Figure 11-76 IBM Key management

In the Import Key window, select **PKCS12** and browse to the previously saved keyring file, as shown in Figure 11-77. Then, click **OK**.

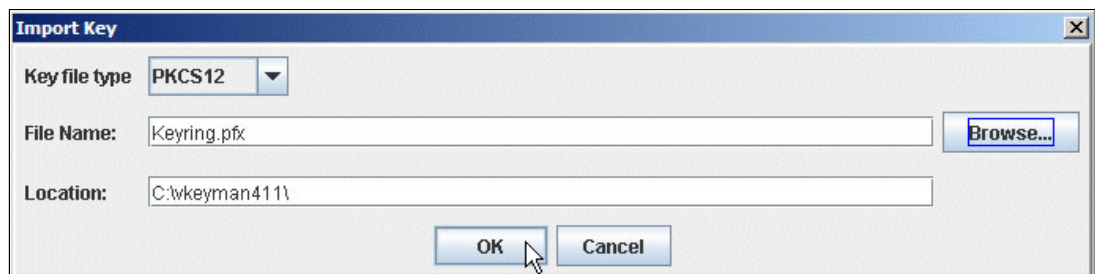


Figure 11-77 Import key

In the Password Prompt window, enter your keyring file password and click **OK** (see Figure 11-78).

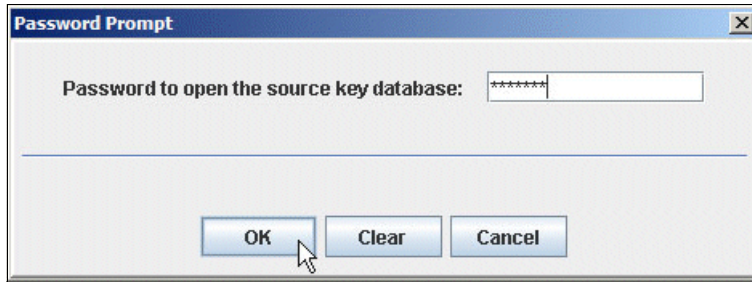


Figure 11-78 Password of source key database

Figure 11-79 shows the Change Labels window. It includes the name of the two certificates that were created with Keyman/VSE. For the z/VSE certificate, you see that the label `ibmwebspheremqm` is concatenated with your queue manager name. Click **OK**.

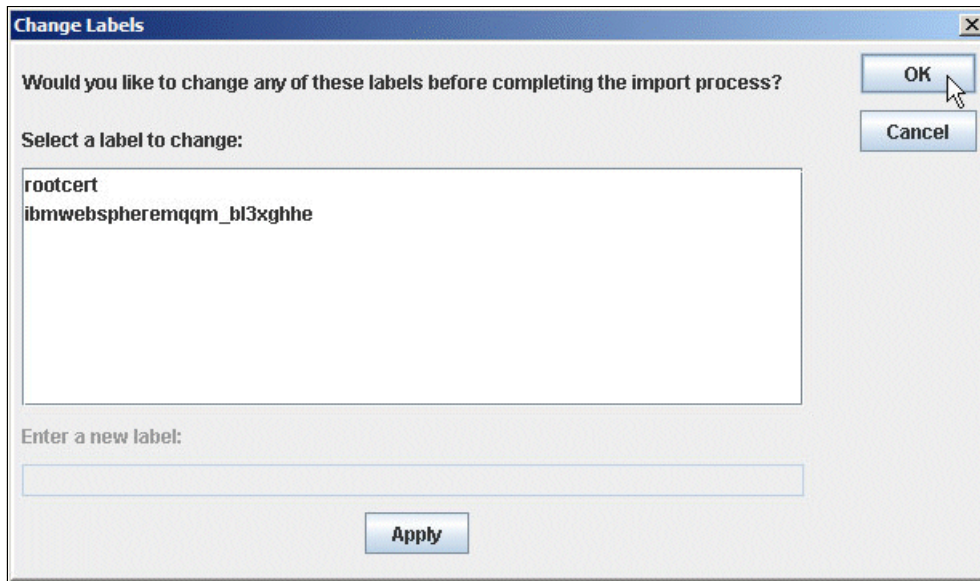


Figure 11-79 List of labels

The IBM Key Management window opens, as shown in Figure 11-80.

The two certificates are now imported into the MQ key database.

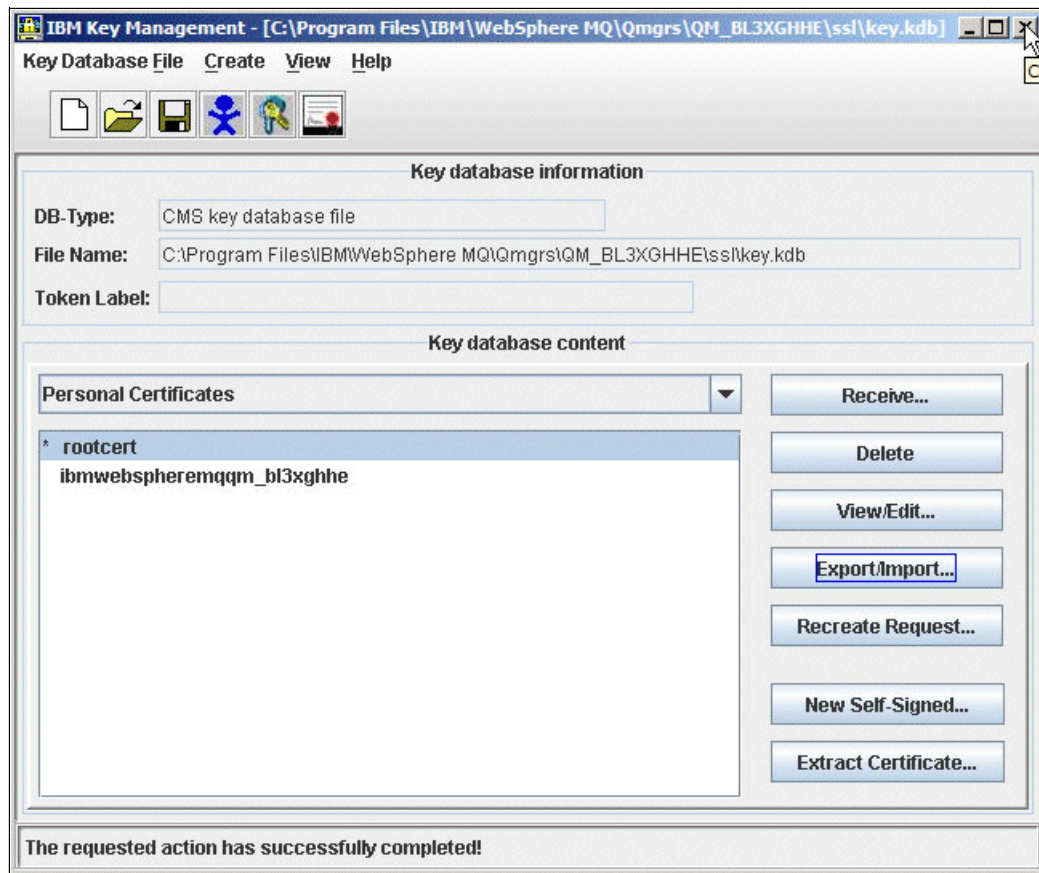


Figure 11-80 IBM Key management

Now you can close the IBM Key Management GUI.

## 11.4.2 SSL configuration on z/VSE

You must configure SSL in the queue manager definition and in the MQ channels. MQ queues are not affected by SSL.

### Configuring the queue manager for SSL

Start transaction MQMT, path 1.1 and enter the z/VSE keyring library and the name of the keyring members (see Figure 11-81 on page 390).



```

12/16/2008      IBM WebSphere MQ for z/VSE Version 3.0.0      DBDCCICS
11:33:40      Global System Definition                      CIC1
MQWMSYS      Communications Settings                          A000

TCP/IP settings      Batch Interface settings
TCP/IP listener port : 01414      Batch Int. identifier: MQBISERV
Licensed clients . . : 00000      Batch Int. auto-start: Y
Adopt MCA . . . . . : N      Channel Auto-Definition
Adopt MCA Check . . : N      Auto-definition . . : Y
      Auto-definition exit :

SSL parameters
Key-ring sublibrary : CRYPTO.KEYRING
Key-ring member . . : MQVSE

PCF parameters
System command queue : SYSTEM.ADMIN.COMMAND.QUEUE
System reply queue . : SYSTEM.ADMIN.REPLY.QUEUE
Cmd Server auto-start: N
Cmd Server convert . : N
Cmd Server DLQ store : N

PF2=Queue Manager details  PF3=Quit  PF4/Enter=Read  PF6=Update

```

Figure 11-81 Communication settings

## Checking for available SSL cipher suites

To ensure that SSL is licensed on your z/VSE system, run the **q prod,all** command to the TCP/IP partition to see your current TCP/IP license keys, as shown in Example 11-1<sup>1</sup>.

Example 11-1 TCP/IP product keys

```

100 q prod,all
F7 0097 IPN253I << TCP/IP Product Keys >>
F7 0097 IPN885I CPU ID: 0572AF (057216)
F7 0097 IPN886I Stack..... (IBM)
F7 0097 IPN886I Base..... included in Stack
F7 0097 IPN886I Telnet..... included in Stack
F7 0097 IPN886I FTP..... included in Stack
F7 0097 IPN886I LPR..... included in Stack
F7 0097 IPN886I HTTP..... included in Stack
F7 0097 IPN886I CAF..... Not licensed
F7 0097 IPN886I NFS..... Not licensed
F7 0097 IPN886I SSL..... included in Stack <-- SSL must be included.
F7 0097 IPN886I GPS..... Not licensed
F7 0097 IPN886I SecureFTP... included in Stack
F7 0097 IPN886I SeeVSE..... Not licensed
F7 0097 IPN886I HFS..... Not licensed
F7 0097 IPN886I eMail..... Not licensed
F7 0097 IPN886I AES..... Not licensed

```

Table 11-2 on page 391 lists the combinations of cipher suites on z/VSE and Windows, which worked in the test setup.

**Note:** At the time of this writing, the two AES cipher suites did not work for an unknown reason.

<sup>1</sup> At the time of this writing, the output of the Q PROD,ALL command for AES is misleading. AES is included in the SSL component, but is displayed separately as not licensed. If SSL is licensed, AES also is available.

Table 11-2 Valid combinations of cipher suites on z/VSE and Windows

VSE cipher suite hex code	WebSphere MQ 7.0 cipher suite name	Encryption strength
01	NULL_MD5	None
02	NULL_SHA	None
09	DES_SHA_EXPORT	56 bits
0A	TRIPLE_DES_SHA_US	168 bits

We did not find any matching cipher suite in the WebSphere MQ Explorer for the VSE cipher specs 08 and 62. Also, the names of the cipher suites in WebSphere MQ are different from the names that are used on z/VSE. The combinations that must be used in the WebSphere MQ 7.0 Explorer are listed in Table 11-2.

### Configuring the channels for SSL

To define the SSL parameters for the sender channel, press PF10 at the Maintain Channel Record screen (MQMT option 1.3) to open the next panel, as shown in Figure 11-82.

```

11/04/2008          IBM WebSphere MQ for z/VSE Version 3.0.0          DBDCCICS
15:13:04           Channel SSL Parameters                          CICI
MQWMCHN                                                    A000

      Channel Name: VSE.TO.WIN          Type: S

      SSL Cipher Specification. : 0A      (2 character code)
      SSL Client Authentication : 0      (Required or Optional)

      SSL Peer Attributes:
      >
      >
      >
      >
      >
      >

      SSL channel parameters displayed.
      F2=Return PF3=Quit PF4=Read F6=Update
  
```

Figure 11-82 Channel SSL parameters

Press PF6 to update the channel definition.

The definition for the receiver channel is identical. Now, restart MQ on z/VSE.

### 11.4.3 SSL configuration on Windows

At this point, we assume that the MQ key database is set up as described in “Creating an MQ key database” on page 386.

#### Configuring the queue manager for SSL

Select your Windows queue manager and display its properties. As shown in Figure 11-83 on page 392, select **SSL**. Enter the full path to the key database, including the name of the key database file without .kdb as the ending.

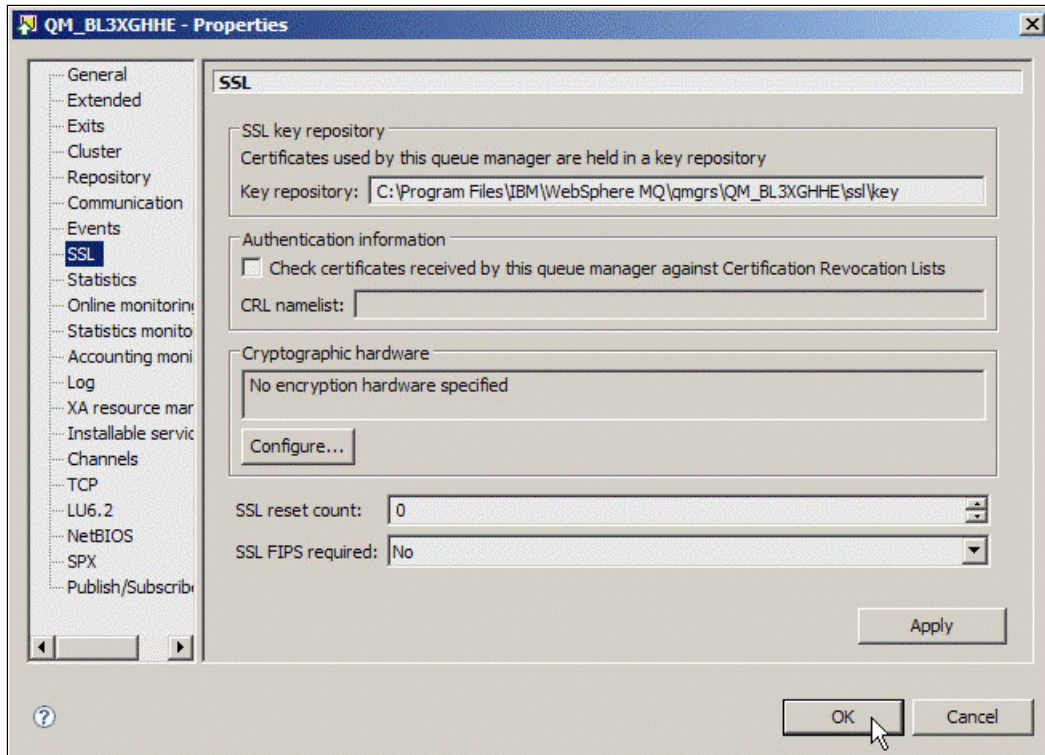


Figure 11-83 Queue manager properties

## Configuring the channels for SSL

Display the properties of the sender channel and select **SSL**, as shown in Figure 11-84.

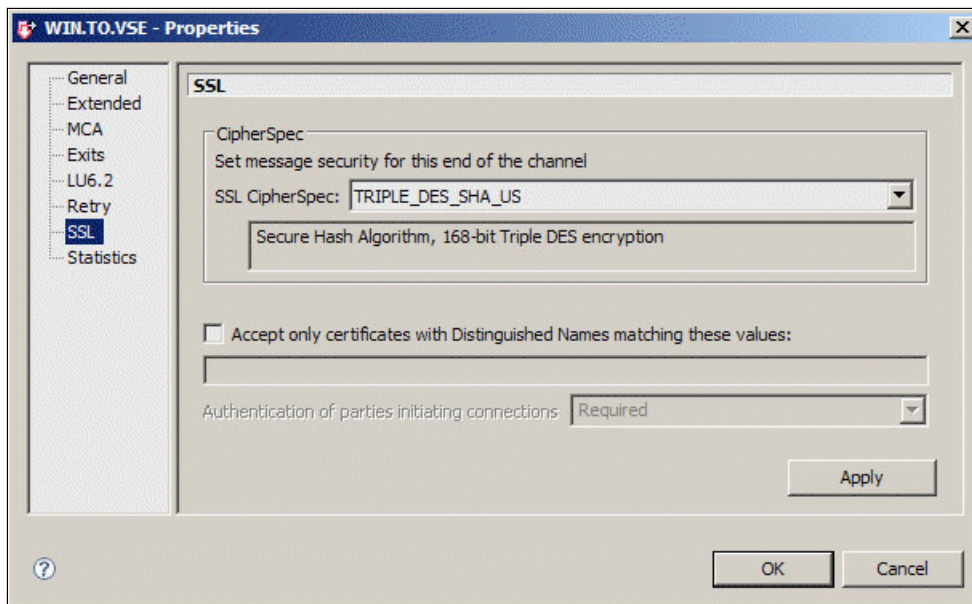


Figure 11-84 Select SSL cipher specification

Select a cipher suite that is also supported by z/VSE and click **OK**. Valid combinations of cipher suites are listed in Table 11-2 on page 391. Then, restart the channel.

The definition for the receiver channel is identical.

## 11.5 Implementing SSL client authentication

SSL client authentication is always configured for the receiver channel, which is the SSL server.

### 11.5.1 Configuring for client authentication on z/VSE

When z/VSE is the receiver (server), SSL client authentication is defined for the receiver channel by changing parameter SSL Client Authentication to R (required). In Windows, the sender channel can be left unchanged.

SSL client authentication means that the client authenticates by sending a client certificate to the SSL server. In this case, the client certificate is given by the user certificate with label `ibmwebspheremqmqm_b13xghe` in the key database. No other setup is necessary.

### 11.5.2 Configuring for client authentication on Windows

When Windows is the receiver (server), SSL client authentication is defined for the receiver channel by selecting **Required** from the drop-down list box named “Authentication of parties initiating connections.” On z/VSE, the sender channel can be left unchanged.

In this case, the client certificate is given by the CERT member in the z/VSE keyring library. No other setup is necessary.

## 11.6 Using SSL peer attributes

When creating a certificate, you must specify personal information, which becomes part of the certificate as the *subject name*. When signing the certificate, the name of the signer becomes part of the certificate as the *issuer name*. These names are called *Distinguished Names* and are strings that consist of a series of keyword-value pairs.

The following keywords are supported:

<b>CN</b>	Common name
<b>C</b>	Country
<b>ST</b>	State or province
<b>L</b>	Locality
<b>O</b>	Organization
<b>OU</b>	Organization Unit
<b>SERIAL</b>	Serial number

The SSL Peer Attributes field in the channel definition is a 256-character case-sensitive field that can be used to ensure that a remote partner’s certificate contains identifiable attributes. This process requires that the remote partner provides a certificate during SSL initial negotiation. If the remote partner fails to provide a certificate, any check against the SSL Peer Attributes field fails, and the channel is ended.

The SSL Peer Attributes field expects a value (if any) in the following form, where key is one of the supported keywords from supported keywords list:

`key=value,key=value, ...`

For more information about specifying peer attributes, including the use of wildcards and white space characters, see *WebSphere MQ for z/VSE System Management Guide*, GC34-6981.

You can display the subject name of your certificates through Keyman/VSE or by using the IBM Key Management tool.

In Keyman/VSE, double-click a certificate to display its properties. You can directly view the certificates on z/VSE by clicking the **Show keyring library** toolbar button, as shown in Figure 11-85.



Figure 11-85 Show keyring library

In the window that is shown in Figure 11-86, double-click a certificate to view its properties.

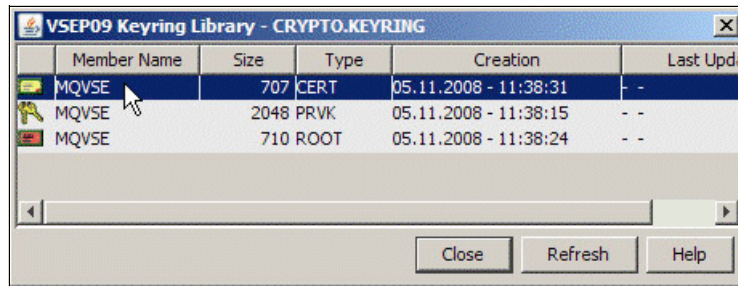


Figure 11-86 Keyring library

Figure 11-87 shows the certificate properties.

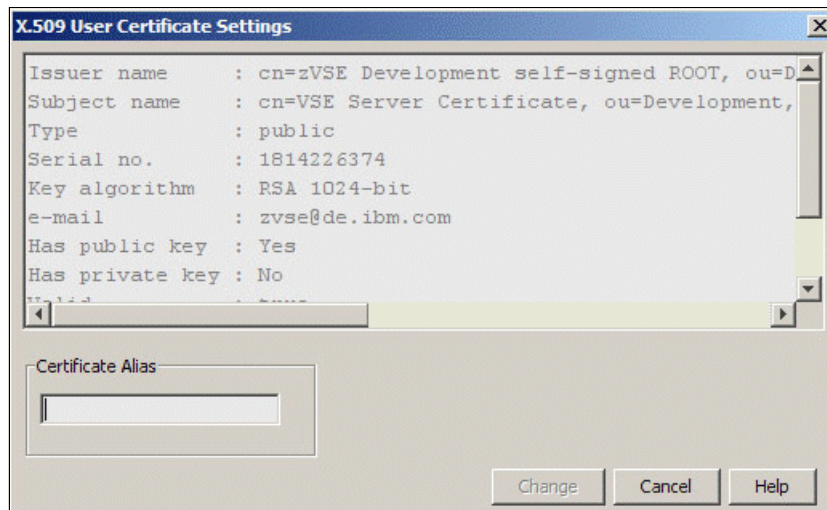


Figure 11-87 User certificate settings

The subject name string shows the attributes that can be specified on the z/VSE side as peer attributes. The next examples show how to specify peer attributes.

### 11.6.1 Example 1: Specifying matching peer attributes

In this example, we show how to specify several peer attributes that match with the partner's client certificate.

In z/VSE, display the properties of the receiver channel and add the peer attributes, as shown in Figure 11-88. You must stop the channel before you specify any peer attributes.

```
12/16/2008          IBM WebSphere MQ for z/VSE Version 3.0.0          DBDCCICS
11:35:17           Channel SSL Parameters                          CICI
MQWMCHN                                                    A000

      Channel Name: WIN.TO.VSE          Type: R

      SSL Cipher Specification. : 0A          (2 character code)
      SSL Client Authentication : R          (Required or Optional)

      SSL Peer Attributes:
      > C=DE,O=IBM                                <
      >                                           <
      >                                           <
      >                                           <
      >                                           <

      Channel record updated OK.
      F2=Return PF3=Quit PF4=Read F6=Update
```

Figure 11-88 Channel SSL parameters

With this setup, the SSL connection can be established.

### 11.6.2 Example 2: Specifying peer attributes that do not match

This example shows what happens when you enter peer attributes on Windows and that do not match with the certificate sent from z/VSE.

Display the properties of the receiver channel and change the SSL peer properties, as shown in Figure 11-89 on page 396.



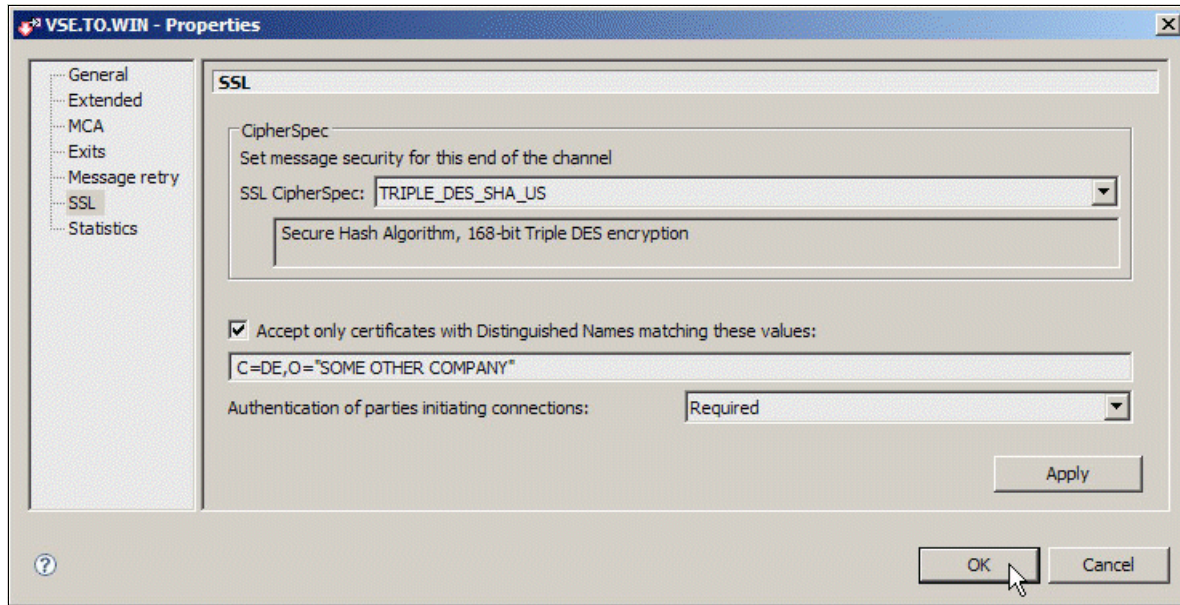


Figure 11-89 Change properties

In this case, the connection cannot be established. The SYSTEM.LOG shows the following message:

```
Receiver responded with error
CHANNEL ID:    VSE.TO.WIN
SSL Peer Name mismatch
NEGOTIATIONS FAILED TO COMPLETE.
```

## 11.7 Configuring a z/VSE queue manager remotely

Configuring a z/VSE queue manager remotely is a new feature with WebSphere MQ for z/VSE V3.0. The process is described in *WebSphere MQ for z/VSE System Management Guide*, GC34-6981.

The following software requirements must be met:

- ▶ WebSphere MQ Explorer V6.0.2.6, or later
- ▶ WebSphere MQ Explorer V7.0.0.1, or later

Updates to the WebSphere MQ Explorer are available at this web page:

<http://www.ibm.com/software/integration/wmq/support/>

### 11.7.1 What you can do remotely

By using the remote configuration functions of the MQ Explorer, you can perform the following tasks:

- ▶ Display and change queue manager attributes
- ▶ Display VSE queues, channels, and namelists
- ▶ Display and change the queue, channel, and namelist properties
- ▶ Put test messages on both sides: Windows and VSE
- ▶ Configure the channels for SSL



However, you cannot start the WebSphere MQ environment on VSE. This process must be done with MQMT on VSE.

The following sections describe the required steps to enable remote configuration.

## 11.7.2 Preparing the z/VSE side for PCF

Remote administration of WebSphere MQ for z/VSE is done by using Programmable Command Format (PCF) messages. Therefore, you must define and start a PCF command server.

For Cmd Server auto-start and Cmd Server convert, specify **Y**, as shown in Figure 11-90.

```
12/16/2008      IBM WebSphere MQ for z/VSE Version 3.0.0      DBDCCICS
12:10:45        Global System Definition                  CIC1
MQWMSYS         Communications Settings                    A000

TCP/IP settings                                Batch Interface settings
TCP/IP listener port : 01414                    Batch Int. identifier: MQBISERV
Licensed clients . . : 00000                    Batch Int. auto-start: Y
Adopt MCA . . . . . : N                          Channel Auto-Definition
Adopt MCA Check . . : N                          Auto-definition . . : N
                                                    Auto-definition exit :

SSL parameters
Key-ring sublibrary : CRYPTO.KEYRING
Key-ring member . . : MQVSE

PCF parameters
System command queue : SYSTEM.ADMIN.COMMAND.QUEUE
System reply queue . : SYSTEM.ADMIN.REPLY.QUEUE
Cmd Server auto-start: Y
Cmd Server convert . : Y
Cmd Server DLQ store : N

Record updated OK.
PF2=Queue Manager details  PF3=Quit  PF4/Enter=Read  PF6=Update
```

Figure 11-90 Communication settings

Channel Auto-Definition or an Auto-definition exit is not required for MQ Explorer. By default, MQ Explorer uses the SYSTEM.ADMIN.SVRCONN channel, which is a default definition; therefore, auto-definition is not required. However, the use of auto-definition poses a security risk and an exit should be used in this case.

When restarting WebSphere MQ on z/VSE, the following line should appear in the SYSTEM.LOG or on the console if optional logging to console is enabled:

```
MQI0200I - MQI007000I PCF command server started
```

Now you must define more queues. You can use the MQJINSG.Z sample MQSC job to create these queues, but you must define the system command and reply queues before you can run the job.

## 11.7.3 Defining more queues

Default queues must be defined on the z/VSE side before you can use the MQ Explorer to remotely administer your z/VSE queue manager. The default names for these queues and their default CICS file names are listed in Table 11-3 on page 398.

Table 11-3 Default names

Default queue name	Default CICS file name
SYSTEM.DEFAULT.ALIAS.QUEUE	-
SYSTEM.DEFAULT.LOCAL.QUEUE	MQFDEFS
SYSTEM.DEFAULT.MODEL.QUEUE	MQFDEFS
SYSTEM.DEFAULT.REMOTE.QUEUE	-
SYSTEM.MQEXPLORER.REPLY.MODEL	MQFADMN

For more information about how to define the MQFADMN file, see 11.8.3, “Open of file MQFADMN failed” on page 410.

### 11.7.4 Defining the MQ Explorer reply model queue

The SYSTEM.MQEXPLORER.REPLY.MODEL queue should be defined as a temporary queue so that the queue is deleted when the queue is closed. Defining it as a dynamic queue leads to the VSAM file defined for the model queue filling up. In the test setup, the reply model queue was first defined as dynamic, which caused the problem that is described in 11.8.4, “No space available for PUT request” on page 411.

The size of the VSAM file that holds the MQ Explorer reply queue must be large enough to hold all of the reply messages during the MQ Explorer session. A temporary dynamic queue is not deleted until the queue handle is closed when MQ Explorer disconnects from the z/VSE system. In WebSphere MQ for z/VSE, messages stay in the VSAM file marked as Deleted after they are retrieved from the queue. The message records are physically deleted only when the queue is deleted or reorganized.

To change the queue definition, use the queue maintenance transaction MQMQ or MQMT (options 1.2) and edit the SYSTEM.MQEXPLORER.REPLY.MODEL queue. The Def. type sets the model’s definition type to T or P, as shown in Figure 11-91.

```

02/26/2009          IBM WebSphere MQ for z/VSE Version 3.0.0          DBDCCICS
08:35:48           Queue Extended Definition                       CIC1
MQWMQUE                                                    A000

Object Name: SYSTEM.MQEXPLORER.REPLY.MODEL

General              Maximums              Events
Type . . . : Model  Max. Q depth . . : 00100000  Service int. event: N
File name . . : MQFADMN  Max. msg length: 00002048  Service interval . : 00000000
Usage . . . : N        Max. Q users . . : 00000100  Max. depth event . : N
Shareable . . : Y      Max. gbl locks : 00001000  High depth event . : N
Def. type . . : T      Max. lcl locks : 00001000  High depth limit . : 000
                                                    Low depth event . . : N
                                                    Low depth limit . . : 000

Triggering
Enabled . . : N        Transaction id.:
Type . . . :          Program id . . :
Max. starts: 0001     Terminal id . . :
Restart . . : N       Channel name . . :
User data . . :
:

Requested record displayed.
PF2=Return PF3=Quit PF4/Enter=Read PF5=Add PF6=Update
PF9=List PF10=Queue
  
```

Figure 11-91 Temporary queue definition

The definition of the initial setup also is still shown in Figure 11-91.

### 11.7.5 Defining a server-connection channel

In this setup, we defined a server-connection channel, which by default is named SYSTEM.ADMIN.SVRCONN, as shown in Figure 11-92. However, the MQ Explorer can be configured to use any channel name that should map to an SVRCONN channel on z/VSE (or you can use channel auto-definition).

```

02/20/2009          IBM WebSphere MQ for z/VSE Version 3.0.0          DBDCCICS
11:57:56            Channel Record          DISPLAY          CIC1
MQWMCHN            A000
Channel : SYSTEM.ADMIN.SVRCONN
Desc. . : MQ Explorer server-connection channel
Protocol: T (L/T)  Type : C (S=Snd/R=Rcv/V=Srv/Q=Req/C=svrConn)  Enabled : Y

Sender/Server
Remote TCP/IP port . . . . . : 00000          Short/Long retry count . . : 000000000
Get retry number . . . . . : 000000000      Short retry interval . . . : 000000000
Get retry delay (secs) . . . : 000000000     Long retry interval . . . : 000000000
Convert msgs(Y/N) . . . . . : N              Batch interval . . . . . : 000000000
Transmission queue name. . . :
TP name. . . . . :

Sender/Receiver/Server/Requester
Connection :
Max Messages per Batch . . : 000001          Message Sequence Wrap . . . : 999999999
Max Message Size . . . . . : 00180000        Dead letter store(Y/N) . . : N
Max Transmission Size . . . : 065535          Split Msg(Y/N) . . . . . : N
Max TCP/IP Wait . . . . . : 000000

Channel record displayed.
F2=Return PF3=Quit PF4=Read PF5=Add PF6=Upd PF9=List PF10=SSL PF11=Ext PF12=Del

```

Figure 11-92 Channel record

The z/VSE side is now ready; therefore, we can add the z/VSE queue manager as remote queue manager in MQ Explorer.

### 11.7.6 Defining a remote queue manager

Before you can define a remote queue manager by using the MQ Explorer, MQ must be started on z/VSE.

To define a remote queue manager, select **Queue Managers** → **Add Remote Queue Manager**, as shown in Figure 11-93.

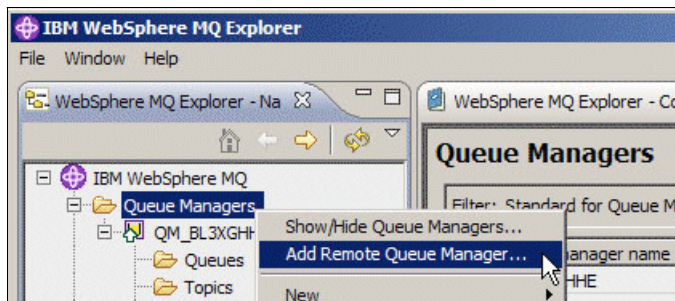


Figure 11-93 Add remote queue manager

In the next window, enter the name of the z/VSE queue manager and click **Next**, as shown in Figure 11-94.

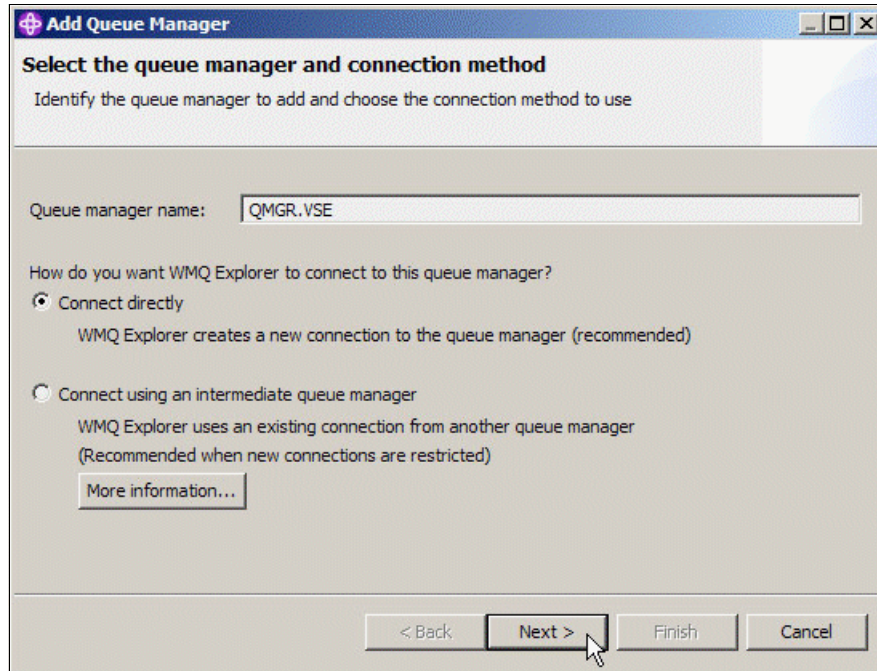


Figure 11-94 Add queue manager

In the window that is shown in Figure 11-95, enter the IP address or host name of your z/VSE system, and then, click **Finish**.

**Add Queue Manager**

**Specify new connection details**  
Provide details of the connection you want WMQ Explorer to set up

Queue manager name:

Specify connection details  
 Use client channel definition table

Host name or IP address:   
Port number:   
Server-connection channel:

Autoreconnect  
 Automatically refresh information shown for this queue manager  
Refresh interval (seconds):

< Back   Next >   **Finish**   Cancel

Figure 11-95 Specify new connection details

The MQ Explorer should now display the z/VSE queue manager with its queues and channels.



Figure 11-96 shows the z/VSE queues.

**Note:** Figure 11-96 on page 402 still shows the MQ Explorer reply model queue defined as permanent/dynamic, which caused some problems. For more information, see 11.7.4, “Defining the MQ Explorer reply model queue” on page 398.

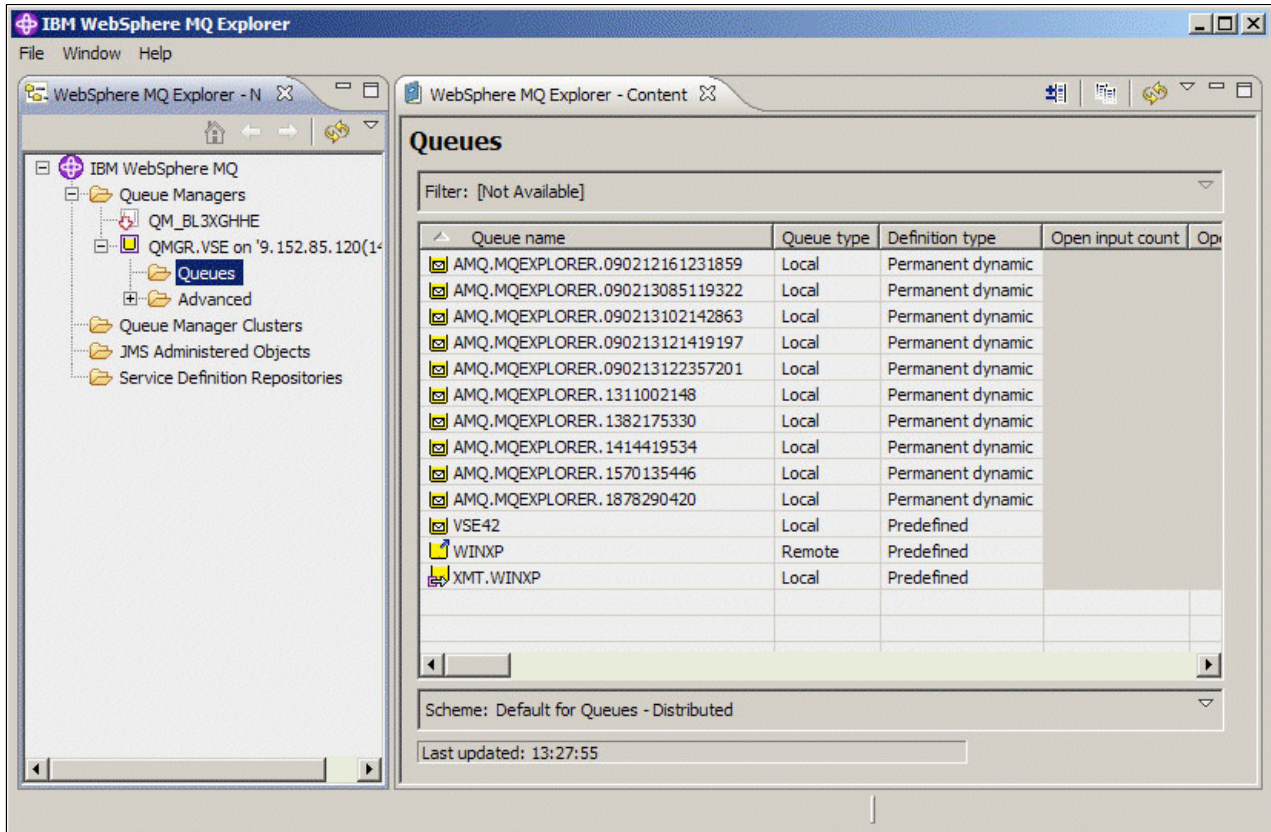


Figure 11-96 List of queues

Figure 11-97 shows the z/VSE channels.

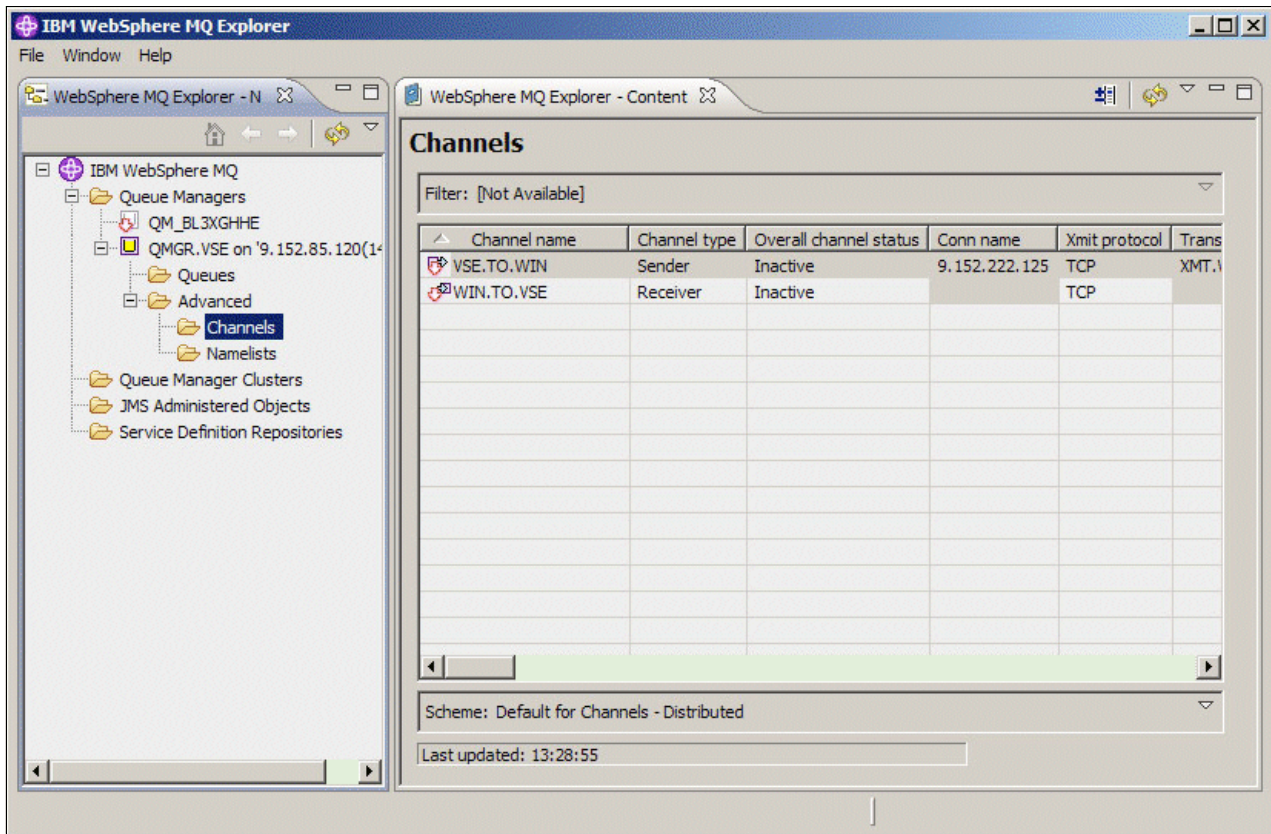


Figure 11-97 List of channels

When MQ on z/VSE is stopped, the remote queue manager is automatically disconnected. After restarting MQ on z/VSE, you can reconnect the remote queue manager. No other actions are possible until the queue manager is connected.



## 11.7.7 Exchanging test messages

You can now exchange test messages between z/VSE and Windows through the MQ Explorer. Figure 11-98 shows an example of how to remotely send a test message from z/VSE to Windows.

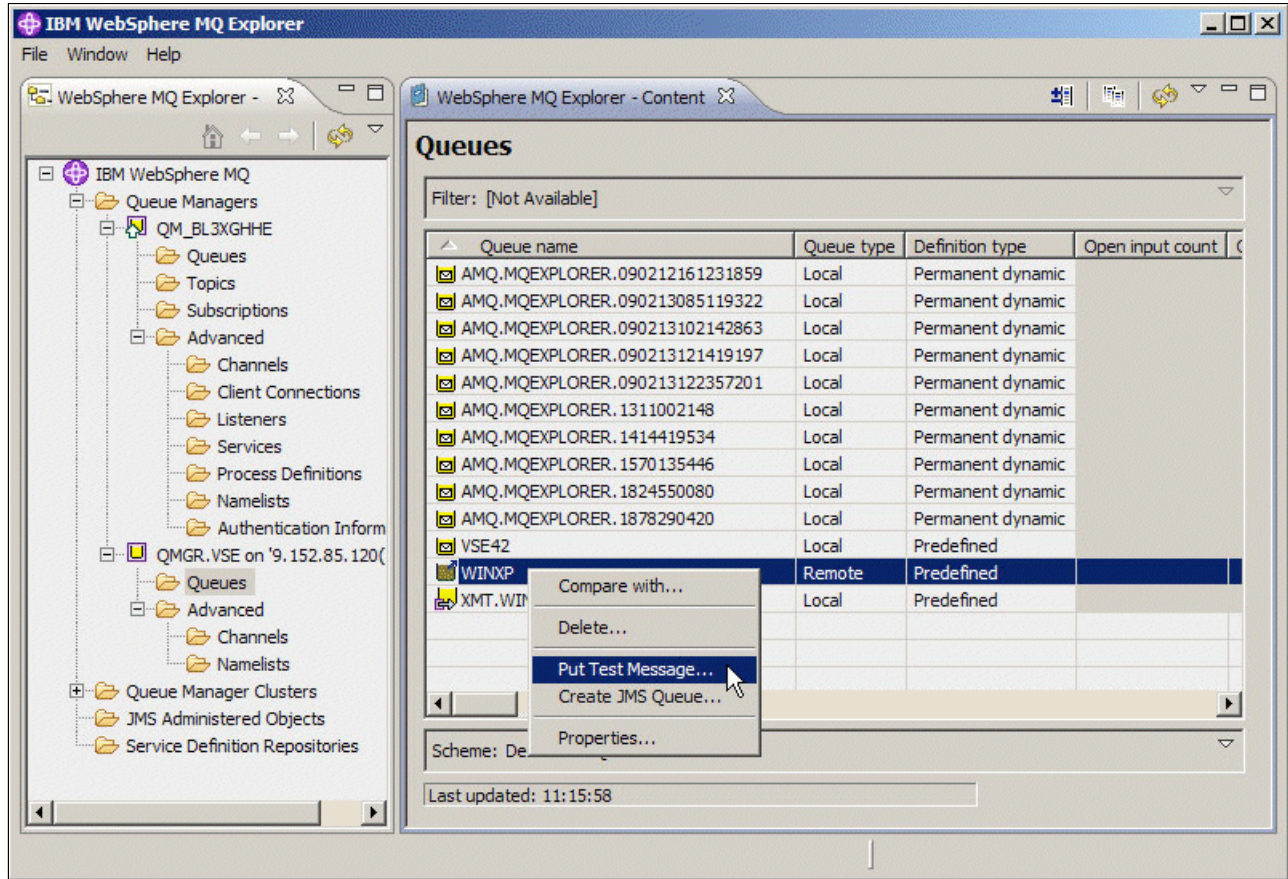


Figure 11-98 Put test message from z/VSE to Windows

Enter some message text in the next window, as shown in Figure 11-99. Click **Put message** to send the message from z/VSE to Windows through remote configuration in the MQ Explorer.

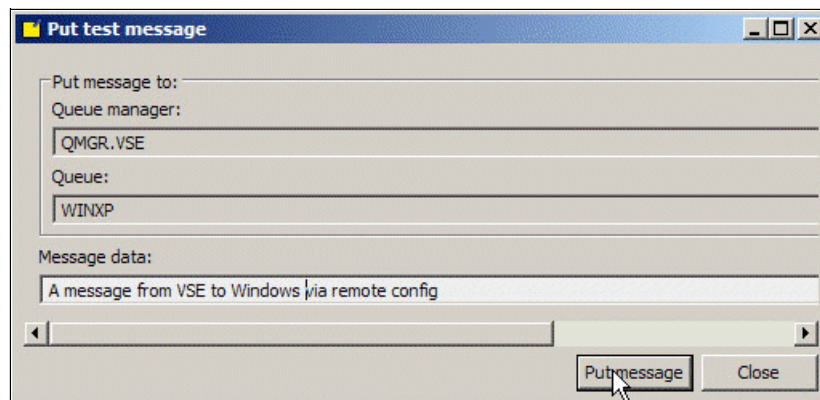


Figure 11-99 Put test message



Figure 11-100 shows how to browse this message in the WINXP queue on Windows.

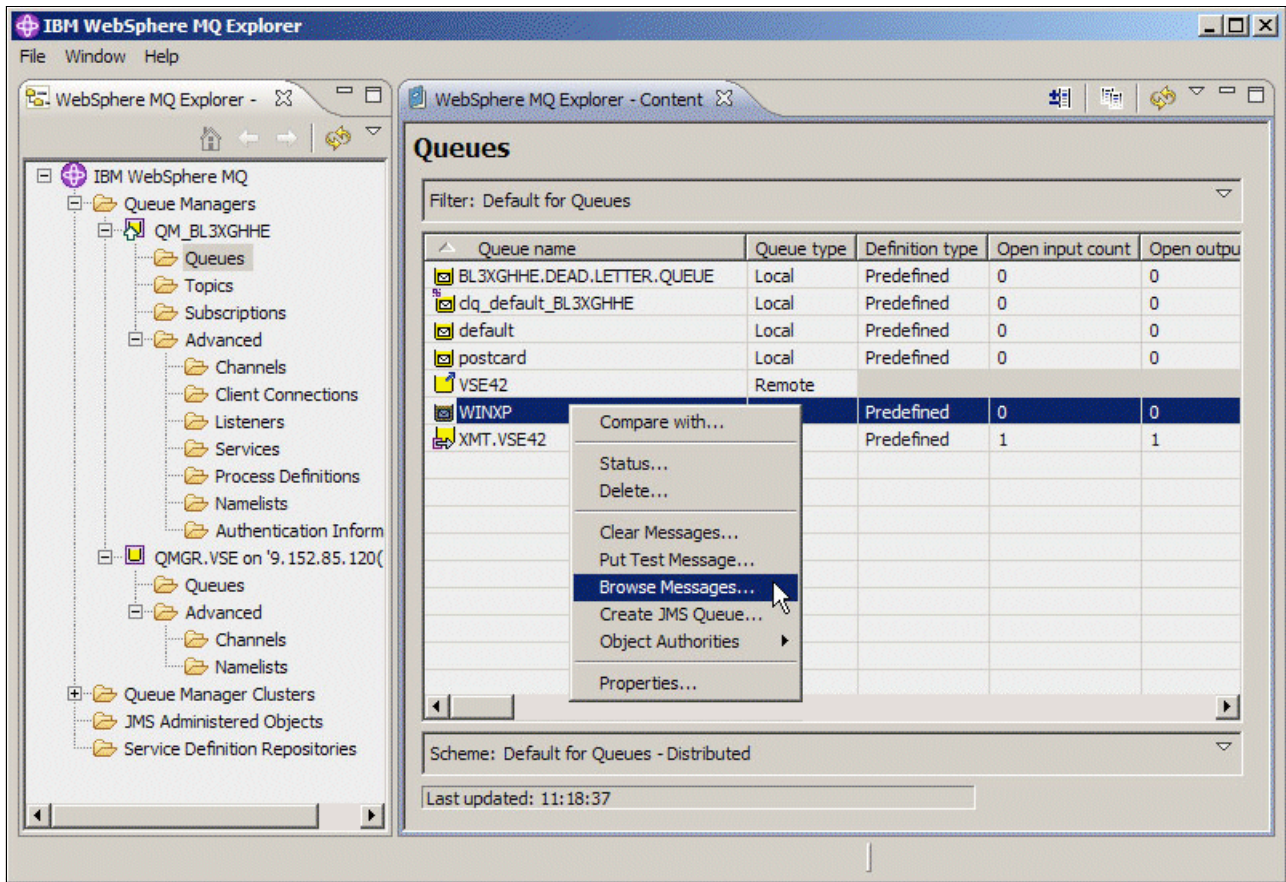


Figure 11-100 Browse messages

Figure 11-101 shows the next window with the message arrived in the WINXP queue on Windows.

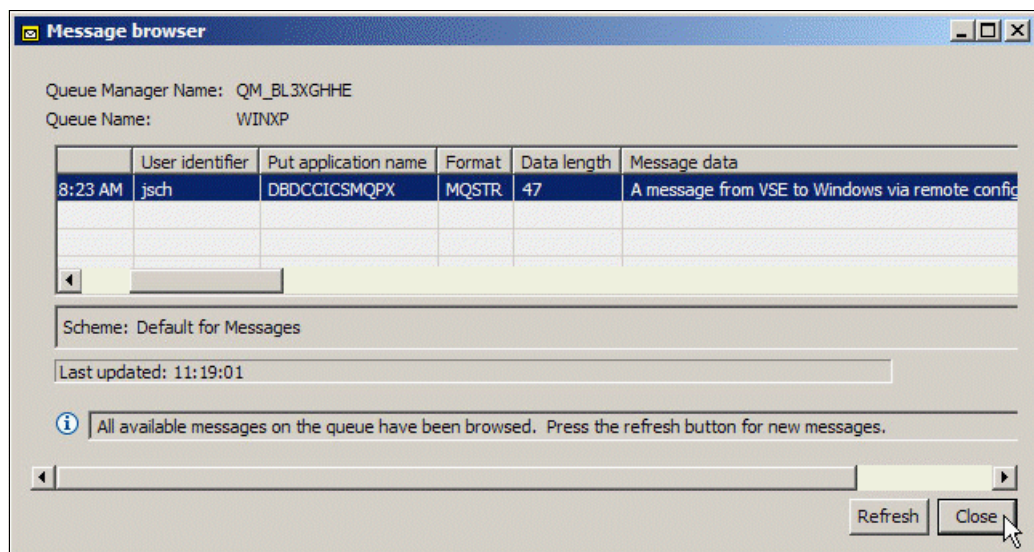


Figure 11-101 Message browser

Click **Close**.

## 11.7.8 Defining SSL

Thus far, we described the problem of finding the right SSL cipher suites so that the defined hex code on z/VSE matches with the defined cipher suite on Windows (see Table 11-2 on page 391).

With the remote configuration through the MQ Explorer, this process is now easy because you define the same cipher suite name on both sides. The MQ Explorer translates the cipher suite name into the hex code that is used on the z/VSE side.

### Defining SSL for the Windows side

First, check again that the Windows queue manager includes the correct definitions.

Open the Windows queue manager properties window, as shown in Figure 11-102. Select **SSL**, verify that the displayed information is correct, and click **OK**.

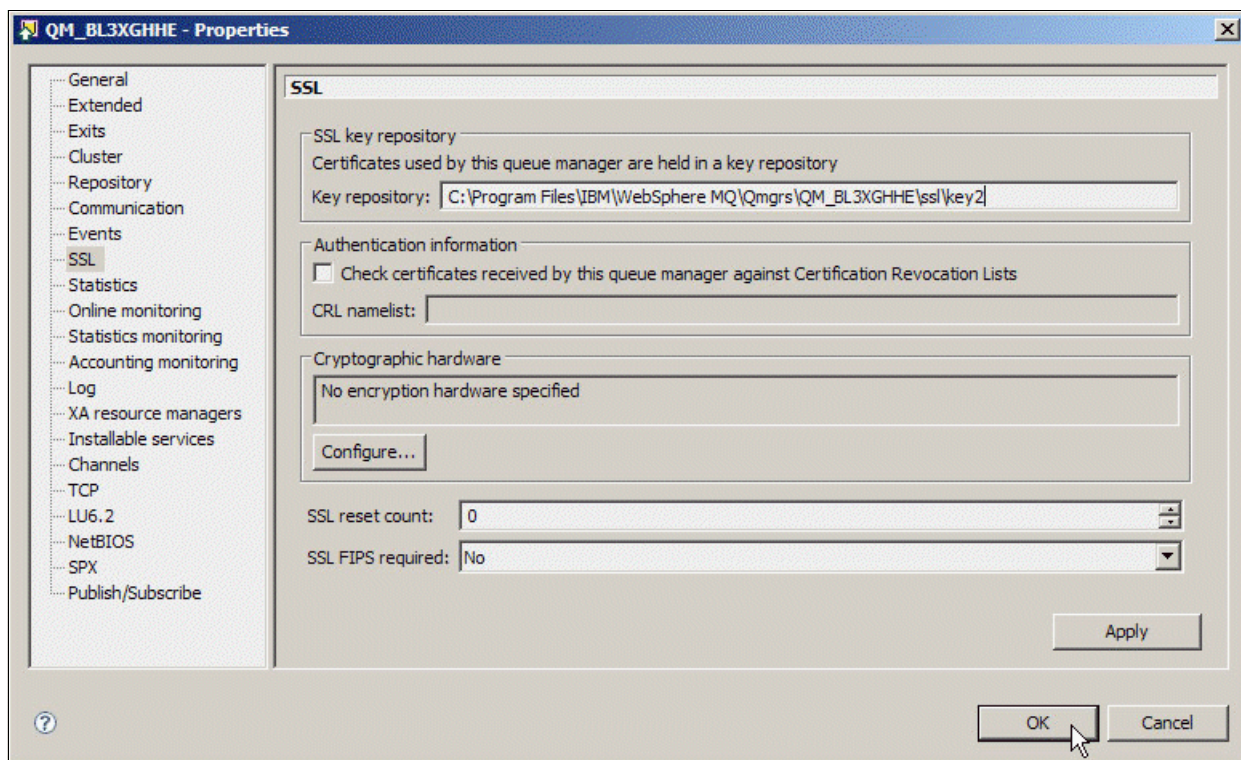


Figure 11-102 Select SSL information

Before continuing with defining SSL for the sender channel, stop the sender channel on Windows.

Then, display the sender channel properties window, as shown in Figure 11-103 on page 407.

Select **SSL**, choose one of the SSL cipher suites that can be used with z/VSE, and then, click **OK**. For more information, see Table 11-2 on page 391.



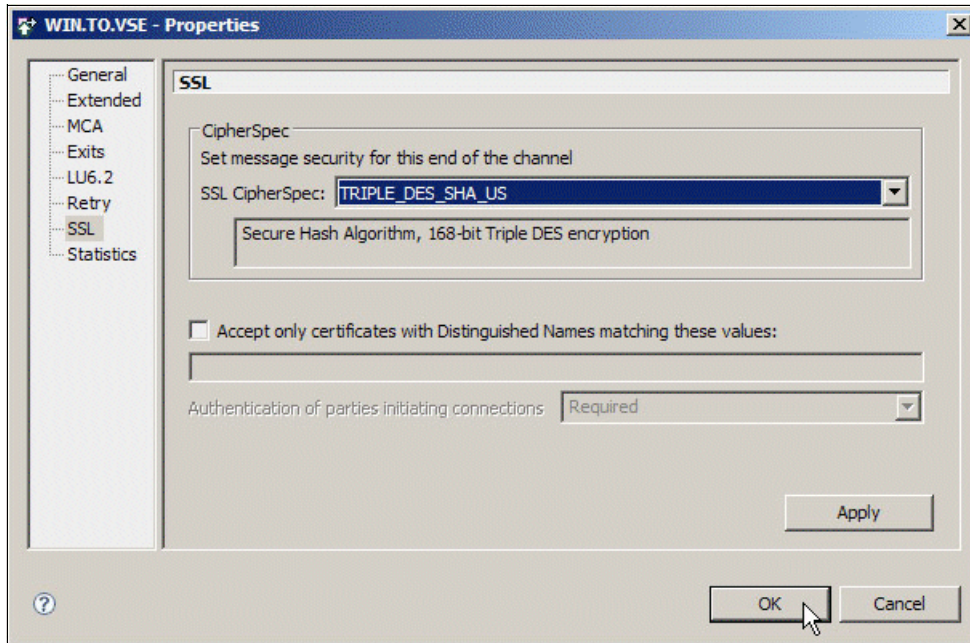


Figure 11-103 SSL properties

Do not restart the sender channel at this point because you first must define SSL for the z/VSE receiver channel.

### Defining SSL for the z/VSE side

Check that the z/VSE queue manager includes the correct definitions. Display the z/VSE queue manager properties, as shown in Figure 11-104, select **SSL**, and click **OK**.

Without remote configuration, this step is done on z/VSE by using the MQMT transaction.

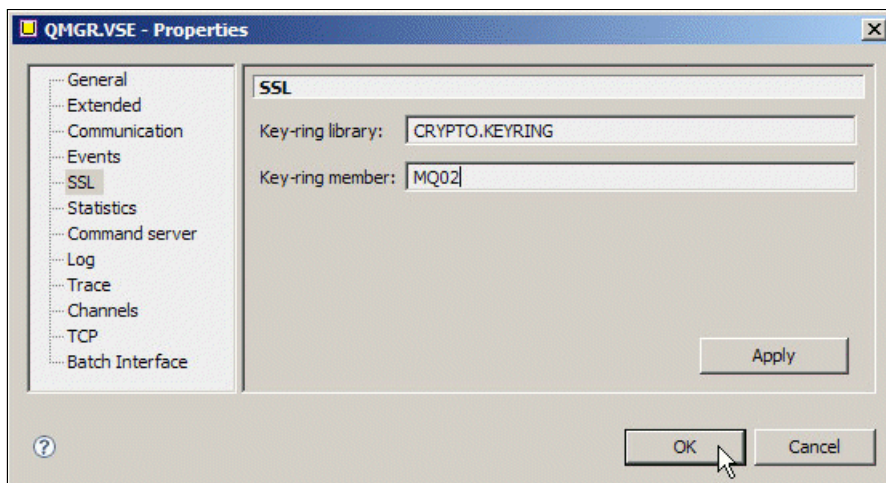


Figure 11-104 Select SSL on z/VSE queue manager properties

Now, define the SSL cipher suite for the z/VSE receiver channel. Display the receiver channel properties window, as shown in Figure 11-105 on page 408. Select **SSL**, and then, click **OK**.

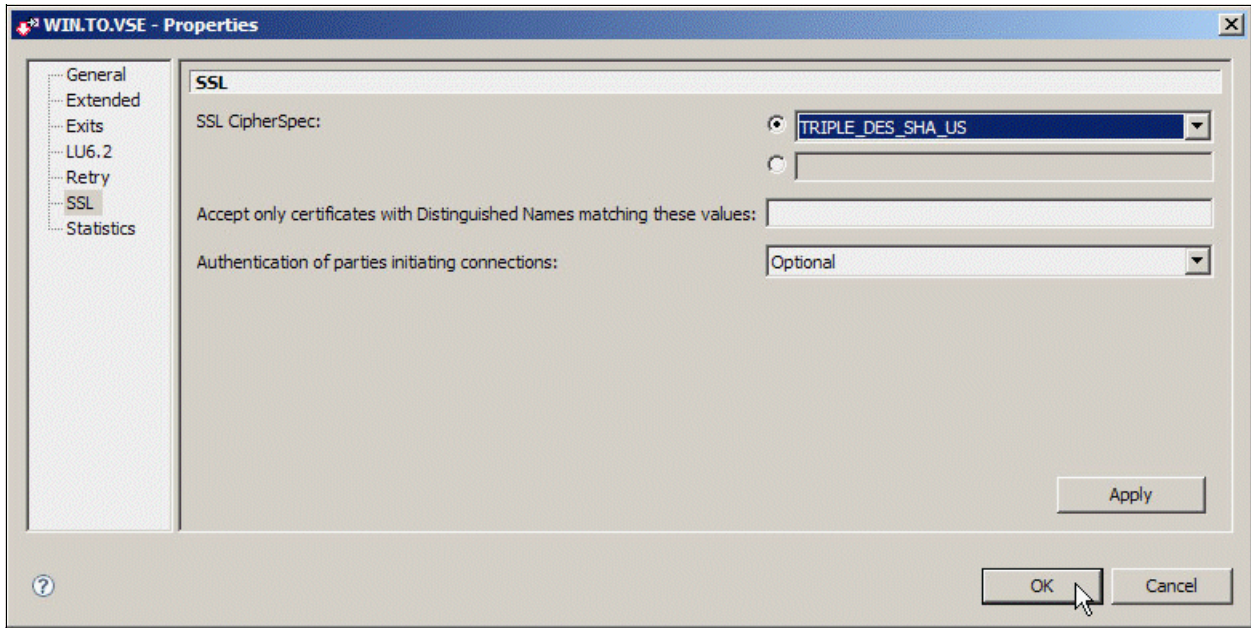


Figure 11-105 z/VSE SSL CipherSpec

**Note:** The AES-based cipher suites that do not work with z/VSE (see Table 11-2 on page 391) are not displayed in the drop-down list for selecting the SSL CipherSpec.

For the sake of completeness, check the z/VSE side to determine how the MQ Explorer made the definitions remotely on the z/VSE side.

Start the MQMT transaction and display the properties of the receiver channel, as shown in Figure 11-106.

```

02/19/2009          IBM WebSphere MQ for z/VSE Version 3.0.0          DBDCCICS
11:43:58           Channel SSL Parameters                          CIC1
MQWMCHN                                                    A000

    Channel Name: WIN.TO.VSE          Type: R

    SSL Cipher Specification. : 0A      (2 character code)
    SSL Client Authentication :        (Required or Optional)

    SSL Peer Attributes:
    >                                     <
    >                                     <
    >                                     <
    >                                     <

    SSL channel parameters displayed.
    F2=Return PF3=Quit PF4=Read F6=Update
  
```

Figure 11-106 Channel SSL parameters

MQ Explorer correctly used the hex code of the cipher suite TRIPLE\_DES\_SHA\_US.

You can now put a test message from Windows to z/VSE by using SSL.

## 11.8 Observations

This section describes several observations that we made during our tests.

### 11.8.1 Message sequence number error

The symptom and possible reason are described.

#### Symptom

When starting the sender channel on Windows, the following messages appear on the z/VSE console:

```
MQI0200I - MQI501028W Channel re-synchronization error
MQI0200I - MQI000003E Channel Message Sequence Number error
```

#### Reason

A mismatch of the message sequence numbers on the sending and receiving ends occurred. In our test setup, this problem is most likely caused by the fact that we exchanged messages between Windows and the first z/VSE system that is running MQSeries V2R1M2.

To resolve the problem, you must reset the message sequence number on the sending end. In the MQ Explorer Window, right-click the sender channel and click **Reset**, as shown in Figure 11-107.

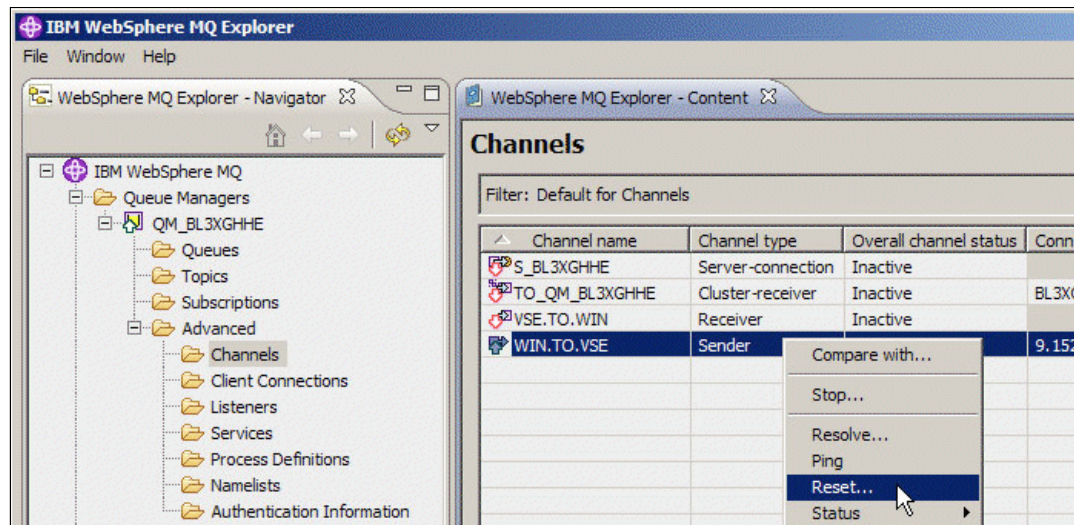


Figure 11-107 Reset the message sequence number on the sender

### 11.8.2 RC=2092 when sending a test message to Windows

The symptom and possible reason are described.

#### Symptom

When sending a test message to Windows through TST2, a return code (RC) of 2092 is issued. TST2 displays the following message:

```
MQ ERROR: LEVEL =INIT , FUNC =OPEN , CC =0002, RC =2092****
```

## Reason

The USAGE parameter of the transmission queue is invalid. This error often happens because the default USAGE is set to N (normal) when a local queue is defined. For the transmission queue, you must change the value to T (transmission). For more information, see “Defining the transmission queue” on page 359.

### 11.8.3 Open of file MQFADMN failed

The symptom and possible reason are described.

## Symptom

The following messages appear on the z/VSE console when you try to define the z/VSE queue manager as a remote queue manager in MQ Explorer:

```
F2 0110 4228I FILE MQFADMN OPEN ERROR X'DC'(220) CAT=MQMCAT
      (OPNRP-20) THE BUFFERS IN BLDVRP TOO SMALL OR CI SIZE TOO LARGE
F2 0109 DFHFC0964 DBDCCICS Open of file MQFADMN failed. VSAM codes - 8502,
      0008,00DC
```

## Reason

The MQFADMN file is defined by job skeleton MQJQUEUE.Z with a maximum record size of 16000, which leads to a CI size of 16384, which is too large. Delete the file and define it again with values that are listed in Example 11-2.

#### *Example 11-2 Job MQJADMN*

---

```
* $$ JOB JNM=MQJADMN,DISP=D,CLASS=A
* $$ LST DISP=H,CLASS=Q,PRI=3
// JOB MQJADMN DEFINE ADMN FILE
// EXEC IDCAMS,SIZE=AUTO
      DELETE (WMQZVSE.MQFADMN) CL NOERASE PURGE -
          CATALOG(MQ.USER.CATALOG)
      SET MAXCC = 0
DEF
      CLUSTER(NAME(WMQZVSE.MQFADMN)
          FILE(MQFADMN)
          VOL(SYSWK2)
          RECORDS(1000 400)
          RECORDSIZE(200 8000)
          INDEXED
          KEYS(56 0 )
          SHR(2))
      DATA(NAME(WMQZVSE.MQFADMN.DATA) CISZ(4096)) -
      INDEX(NAME(WMQZVSE.MQFADMN.INDEX) CISZ(512)) -
          CATALOG(MQ.USER.CATALOG)
/*
/&
* $$ EOJ
```

---

The same problem occurred for file MQFDEFS. Redefining the file as shown in Example 11-2 solved the problem.



## 11.8.4 No space available for PUT request

The symptom and possible reason are described.

### Symptom

The following messages appear repeatedly on the z/VSE console:

```
MQI0200I - MQI102091E No space available for PUT request
MQI0200I - MQI007022W PCF command processor could not send response message
```

### Reason

In our setup, we first defined the MQFADMN file with RECORDS (300 200), which caused this problem. Obviously, the number of records should be increased. Redefining the file with RECORDS (1000 400) solved the problem.

The error occurs when an inbound queue is full. The following methods can be used to check which queue is full:

- ▶ Use transaction MQQM and page through the display looking for FULL inbound status, as shown in Figure 11-108. The VSAM file that must be DELETE/DEFINE also displays.

```
02/20/2009          IBM WebSphere MQ for z/VSE Version 3.0.0          DBDCCICS
08:41:22           Monitor Queues                               CIC1
MQWMMOQ                                                    A000

                      QUEUING SYSTEM IS ACTIVE

S QUEUE              FILE      T INBOUND  OUTBOUND          LRDepth
XMT.WINXP            MQFO001 Y IDLE    IDLE              32              0
AMQ.MQEXPLORER.29884872 MQFADMN N FULL  ACTIVE           0              2
```

Figure 11-108 Monitor queues

- ▶ Browse the SYSTEM.LOG queue. Press PF1 to display last messages and then PF5 to display previous messages until you get the MQI102091E message, as shown in Figure 11-109. You can then use MQMQ to display the queue details to obtain the VSAM file. If the SYSTEM.LOG queue is full, messages are then sent to the CICS log, where the MQI102091E messages might be stored.

```
02/20/2009          IBM WebSphere MQ for z/VSE Version 3.0.0          DBDCCICS
08:44:45           Browse Queue Records                          CIC1
MQWDISP            SYSTEM IS ACTIVE                               A000

Object Name: SYSTEM.LOG
QSN Number : 00000172      LR-          0, LW-          173, DD-MQFLOG
Queue Data Record
Record Status : Written.      PUT date/time : 20090220084421
Message Size : 00000711      GET date/time :

MQI102091E PRG:MQPQUE1 TRN:MQCX TRM:.... TSK:00500 02/20/2009 08:44:21
No space available for PUT request
QUEUE ID : AMQ.MQEXPLORER.29884872

2100-PUT-SET-QSN - QFULL status
9999-NOSPACE
EIBFN: 1206      EIBRCODE: 000000000000 EXEC LINE: 000000
EIBRESP: 00000000 EIBRESP2: 00000000 EIBRSRCE: ABCODE:

Information displayed.
5655-U97 Copyright IBM Corp. 2008. All rights reserved.
Enter=Process PF2=Return PF3=Quit PF4=Next PF5=Prior
PF11=MD PF12=Explain
```

Figure 11-109 Browse queue records





# A

## Security APIs

In this appendix, we describe application programming interfaces (APIs) that can be used under z/VSE to secure your client or server applications.

We distinguish between APIs for host applications and applications for clients on systems other than z/VSE. For these clients, we provide Java-based APIs.

The description of the host APIs includes hardware-based encryption support and a summary of available SSL functions.

This appendix includes the following topics:

- ▶ A.1, “Client-side Java APIs” on page 414
- ▶ A.2, “Host-side APIs” on page 414

## A.1 Client-side Java APIs

z/VSE functions and data can be accessed by various Java-based APIs. The following security and crypto-related Java APIs are available:

- ▶ z/VSE Connector Client
- ▶ Security class library

### A.1.1 z/VSE Connector Client

The z/VSE Connector Client class library provides Java classes for writing their own Java programs by using SSL for connecting to the z/VSE Connector Server. Coding examples can be found in the z/VSE Connector Client installation. The folder `samples/com/ibm/vse/samples` contains the following SSL coding samples:

- ▶ `SSLApiExample.java`
- ▶ `SSLVolExample.java`
- ▶ `SSLConsExample.java`

In addition to Java applications, SSL can be used in any kind of Java program, such as Java applets, Java servlets, and Enterprise JavaBeans (EJBs). For more information about SSL configuration in a WebSphere environment, see *WebSphere V5 for Linux on zSeries Connectivity Handbook*, SG24-7042, and the HTML-based documentation that is provided by the z/VSE Connector Client.

### A.1.2 Security class library

The z/VSE security class library provides Java classes to obtain security-relevant information from a z/VSE system. For more information, see the related Javadoc. The security class library is not an official z/VSE component. It can be downloaded for no charge from the z/VSE home page. It requires the installation of the z/VSE Connector Client.

**Note:** The security class library can be used with only the Basic Security Manager. Currently, other security managers are not supported.

## A.2 Host-side APIs

Different host-side APIs exist for writing security and crypto-related user programs. They are provided by TCP/IP for VSE/ESA, the z/VSE base operating system, or through the System z hardware.

### A.2.1 Using APIs to write your own SSL/TLS applications

In this section, we describe APIs that can be used under z/VSE to secure your client or server application by using SSL/TLS for z/VSE.

### A.2.1.1 Language Environment for VSE C Run-Time Socket API

Figure A-1 shows the basic structure of the elements that are required in your SSL source program.

Client	Server
	<b>gsk_initialize()</b>
<b>gsk_initialize()</b>	socket()
socket()	bind()
connect()	listen()
	accept()
<b>gsk_secure_soc_init()</b>	<b>gsk_secure_soc_init()</b>
*skwrite() .....	*skread()
*skread() .....	*skwrite()
<b>gsk_secure_soc_write()</b>	<b>gsk_secure_soc_read()</b>
*skwrite() .....	*skread()
<b>gsk_secure_soc_read()</b>	<b>gsk_secure_soc_write()</b>
*skread() .....	*skwrite()
<b>gsk_secure_soc_close()</b>	<b>gsk_secure_soc_close()</b>
*skwrite() .....	*skread()
*skread() .....	*skwrite()
close()	close()
	close()

Figure A-1 Socket programming model that uses SSL

For more information about the used functions, see the following publications:

- ▶ *IBM Language Environment for z/VSE C Run-Time Library Reference*, SC33-6689
- ▶ *TCP/IP for VSE/ESA - IBM Program Setup and Supplementary Information*, SC33-6601

### A.2.1.2 Other APIs that provide SSL/TLS functions

Similar to the SSL/TLS socket application that is shown in A.2.1.1, “Language Environment for VSE C Run-Time Socket API” on page 415 and Figure A-1, you can also build socket applications by using SSL/TLS with the following APIs:

- ▶ EZASOKET API and EZASMI API

The EZASOKET call interface is provided for COBOL, PL/I, and HLASM programs. The EZASMI macro interface is provided for HLASM programs. The functions are described in *TCP/IP for VSE/ESA - IBM Program Setup and Supplementary Information*, SC33-6601.

- ▶ Secure Sockets Layer API

The functions are described in *TCP/IP for VSE V1R5.0 Optional Features*, SC33-6767. This API can be used with the basic socket interface that is described in *TCP/IP for VSE V1R5.0 Programmer’s Guide*, SC33-6766.

### A.2.1.3 Using CryptoVSE API functions

If you do not want to implement the full SSL/TLS handshaking, but still use some functions for authentication, data encryption, or message integrity, you can use the functions of the CryptoVSE API as described in *TCP/IP for VSE V1R5.0 Optional Features*, SC33-6767.

This API can be used with the basic socket interface that is described in *TCP/IP for VSE V1R5.0 Programmer's Guide*, SC33-6766. It also can be used with the Language Environment/VSE C Run-Time Socket API that is described in *IBM Language Environment for z/VSE C Run-Time Library Reference*, SC33-6689.

## A.2.2 CPU Assist Facility

The CPU Assist Facility (CPACF) is a hardware feature (feature code #3863) that is available on z890, z990, z9, and z10 processors. It provides symmetric algorithms (DES, TDES, AES), SHA hash functions, and a pseudo random number generator (PRNG). The CPACF functionality is given by a set of machine instructions and can be used directly from user programs. The instructions are described in *z/Architecture Principles of Operation*, SA22-7832 (see instructions KM, KMC, and PRNG).

## A.2.3 Summary of available SSL functions

The SSL functions and their dependencies that were available at the time of this writing are listed in Table A-1.

Table A-1 SSL functions and dependencies

SSL functions	CWS with Firefox	CWS with Internet Explorer	FTP	Telnet	HTTPS	z/VSE Java-based Connector	WebSphere MQ
SSL 3.0	Yes	Yes	Yes	Yes	Yes	Yes	Yes
TLS 1.0	-	-	Yes <sup>a</sup>	Yes	Yes	Yes	-
DES	Yes, only strong encryption	Yes	Yes	Yes	Yes	Yes	Yes
TDES	Yes, only strong encryption	Yes	Yes	Yes	Yes	Yes	Yes
AES	-	-	Yes	Yes	Yes	Yes	-
Client authentication	Yes	Yes	-	Yes	Yes	Yes <sup>b</sup>	Yes
Client authentication with user ID mapping	Yes	Yes	-	-	-	Yes <sup>b</sup>	-

a. With TCP/IP fix ZP15F264 (APAR PK33472)

b. Dependent on the Java installation



# B

## Setting up and using Keyman/VSE

Keyman/VSE is a tool that is used to manage the public key infrastructure in z/VSE. It is a Java application that is typically installed on a personal computer (PC).

In this appendix, we describe the setup and use of the Keyman/VSE tool with the help of examples.

In addition, we provide some basic information about RSA keys and show the related TCP/IP utilities.

This appendix includes the following topics:

- ▶ B.1, “Keyman/VSE” on page 418
- ▶ B.2, “Installing the prerequisite programs” on page 418
- ▶ B.3, “Initial Keyman/VSE set-up” on page 419
- ▶ B.4, “Basic characteristics of RSA keys” on page 420
- ▶ B.5, “Relationship to TCP/IP utilities” on page 422
- ▶ B.6, “Keystores” on page 425
- ▶ B.7, “Using Keyman/VSE” on page 426
- ▶ B.8, “Selected Keyman/VSE functions” on page 429
- ▶ B.9, “Observation” on page 437



## B.1 Keyman/VSE

With Keyman/VSE, you can manage the z/VSE-specific public key infrastructure (PKI). This feature includes the creation of RSA key pairs and SSL certificates, which you can upload to z/VSE and store in z/VSE-side keyrings. You also can use Keyman/VSE for workstation-based keystores and importing and exporting of PGP public keys.

Keyman/VSE is intended as a system administrator tool. Although it provides wizard prompts to create z/VSE keyrings, it requires some basic knowledge about RSA keys, SSL certificates, and keystores.

For more information about the use of Keyman/VSE for specific tasks, see Chapter 4, “Cryptography on z/VSE” on page 93, and Chapter 5, “Secure Sockets Layer with z/VSE” on page 197.

In this appendix, we show you some new examples in “Using Keyman/VSE” on page 426 and “Selected Keyman/VSE functions” on page 429.

In our test setup, we used the following software:

- ▶ z/VSE V4R3 with APAR DY47171 / PTF UD53607
- ▶ Java 1.6.0\_18 from Sun Microsystems
- ▶ VSE Connector Client on the workstation side
- ▶ VSE Connector Server that is running on VSE
- ▶ Keyman/VSE, update from Nov. 2010
- ▶ TCP/IP for VSE/ESA 1.5F with APAR PM28847 / PTF UK63070, including following zaps:
  - ZP15F411 for the IPCRYPTO.PHASE
  - ZP15F412 for the IPDSCIAL.PHASE
  - ZP15F413 for the CIALCERT.PHASE
  - ZP15F414 for the CIALROOT.PHASE
  - ZP15F415 for the CIALGPRV.PHASE
  - ZP15F416 for the CIALSIGV.PHASE
  - ZP15F417 for the CIALSRVR.PHASE
  - ZP15F418 for the CIALPRVK.PHASE
  - ZP15F419 for the IPCRYPTS.OBJ

## B.2 Installing the prerequisite programs

The Keyman/VSE tool requires the installation of the z/VSE Connector Client. You can download the most current versions of the z/VSE Connector Client and Keyman/VSE from this web page:

<http://www.ibm.com/systems/z/os/zvse/downloads/>

After downloading and unpacking the .zip file, run one of the contained installation scripts:

- ▶ For Windows: Setup.bat
- ▶ For Windows NT: Setup.cmd
- ▶ For UNIX/Linux: Setup.sh

## B.3 Initial Keyman/VSE set-up

Before you can upload keys to z/VSE, you must define your z/VSE system to Keyman/VSE.

When starting Keyman/VSE the first time, the GUI shows that no z/VSE system is defined. To define your first z/VSE system, click the **VSE Host properties** button, as shown in Figure B-1.

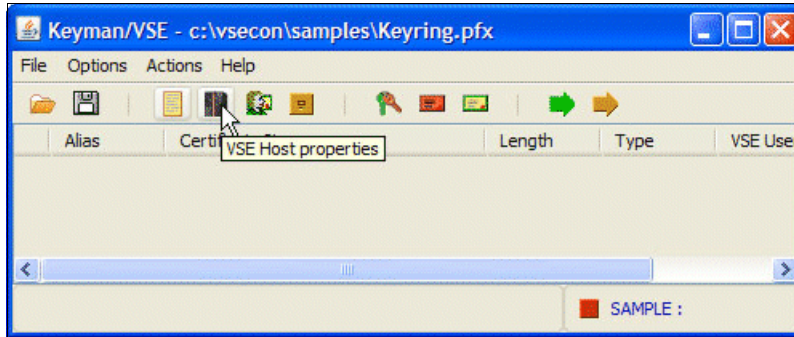


Figure B-1 Initial Keyman/VSE window

In the VSE Host - Properties window, click **New** to define a new z/VSE host. Then, enter the z/VSE properties, such as IP address, VSE user, and password, as shown in Figure B-2.

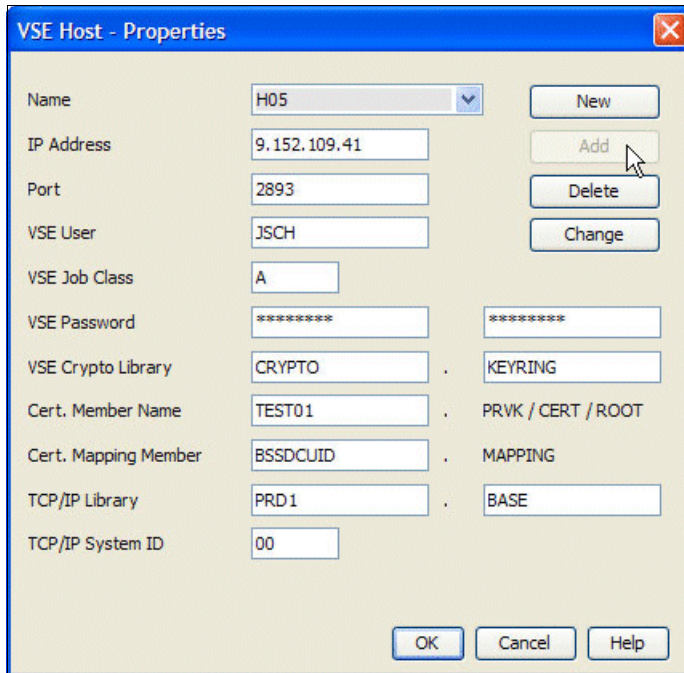


Figure B-2 z/VSE host properties

Click **Add** to add this definition to the Keyman/VSE configuration and continue adding z/VSE systems or click **OK** to finally add this definition and exit the window.

Now, the initial setup is done. You can start the z/VSE Connector Server on the z/VSE side.

## B.4 Basic characteristics of RSA keys

This section provides selected information about RSA keys and certificates that are related to Keyman/VSE. For more information about cryptography on z/VSE, see Chapter 4, “Cryptography on z/VSE” on page 93.

### B.4.1 Symmetric and asymmetric keys

RSA keys are called *asymmetric keys* because they consist of key pairs. One key is used for encrypting data while a second key is used for decrypting.

In contrast to RSA, a second type of digital key is called *symmetric key*. With symmetric keys, encryption and decryption of data is performed by using the same key. Examples of symmetric encryption algorithms include DES, Triple-DES, and AES.

Usually, RSA keys are not used for encrypting huge amounts of data because of their high computational effort. Therefore, the secure data exchange uses a combination of both methods. The data is encrypted by using a symmetric encryption algorithm.

The symmetric key, which consists of a few bytes (normally 8 up to 32 bytes) is encrypted with an RSA key. The encrypted symmetric key is then put together with the encrypted data. This technique is used by Encryption Facility for z/VSE. For more information, see Chapter 4, “Cryptography on z/VSE” on page 93.

SSL uses RSA encryption when opening a connection. After the connection is established, the data is symmetrically encrypted.

Keyman/VSE does not support the creation of symmetric keys.

### B.4.2 Internal structure of RSA keys

An RSA key pair consists of two keys: the public key and the private key. The public key consists of the public exponent and the public modulus. The private key can exist in two different formats: the Modulus-Exponent (ME) format and the Chinese-Remainder-Theorem (CRT) format. In the private ME form, the private key consists of the private exponent and the public modulus.

In the CRT form, the private key consists of the following CRT parts:

- ▶ First prime  $p$
- ▶ Second prime  $q$
- ▶ Inverse first prime  $dp$
- ▶ Inverse second prime  $dq$
- ▶ CRT coefficient  $U$  and the public modulus

The ME format is usually used when encrypting information, while the CRT format is used for decrypting.

The following structure is used:

- ▶ Public key:
  - Public exponent ( $e$ )
  - Public modulus ( $n$ )

- ▶ Private key:
  - Private exponent
    - CRT-form: p, q, dp, dq, U
    - ME-form: private exponent (d)
  - Public modulus (n)

**Note:** The public modulus is part of the public key and the private key. You can see these key parts when the settings of an RSA key are displayed in Keyman/VSE.

### B.4.3 Types of SSL certificates

The following types of SSL certificates are available:

**Root certificates** These certificates belong to a certificate authority (CA) and can be self-signed or signed by an official CA, such as Thawte or Verisign. Root certificates can be used to sign user certificates and establish trust.

**User certificates** These certificates are signed by a CA root certificate and are a vehicle to securely transport a public key over a network.

You can use Keyman/VSE to create self-signed root certificates for testing purposes in a closed environment. Because they are not signed by an official CA, no one outside your test environment trusts them. With Keyman/VSE, you can also create CA-signed root certificates.

### B.4.4 Structure of a keyring

A complete z/VSE keyring consists of the following components:

- ▶ The RSA key pair, with member type PRVK
- ▶ The root certificate, with member type ROOT
- ▶ The user certificate, with member type CERT

Figure B-3 on page 422 shows how these parts are related to each other with the steps to create these parts.

Keyman/VSE supports these steps to create a z/VSE keyring through wizard prompts. You also can perform each step manually.

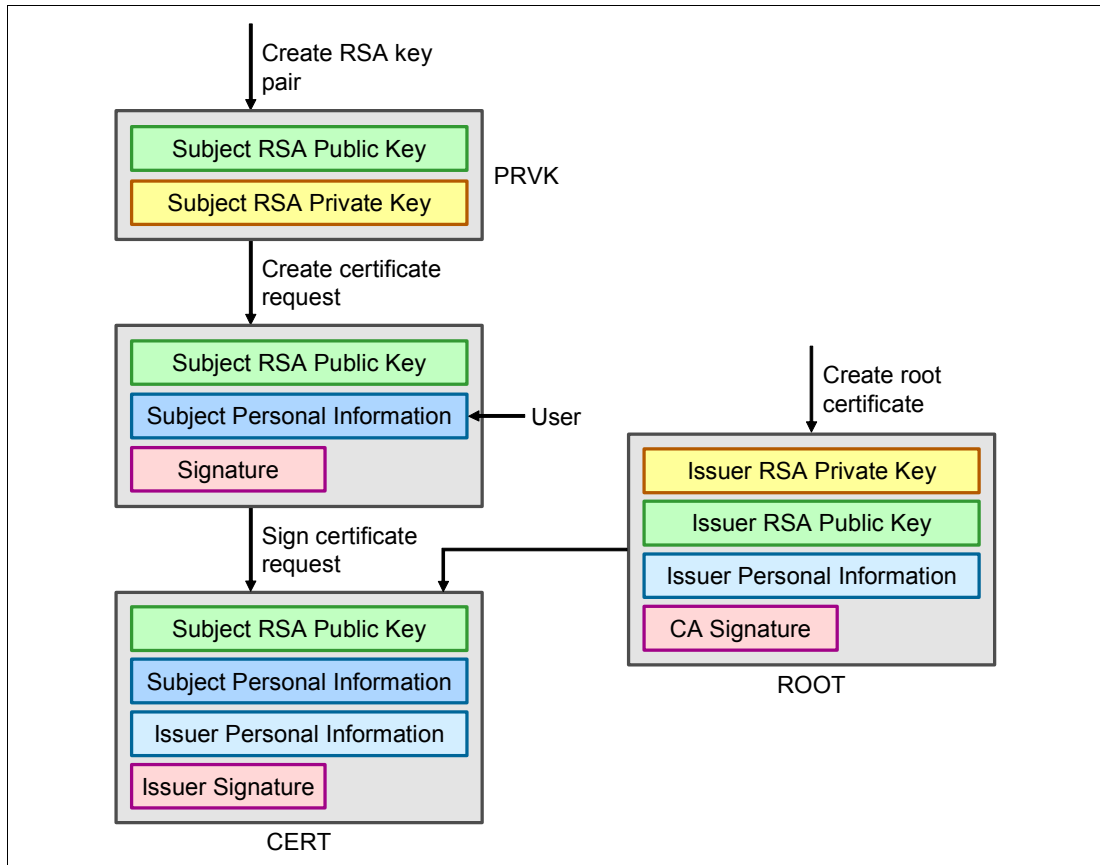


Figure B-3 z/VSE keyring structure

The term *subject* always refers to the person or site who requests an SSL certificate. The term *issuer* refers to the authority that signs a certification request.

## B.5 Relationship to TCP/IP utilities

Keyman/VSE uses utilities that are provided by TCP/IP for VSE/ESA to upload keys and certificates to z/VSE. These utilities are primarily intended to be used manually, but are used internally by Keyman/VSE.

### B.5.1 CIALSRVR

The Keyman/VSE starts the utility CIALSRVR when uploading an RSA key pair. CIALSRVR receives the encrypted key material from a CIAL client program and stores the RSA key in a z/VSE library member with member type PRVK. Keyman/VSE acts here as the CIAL client; it implements the CIAL client/server protocol.

You can use the JCL that is shown in Example B-1 to start the CIALSRVR utility.

*Example: B-1 Job CIALSRVR*

---

```
// JOB CIALSRVR
// OPTION SYSPARM='00'
// LIBDEF PHASE,SEARCH=(PRD1.BASE)
// EXEC CIALSRVR,SIZE=CIALSRVR,PARM='CRYPTO.KEYRING.TEST01'
SETPORT 6045
/*
/ &
```

---

When started, CIALSRVR listens on the specified port to receive the RSA key material that is sent from a CIAL client. After storing the key in a PRVK member, CIALSRVR ends.

## B.5.2 CIALROOT

The CIALROOT utility uploads a CA-root certificate to z/VSE and catalogs it as a z/VSE library member with member type ROOT. Keyman/VSE uses JCL that is similar to the JCL that is shown in Example B-2.

*Example: B-2 Job CIALROOT*

---

```
// JOB CIALROOT
// OPTION SYSPARM='00'
// LIBDEF PHASE,SEARCH=(IJSYSRS.SYSLIB,PRD1.BASE)
// EXEC CIALROOT,SIZE=CIALROOT,PARM='CRYPTO.KEYRING.TEST01'
-----BEGIN CERTIFICATE-----
MIICsDCCAhkCBB89tb0wDQYJKoZIhvcNAQEFBQAwwZ4xHjAcBgkqhkiG9w0BCQEW
D3p2c2VAZGUuaWJtLmNvbTElMAkGA1UEBhMCREUxGzAZBgNVBAGTEkHhZGVuLWV1
ZXJ0dGVtYmVyZzETMBEGA1UEBxMKQm91Ymxpbmd1bjEMMAoGA1UEChMDSUJNMRQw
EgYDVQQLFwltJQk0gr2VybWVudFUEZMBCGA1UEAxMQUHJpdmFOZSBLZXkgQ2VydAe
...
K1Ky08BTiqvh2ZJZZT0TDFPPrM6nmKT2Y9dqz6i0kKpIkQZ8GcT+oEgiJTg29tB
3wnt+E0jVFzvnprlAgMBAAEwdQYJKoZIhvcNAQEFBQAwwZ4xHjAcBgkqhkiG9w0BCQEW
D3p2c2VAZGUuaWJtLmNvbTElMAkGA1UEBhMCREUxGzAZBgNVBAGTEkHhZGVuLWV1
ZXJ0dGVtYmVyZzETMBEGA1UEBxMKQm91Ymxpbmd1bjEMMAoGA1UEChMDSUJNMRQw
EgYDVQQLFwltJQk0gr2VybWVudFUEZMBCGA1UEAxMQUHJpdmFOZSBLZXkgQ2VydAe
-----END CERTIFICATE-----
/*
/ &
```

---

The certificate data is given in its Base-64 encoded portable text form. A root certificate can be self-signed or signed by an official CA, such as Thawte or Verisign.

## B.5.3 CIALCERT

The CIALCERT utility uploads a user certificate to z/VSE and catalogs it as a z/VSE library member with member type CERT. The user certificate contains the public key part of the related PRVK member and is signed by the related root certificate from member ROOT.

Example B-3 on page 424 shows the JCL to use CIALCERT.

*Example: B-3 Job CIALCERT*

---

```
// JOB CIALCERT
// OPTION SYSPARM='00'
// LIBDEF PHASE,SEARCH=(IJSYSRS.SYSLIB,PRD1.BASE)
// EXEC CIALCERT,SIZE=ICIALCERT,PARM='CRYPTO.KEYRING.TEST01'
-----BEGIN CERTIFICATE-----
MIICpjCCAg8CBB90yGgwDQYJKoZIhvcNAQEFBQAwwZ4xHjAcBgkqhkiG9w0BCQEW
D3p2c2VAZGUuaWJtLmNvbTELMAGGA1UEBhMCREUxGzAZBgNVBAGTEkjhZGVuLVd1
ZXJ0dGVtYmVyZzETMBEGA1UEBxMKQm91Ymtpbmd1bjEMMAoGA1UEChMDSUJNMRQw
EgYDVQQLExtJQk0gR2VybWFueTEZMBCGA1UEAxMQUHJpdmFOZSBLZXkgQ2VydDAe
...
e9gAC2j+bGbk4X2/qNd89YMH4zNjQdr24QNLQ/KTA4/kHQXMj0/E/5iqtGk188Z
MPCAwEAATANBgkqhkiG9w0BAQUFAA0BgQBEEREKSunbLAHCo36IN8N1K2Ly7rf6
yaABtu01z77jw3K1ar1SVp9/Rynzi57jKVvh80w4E0G87IUCZKwb8RUmt119BC3i
852CEKo1X6zE0JY1nqF73oXVMuFVLJp1r0XPY1uCfYdv8RYeAYyGEV0i0L7hY99W
m0JiBdh8Dwa1AA==
-----END CERTIFICATE-----
/*
/ &
```

---

## B.5.4 CIALSIGV

The CIALSIGV utility verifies the signature of a z/VSE keyring.

The keyring consists of three z/VSE library members with following member types:

- PRVK**      Contains the RSA key pair.
- CERT**     Contains the public key of the member PRVK with some personal information and a signature that is created by the root certificate from member ROOT.
- ROOT**     It is a self-signed or CA-signed root certificate.

After creating a complete z/VSE keyring, you can use CIALSIGV to verify its accuracy.

Example B-4 shows the JCL to use the utility CIALSIGV.

*Example: B-4 Job CIALSIGV*

---

```
// JOB CIALSIGV VERIFY SIGNATURE
// OPTION SYSPARM='00'                    SYSID OF MAIN TCP/IP PARTITION
// LIBDEF *,SEARCH=(PRD2.CONFIG,PRD1.BASE)
// EXEC CIALSIGV,SIZE=ICIALSIGV,PARM='CRYPTO.KEYRING.TEST01'
/*
/ &
```

---

In Keyman/VSE, the CIALSIGV utility is used when you select **Actions** → **Validate VSE keyring**.



## B.5.5 CIALCREQ

The CIALCREQ utility is used to create a certificate request from a specific PRVK member, which means to take the public key of the member PRVK and add some user-provided personal information. Then, a hash value is calculated from public key and personal information, and finally, signed with the private key.

The certificate request (personal information, public key, hash algorithm, and signature) is then sent to a CA, which constructs a x.509 certificate from the provided information. The obtained x.509 certificate is then cataloged as a CERT member on z/VSE.

For more information about the syntax for certification requests, see RFC 2314. For more information about how to create a certificate request from a specific RSA key pair, see “Using a PRVK member on z/VSE” on page 432.

**Restriction:** CIALCREQ cannot create a certificate request from a 4096-bit key.

Example B-5 shows the JCL that is used to create a certificate request with CIALCREQ. The keywords, such as Common-name and Organization Unit, must be specified in mixed-case letters.

*Example: B-5 Job CIALCREQ*

---

```
// JOB CIALCREQ CREATE CERT REQUEST
// OPTION SYSPARM='00'                SYSID OF MAIN TCP/IP PARTITION
// LIBDEF *,SEARCH=(PRD2.CONFIG,PRD1.BASE)
// EXEC CIALCREQ,SIZE=CIALCREQ,PARM='CRYPTO.KEYRING.TEST01'
Common-name: z/VSE Development
Organization Unit: Development
Organization: IBM Germany
Locality: Boeblingen
State: BW
Country: DE
/*
/ &
```

---

Keyman/VSE uses the CIALCREQ utility to create a certificate request from an RSA key that is stored on z/VSE in a PRVK member. For more information, see “Using a PRVK member on z/VSE” on page 432.

## B.6 Keystores

You find many different types of keystores that are used on workstation platforms. The following formats are typical and supported by Keyman/VSE:

- PFX** The Personal Information Exchange (PFX) format was initially defined by RSA Security and conforms to the PKCS#12 standard. PFX files can contain multiple keys and certificates. The files are password-protected. Web browsers, such as Microsoft Internet Explorer and Mozilla Firefox, support PFX files. At times, the file extension p12 is also used for the PFX format.
- JKS** The Java Keystore (JKS) format is provided by Sun Microsystems. It is the standard keystore format for Java applications. JKS files are also protected by a password.

Usually, web browsers cannot handle JKS files. Part of each Java installation is the `keytool.exe`, which can be used for maintaining JKS keystores.

You can use both formats for storing SSL certificates on a workstation when setting up an SSL connection to z/VSE. On the z/VSE side, the related keys and certificates are stored in a z/VSE keyring that consists of z/VSE library members.

In Keyman/VSE, you can specify the type of keystore. When you select **Options** → **Keyring file properties....**, the Local Keyring File - Properties window opens. Select the file type as shown in Figure B-4.



Figure B-4 Local keyring file properties

This window is also displayed when you save keys and certificates.

Pretty Good Privacy (PGP) uses the keystore format Key Database (KDB). KDB is not supported by Keyman/VSE. However, Keyman/VSE can make PGP public keys usable for z/VSE.

Encryption Facility for z/VSE V1.2 supports the OpenPGP message format. Therefore, Keyman/VSE supports importing PGP public keys and converting them into the x.509 certificate format. This ability is necessary for exchanging PGP public keys with other platforms, such as workstations and z/OS.

## B.7 Using Keyman/VSE

The Keyman/VSE tool features the following main tasks:

- ▶ Setting up SSL between a workstation and z/VSE
- ▶ Setting up the RSA keys for Encryption Facility for z/VSE

In this section, we describe two other Keyman/VSE functions to support these main tasks:

- ▶ Support for a CIALEXIT phase
- ▶ Support the CIAL trace

## B.7.1 Support a CIALEXIT phase

The version of Keyman/VSE from October 2010 supports a CIALEXIT phase. With the CIALEXIT phase, you define a passphrase that must be entered whenever you upload a private key to z/VSE by way of the CIALSRVR utility.

If you do not use a CIALEXIT, a CIAL client encrypts the created binary key material with a hardcoded Triple-DES or AES key. An SHA-1 hash of a hardcoded passphrase is appended to the key material and sent to CIALSRVR, which verifies the passphrase hash before writing the key into a PRVK member.

With a CIALEXIT, this process is much more secure because the passphrase and the involved encryption keys are under full control of the client. A sample CIALEXIT program is provided with the Keyman/VSE installation package.

If a CIALEXIT phase is cataloged on z/VSE, CIALSRVR enforces Keyman/VSE to display a password prompt. The entered passphrase is then SHA-1 hashed and sent to CIALSRVR, which verifies the password hash with the password in CIALEXIT.

To upload a private RSA key to z/VSE, right-click the key pair and select **Upload to VSE** (see Figure B-5).

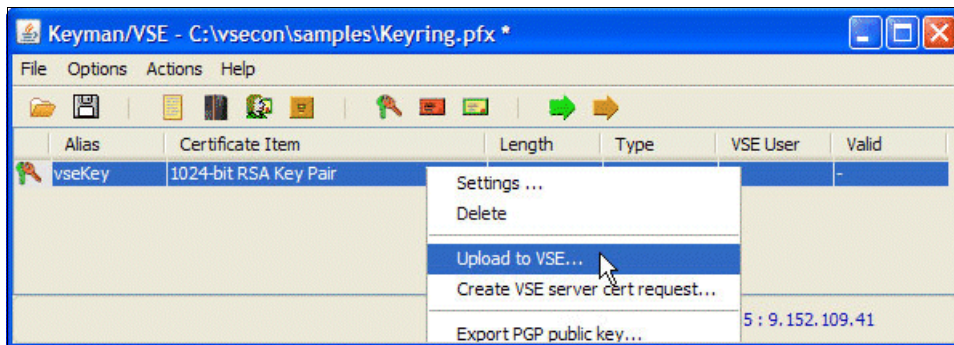


Figure B-5 Keyman/VSE keyring list

Specify your keyring library and member name, as shown in Figure B-6. Click **Upload**.

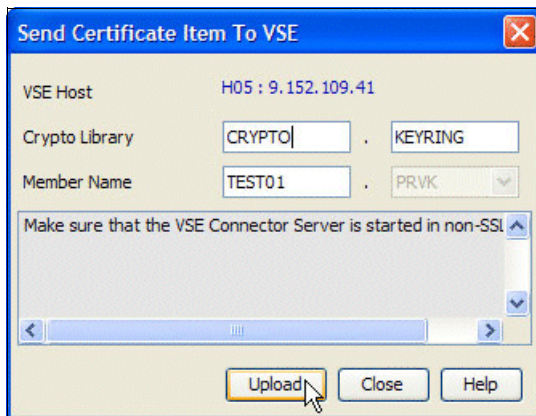


Figure B-6 Send certificate item to z/VSE

Because a CIALEXIT phase is available, you see the passphrase prompt (see Figure B-7).

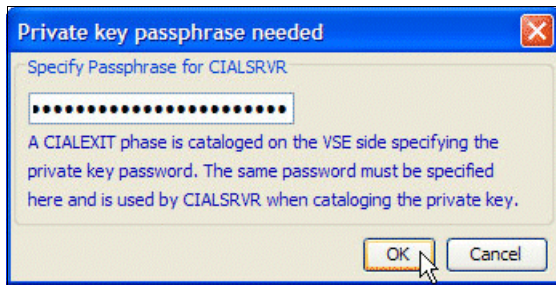


Figure B-7 Private key passphrase needed

Enter the secret passphrase that is specified in CIALEXIT and click **OK**.

Figure B-8 shows that the upload of the private key is finished. Click **Close** to end the procedure.



Figure B-8 Private key sent

When a CIALEXIT phase is used, consider the following points:

- ▶ CIALEXIT not only requests a secret passphrase from the user to be allowed to store the key data, but also uses the hardcoded TDES or AES key in CIALEXIT to encrypt the RSA key data in the PRVK member. This process causes the key and CIALEXIT phase to be associated and makes it impossible to punch or restore a key at another site.  
As a result, if the CIALEXIT phase is lost or changed accidentally, the related RSA key is no longer usable.
- ▶ Regardless of whether a CIALEXIT phase is used, Keyman/VSE encrypts the RSA key with a hardcoded TDES or AES key before sending the key blob to CIALSRVR. If a CIALEXIT phase exists, CIALSRVR reencodes the RSA key with the TDES/AES key in CIALEXIT before writing it to the PRVK member.

## B.7.2 Activating a CIAL trace

If you encounter any problems cataloging keys or certificates, you can activate a CIAL trace. In the Keyman/VSE main window, select **Options** → **Trace settings** (see Figure B-9).

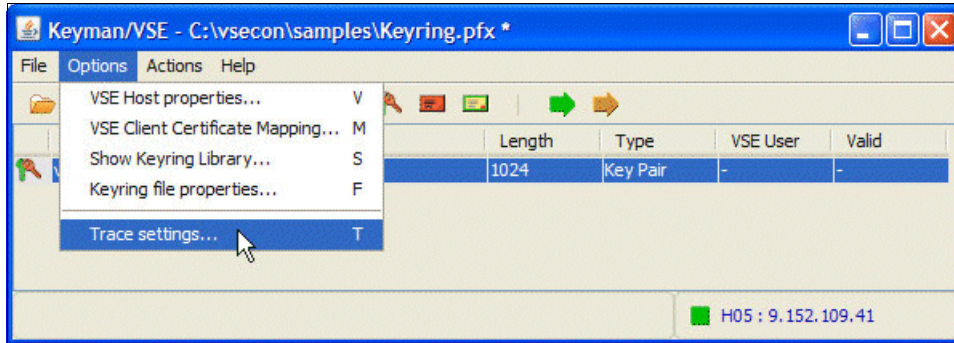


Figure B-9 Keyman/VSE options

In the Trace Settings window, select **CIAL Trace**, as shown in Figure B-10.

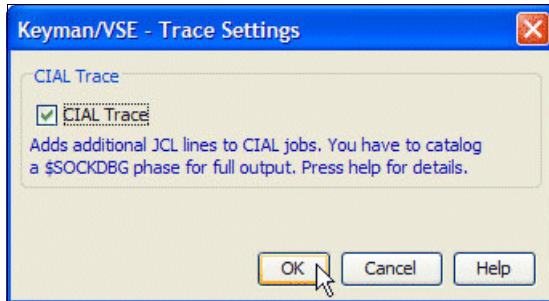


Figure B-10 Keyman/VSE trace settings

Click **OK** to activate this trace.

The CIAL trace generates SDUMPS in the output of all CIAL utilities.

The following JCL statements are added to the JCL created by Keyman/VSE:

```
// OPTION NOSYSDMP  
// UPSI 1
```

In addition to activating the trace in Keyman/VSE, you must catalog a \$SOCKDBG phase for full list output. For more information, see “Tracing on z/VSE” on page 218.

## B.8 Selected Keyman/VSE functions

In this section, we provide more information about the following Keyman/VSE functions that were not described previously in this publication:

- ▶ Creating keys
- ▶ Creating certificates
- ▶ Using certificate mappings
- ▶ Displaying key information



## B.8.1 Creating keys

Keyman/VSE provides the following functions for generating RSA keys:

- ▶ Locally in Keyman/VSE
- ▶ Directly on the mainframe (since z/VSE V4R3)

### B.8.1.1 Local RSA key generation

A key can be created locally in Keyman/VSE by using Java functionality.

Select **Local** in the Specify Properties For New RSA Key window. Enter the alias and specify key length, as shown in Figure B-11.



Figure B-11 Local RSA key definition

The key must then be uploaded to z/VSE in a separate step. Although the key material is encrypted before sending it over the network, this process is a security exposure. You might consider the use of a CIALEXIT phase in this case. For more information, see “Support a CIALEXIT phase” on page 427.

### B.8.1.2 RSA key generation on mainframe

A key can be created directly on the mainframe by way of the CIALSRVR utility. This function requires cryptographic hardware on the host side. Consider the following points:

- ▶ Creating keys of up to 2048-bit key length requires a Crypto Express2 card
- ▶ Creating keys with 4096-bit key length requires a Crypto Express3 card

Select **Mainframe** in the Specify Properties For New RSA Key window. Enter the information for library, member, and key length, as shown in Figure B-12 on page 431.

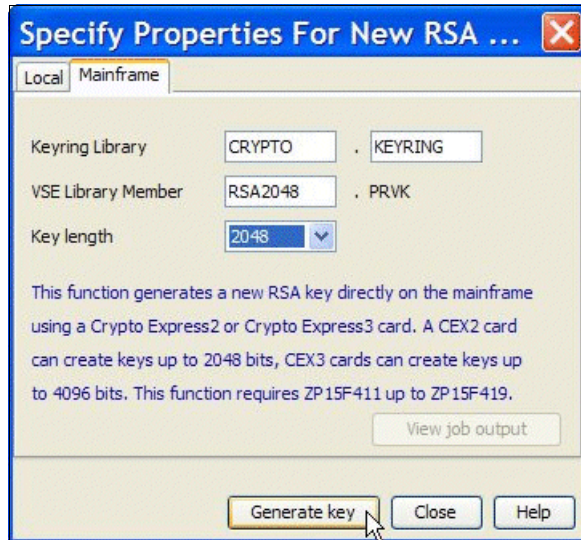


Figure B-12 RSA key definition on mainframe

This function uses the **GENRSAPK** command that is provided by the CIALSRVR utility.

**Note:** The use of keys that are greater than 1024 bits always requires a crypto card on the host side. TCP/IP for VSE/ESA provides a software implementation of the RSA algorithm for 512 and 1024-bit keys only.

## B.8.2 Creating certificates

The following methods can be used to create a certificate based on a specific RSA key, some personal information to be specified by the user, and a root certificate that is used to create the signature:

- ▶ With wizard dialogs
- ▶ Locally in Keyman/VSE
- ▶ By using a PRVK member on z/VSE

### B.8.2.1 Using the wizard dialogs

The easiest way to create a complete z/VSE keyring (including certificates) is to use the wizard dialogs.

When you click the left (green) arrow in the Keyman/VSE main window, Keyman/VSE creates a self-signed keyring that uses a self-signed root certificate. If you click the right (yellow) arrow, you can interact with an external CA, such as Thawte or Verisign.

### B.8.2.2 Creating a certificate locally in Keyman/VSE

This option requires some knowledge about the relationship between RSA key, certificate request, and root certificate, but you can manipulate all items manually. All tasks are performed locally in Keyman/VSE. No CIAL utilities are used.

To create a certificate, complete the following steps:

1. Create an RSA key.
2. Create a self-signed root certificate.
3. Create a certificate request with the public key from the RSA key pair.
4. Sign the certificate request with the root certificate.



5. Delete the certificate request.

For more information about these steps, see the Keyman/VSE online help by selecting **Help** → **General help** → **How to** in the Keyman/VSE main window.

### B.8.2.3 Using a PRVK member on z/VSE

The first two methods assume that the RSA key is present in Keyman/VSE. This option allows the use of a remote PRVK member for creating the certificate request.

Right-click your root certificate and select **Create VSE cert via CIALCREQ**, as shown in Figure B-13.

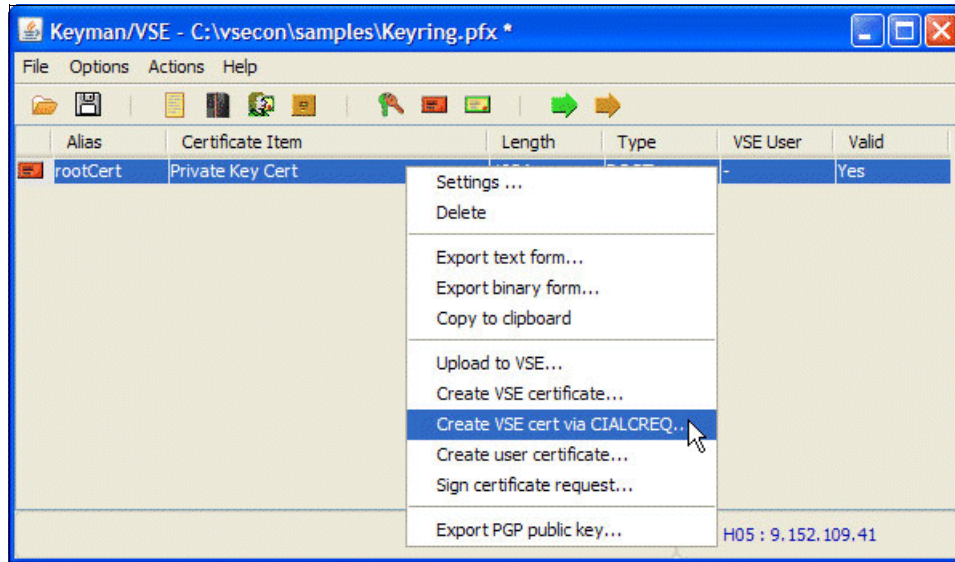


Figure B-13 Create z/VSE certificate

In the next window, specify the personal information and the PRVK member (see Figure B-14).

Personal Information for X509 User Certificate

Common name: VSE Certificate KM1024

Organizational unit: Development

Organization: IBM

City/Location: Boeblingen

State/Province: Baden-Wuerttemberg

Country: DE Germany (DE)

e-mail: zvse@de.ibm.com

Alias: userCert

Expires: 2011-9-17 1 year

VSE key to be used: CRYPTO . KEYRING . TEST01 . PRVK

Make sure that the VSE Connector Server is started on H05

Generate Close View output Help

Figure B-14 Personal information

Click **Generate** to create the certificate request on z/VSE.

**Note:** This function does not work with a 4096-bit key because the CIALCREQ utility cannot handle this key length. Instead, use the native Keyman/VSE function. In the menu of an RSA key, select **Create VSE server cert request**.

### B.8.3 Using certificate mappings

Keyman/VSE supports the mapping of SSL client certificates to z/VSE user IDs. This process is used mainly by CICS Web Support (CWS) with SSL client authentication. When a CWS SSL client attempts to start a CICS transaction, the mapping between SSL client certificate and z/VSE user ID allows CICS to decide whether to grant or deny access.

The z/VSE Java-based connector also supports certificate mapping. The z/VSE Navigator application is an example of the use of certificate mapping to log on to z/VSE without being prompted for a password.

To map a certificate to a z/VSE user ID with the Keyman/VSE, right-click the user certificate and select **Map to VSE user**, as shown in Figure B-15 on page 434.

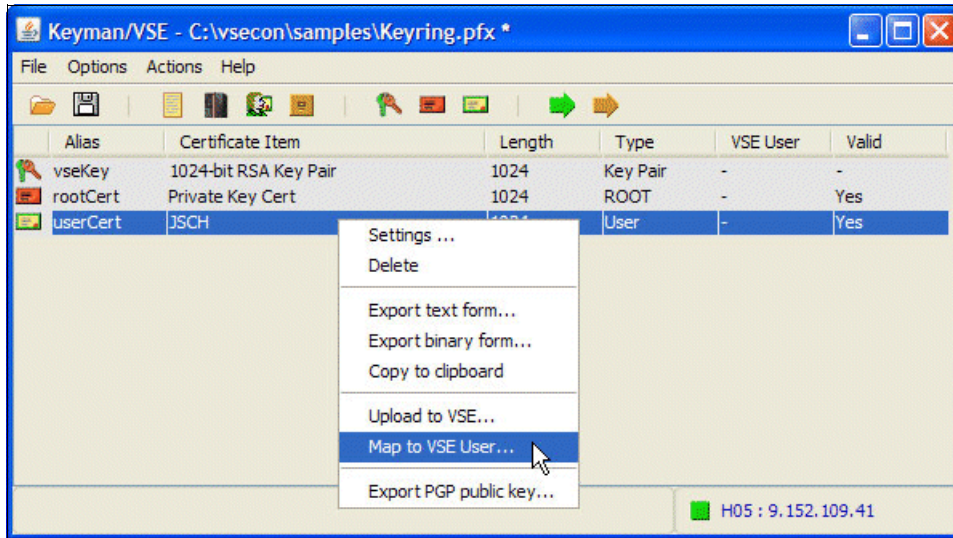


Figure B-15 Map certificate to z/VSE user ID

In the next window, enter the z/VSE user ID to which this client certificate will be mapped (see Figure B-16).

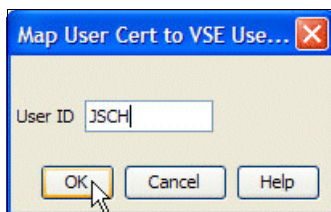


Figure B-16 Specify z/VSE user ID

Click **OK**. The Keyman/VSE stores this mapping information locally.

To make this information available for z/VSE, you must upload this certificate to z/VSE. No difference exists between uploading mapped and unmapped certificates. Select **Upload to VSE** from the menu of the certificate.

Mapped SSL client certificates are not uploaded to z/VSE by way of CIAL utilities. Instead, we use the BSSDCERT utility that is provided by the Basic Security Manager. For more information about the manual use of BSSDCERT, see *z/VSE Administration*, SC33-8304.

After you upload a mapped certificate to z/VSE, you can display the mapping by way of the function Client certificate mapping, as shown in Figure B-17.

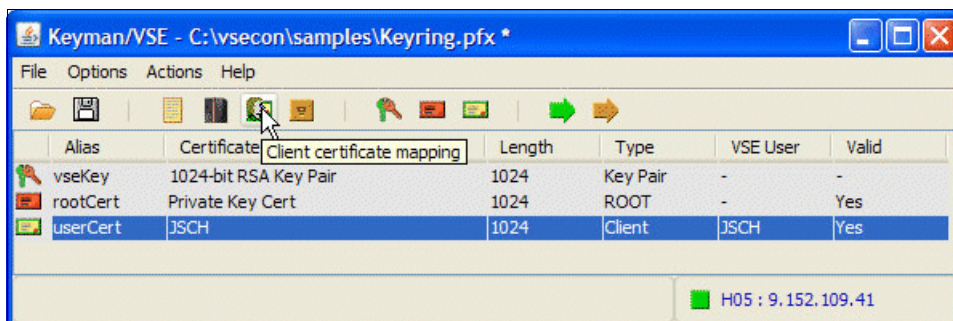


Figure B-17 Click Client certificate mapping



Figure B-18 shows the window with the certificate mapping information as it is available on z/VSE. It also shows the same information as the Interactive Interface window “MAINTAIN CERTIFICATE - USERID LIST” (fast path 2.8.4).

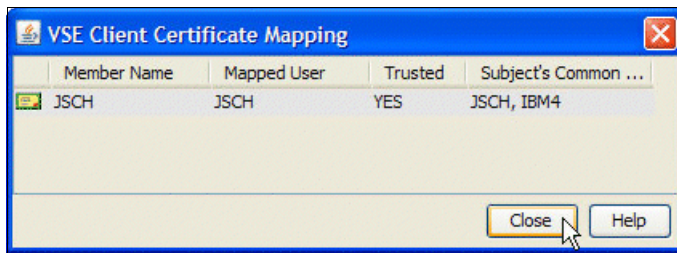


Figure B-18 Show client certificate mapping

These functions cover your normal maintenance process for keys and certificates. The next section describes some advanced functions, which provide more information about keys and certificates.

### B.8.4 Displaying key material

You can display and export the key parts of an RSA key or SSL certificate. This information can help you compare the key parts of different keys or certificates for debugging purposes. Double-click a key or certificate in the Keyman/VSE main window to open the Setting window.

Figure B-19 shows the settings of a certificate.

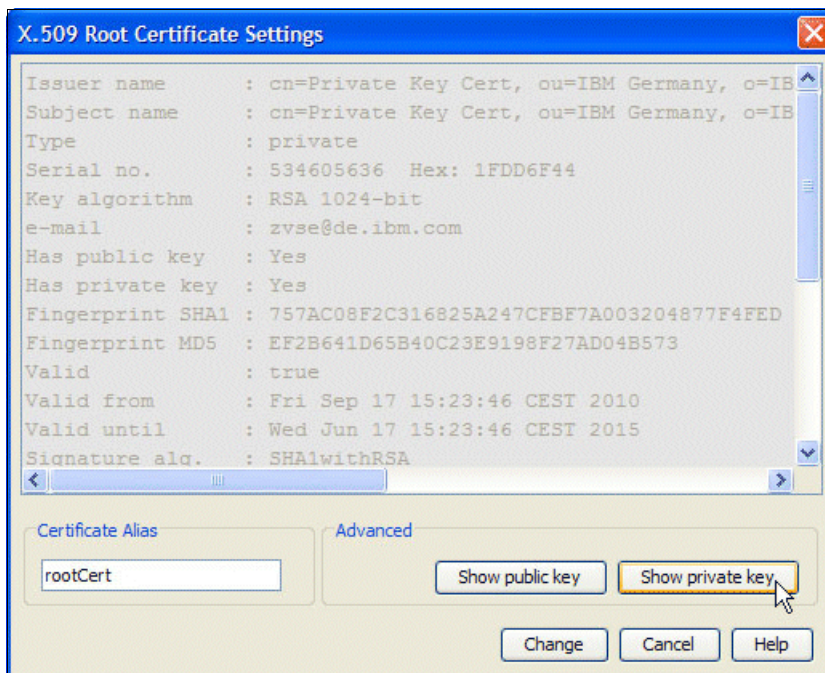


Figure B-19 Certificate settings

For certificates, the public and private key parts can be displayed separately. To get the private key, click **Show private key**.

The window with the private key is displayed, as shown in Figure B-20.

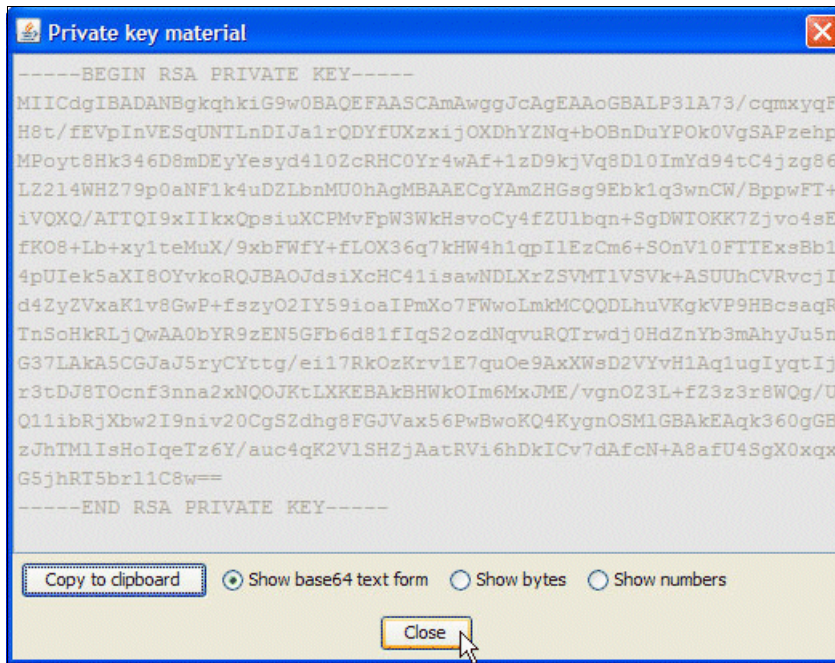


Figure B-20 Private key material

For more processing, you can copy this information to the clipboard by clicking **Copy to clipboard**.

For RSA keys, only one window displays the settings (see Figure B-21).

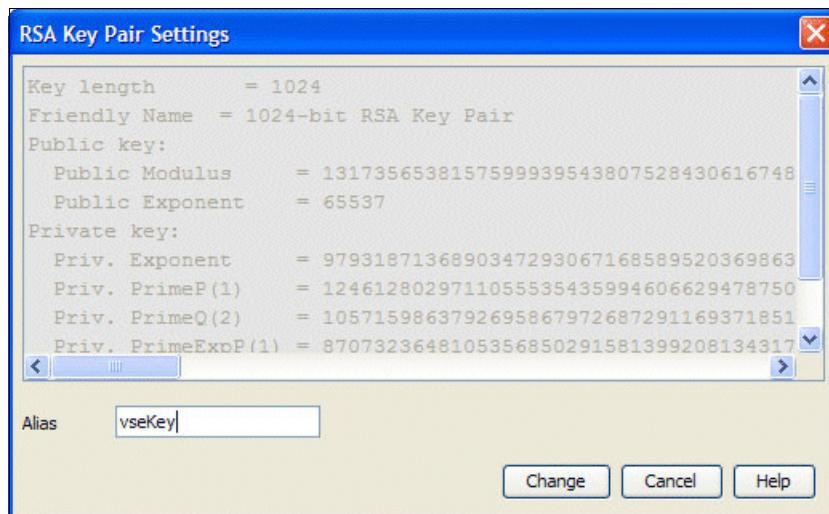


Figure B-21 RSA key pair settings

## B.9 Observation

This section describes an observation that we made during our tests.

### B.9.1 SSL203E RSAD failed

The symptom and possible reason are described.

#### ***Symptom***

Message SSL203E RSAD failed RC=0000002E(RSADNOHC) reason=000004B2 is issued when trying to open an SSL connection.

#### ***Reason***

Return code RSADNOHC indicates “no hardware crypto” when an RSA decryption is performed. Most likely, you used an RSA key with a key length greater than 1024 bits.

TCP/IP for VSE/ESA has a software implementation of the RSA algorithm up to 1024 bits. Greater keys require a crypto card that is installed in your mainframe.

The following cards are supported:

- ▶ PCICA
- ▶ PCIXCC
- ▶ Crypto Express2 (CEX2C and CEX2A)
- ▶ Crypto Express3 (CEX3C and CEX3A)

For more information about cryptographic hardware for System z, see this web page:

<http://www.ibm.com/systems/z/security/cryptography.html>





# Related publications

The publications that are listed in this section are considered particularly suitable for a more detailed discussion of the topics that are covered in this book.

## IBM Redbooks publications

For more information about ordering these publications, see “How to get IBM Redbooks publications” on page 441. Note that some of the documents that are referenced here might be available in softcopy only:

- ▶ *Introduction to the New Mainframe: Security*, SG24-6776
- ▶ *Security on z/VM*, SG24-7471
- ▶ *System z Crypto and TKE Update*, SG24-7848
- ▶ *Encryption Facility for z/OS V1R10*, SG24-7318
- ▶ *Encryption Facility for z/OS V1.2 OpenPGP Support*, SG24-7434
- ▶ *Using MQSeries for VSE*, SG24-5647
- ▶ *WebSphere V5 for Linux on zSeries Connectivity Handbook*, SG24-7042
- ▶ *IBM System Storage Tape Encryption Solutions*, SG24-7320
- ▶ *A Practical Guide to Enterprise Tape Drives and TS3500 Tape Automation*, SG24-6789

## IBM Knowledge Center

IBM Knowledge Center is the home of product documentation, at which the following product documentation, reference material, videos, and many more are available:

- ▶ z/VM 6.4.0:  
[https://www.ibm.com/support/knowledgecenter/SSB27U\\_6.4.0/com.ibm.zvm.v640/zvminfoc03.htm](https://www.ibm.com/support/knowledgecenter/SSB27U_6.4.0/com.ibm.zvm.v640/zvminfoc03.htm)
- ▶ z/VSE:  
[https://www.ibm.com/support/knowledgecenter/SSB27H/zvse\\_welcome.html](https://www.ibm.com/support/knowledgecenter/SSB27H/zvse_welcome.html)
- ▶ CICS Transaction Server for z/VSE 2.1.0:  
[https://www.ibm.com/support/knowledgecenter/en/SSECAB\\_2.1.0/welcome.html](https://www.ibm.com/support/knowledgecenter/en/SSECAB_2.1.0/welcome.html)

## Online resources

The following websites are also relevant as further information sources:

- ▶ z/VSE homepage with links to other documentation:  
<http://www.ibm.com/zvse/>
- ▶ IBM Security Solutions product overview links:  
<http://www.ibm.com/security/products/>
- ▶ Documentation, including papers about security and cryptography on z/VSE homepage:  
<http://www.ibm.com/systems/z/os/zvse/documentation/security.html>

- ▶ CryptoCards, IBM Systems cryptographic hardware products:  
<http://www.ibm.com/security/cryptocards/>
- ▶ Keyman/VSE tool and VSE Connector Client:  
<http://www.ibm.com/systems/z/os/zvse/downloads/>
- ▶ Encryption Facility for z/OS:  
[http://www.ibm.com/systems/z/os/zos/tools/encryption\\_facility/](http://www.ibm.com/systems/z/os/zos/tools/encryption_facility/)
- ▶ KeePass Password Safe (a free Open Source Password Manager):  
<http://keepass.info/>
- ▶ RSA Security (PKCS #5: Password-Based Cryptography Standard):  
<http://www.rsa.com/rsalabs/node.asp?id=2127>
- ▶ PKCS #5: Password-Based Cryptography Specification, Version 2.0:  
<http://tools.ietf.org/html/2898>
- ▶ RFC 4880 OpenPGP Message Format:  
<http://tools.ietf.org/html/rfc4880>
- ▶ The GNU Privacy Guard:  
<http://www.gnupg.org/>
- ▶ TCP/IP for VSE homepage (Vendor: CSI):
  - <http://www.e-vse.com>
  - <http://www.csi-international.com/products/zVSE/TCP-IP/TCP-IP.htm>
- ▶ IPv6/VSE homepage (Vendor: BSI):  
<http://www.bsitcip.com/index.html>
- ▶ Live Virtual Classes on the z/VSE homepage:  
<http://www.ibm.com/systems/z/os/zvse/education/index.html>
- ▶ FileZilla server:  
<http://filezilla-project.org/>
- ▶ Two required files for installing OpenSSL on Windows:  
<http://www.slproweb.com/products/Win320penSSL.html>
- ▶ Active versus passive FTP discussion:  
<http://slacksite.com/other/ftp.html>

## How to get IBM Redbooks publications

You can search for, view, or download Redbooks, Redpapers, Technotes, draft publications, and Additional materials, and order hardcopy Redbooks, at this website:

[ibm.com/redbooks](https://ibm.com/redbooks)

## Help from IBM

IBM Support and downloads

[ibm.com/support](https://ibm.com/support)

IBM Global Services

[ibm.com/services](https://ibm.com/services)



# Index

## Symbols

\$EMAIL 284  
\$EVENT 284  
\$LPD 284  
\$LPR 284  
\$SOCKDBG 218  
\$WEB 216

## A

ACC 48  
access control class (ACC) 48  
Access Control Logging and Reporting (ACLR) 6, 52  
access level 36, 39  
access list 40–41  
ACF/VTAM 2  
ACICSPCT 14  
ACLR 6, 52  
APPL 14  
ASSGN 44  
asymmetric cipher 96  
audit level 36, 39  
audit options 19  
auditing 6  
    commands 20–21, 29  
    DTSECTAB resources 52  
authorization checking 20

## B

BAM 4  
Basic Access Method (BAM) 4  
Basic Security Manager  
    See BSM  
batch processing 2  
batch security 5, 21, 217  
BGINIT 16  
BSM 1, 5–6, 11, 21, 73, 110, 216, 245, 253, 289, 291, 346  
    access list 42  
    add user 23  
    administration 17, 19  
    backups 60  
    concept 12  
    control file 13, 20, 31, 35  
    cross reference report (BSTXREF) 48  
    groups 31–32  
    initialization 22  
    installing and customizing 16  
    logging 15, 21, 50  
    logging status 20  
    migration 62  
    options 19  
    processing 15  
    report writer (BSTRPWTR) 19, 58

    reports 6, 58  
    resource profiles 34  
    status 17  
    summary reports 59–60  
    transactions 362, 377  
BSSDCERT 247, 434  
BSSINIT 16  
BSSTISX 12, 286  
    post-processing exit 290  
    preprocessing exit 290  
BSSTIXE 290  
BSSTXAUT 292  
BSSTXSOF 292  
BSSTXSON 292  
BSTADMIN 18, 62  
    command logging 51  
    command reports 60  
CONNECT 29  
PERFORM 22, 51  
STATUS 18–19  
USERID 22, 29  
BSTCNTL 13, 20  
BSTRPWTR 19, 58  
BSTSAVER 62  
BSTXREF 48

## C

CA 99, 198  
CEMT PERFORM SECURITY 23  
certificate 99, 198, 279–280, 327, 378, 393  
    sending to z/VSE 330  
certificate authority (CA) 99, 198  
CEX2A 7  
CEX2C 7  
CEX3A 7  
CEX3C 7  
CIALCERT 337, 423  
CICS 2, 4, 9, 11–12, 51, 54, 221, 223  
    DMF 6, 52  
    encryption strength 226  
    initialization parameters 226  
    job output 362  
    panel 377  
    partition 61  
    prefixing 36  
    program 256  
    resources 6, 13  
    security functions 279  
    sign-on 73  
SSLDELAY 227  
TCP/IP service 224  
TCPIPSERVICE 278  
temporary storage queue 20  
transaction 34

- transaction security keys 30
- CICS Transaction Server (CICS TS) 2, 11, 221
- CICS Web Support (CWS) 4, 8, 221
- CICS-owned resource 13–14
- cipher
  - definition of 95
  - FTP daemon, current connection 322
  - suites 212, 226
- client authentication 243, 305, 378
- client certificate 244–245, 247, 306
- command logging 20
- common name (CN) 71
- connect user ID 27–28
- connector security 8, 251
- connectors 3
- CPACF 7, 104–105, 145, 416
- CPU Assist Facility (CPACF) 416
- CPU Assist feature 124
- crypto device driver 110
- Crypto Express2 102, 212, 379
- cryptographic
  - hardware 102
  - LPAR configuration 105
- cryptology 93
- CWS 4, 8, 221–222, 224, 278, 416

## D

- Data Management Facility
  - See DMF
- data space 20
  - active 22
  - inactive 22
  - refresh 22
  - size 22
- DCICISDCT 14
- DFHDFOU 57–58
- DFHDFSIP 54
- DFHDMFSP 53
- DFHSIT 14, 16
- DFHSITSP 222, 226
- Directory Information Tree (DIT) 70
- distinguished name (DN) 70
- DITTO 12
- DK 126
- DMF 6, 20, 55
  - data sets 53, 57
  - data space 56
  - flush 57
  - initialization 53–54
  - OS390 environment 56
  - partition 54
  - partition priority 55
  - security server partition 15
  - server 54
  - switch 57
- DMF dump utility DFHDFOU 57
- DMFSTART 54–55
- DS8000 94
- DTSECTAB 15, 56
- DTSECTRC 16

- DTSECTXN 15
- DTSFILE 15

## E

- EEDK 128–129
- EKM 126, 129, 131, 133, 136, 142
- EMAIL client 284
- encryption
  - key, definition of 95
  - technology 93
- Encryption Facility 7, 93, 146
- Encryption Key Manager
  - See EKM
- encryption key or data key (DK) 126
- ESM 9, 12, 73, 253, 286, 292, 344, 346
  - initialization 16
- external security manager
  - See ESM
- externally encrypted data key (EEDK) 128–129

## F

- FACILITY 36, 38, 253–254, 292
- FCICSFCT 14
- File Transfer Protocol
  - See FTP
- firewall 340, 343
- FTP 8, 147, 284, 322, 416
  - access 287–288
  - client 322, 340
  - daemon 321–322
  - ports 340
  - secure 8, 95, 102, 319–320
  - server 335, 341
  - server certificate 328
  - session 285
  - user IDs 287
  - z/VM client 321
- FTPBatch 290, 341

## G

- general options 19
- GENERIC 36
- generic profile 37
- GNU Privacy Assistant 161
- GNU Privacy Guard (GnuPG) 160, 162
- GPG4win 162
- GPGee 161–162
- group definition 30
- GROUP01 30
- GROUP61 30

## H

- Hardware Management Console
  - See HMC
- hardware-based encryption 102
- hash 75, 131
  - function 319, 416
  - mode 128

- size 100
- value 128–129
- HMC 102, 105
- HTTP 226, 273, 278, 284, 287, 296
  - daemon 215–216, 284, 296
- HTTPD 217
- HTTPS 217, 236, 278, 416

## I

- IBM HTTP server 7
- IBM Personal Communications
  - See PCOMM
- ICCF 2, 4, 15, 23, 73
- ICCF DTSFILE 15
- ICCF libraries 4, 25
- IESCNTL 13
- IESIRCVT 21
- IJBCRYPT 110
- independent software vendor
  - See ISV
- Interactive Computing and Control Facility
  - See ICCF
- Interactive Interface 2–3, 21
- intrusion detection 9
- IP address 69
- ISV 9, 12

## J

- Java KeyStore
  - See JKS
- Java-based connector 252–253
- JCICSJCT 14
- JCL 2, 5, 29, 149–150, 298, 320
  - protecting selected operands 44
- JKS 201, 212, 328
- job control language
  - See JCL

## K

- KEK 101, 127, 132
- KEKL 127–128, 131–132
- key encrypting key
  - See KEK
- key encrypting key label
  - See KEKL
- Keyman/VSE 88, 145, 152, 199, 233, 247, 294, 337–338, 378, 430, 433
- keystore 425
  - create with keytool 138
- keytool
  - create keystore 138

## L

- LDAP 1, 7, 23, 70, 80
  - maintain user profile 26–27
  - mapping file 75
  - mapping file administration 83
  - mapping tool 84

- sign-on support 72, 75
- LDAP client 68
  - general interaction 69
- LDAP Data Interchange Format (LDIF) 71
- LDAP server
  - export 71
  - output file 71
- LDIF file 71
- LIBDEF 44
- LIBDROP 44
- Lightweight Directory Access Protocol
  - See LDAP
- logging 6, 11, 21
- logging of administrator accesses 21, 51
- LPD 284
- LPR 284

## M

- MCICSPPT 14
- message 0P68I 144
- message layer security 251, 275
- model user ID 27
- MQ 378, 416
  - BSM transaction profile 347
  - configuration 356
  - installation 348
  - resource classes 14, 346
  - starting 362
- MQADMIN 14
- MQCMDS 14
- MQCONN 14
- MQMT 359, 362
- MQNLIST 14
- MQQUEUE 14
- MQSE 356
- MQSeries 102, 346
- MQSU 346, 348

## N

- native mode 214–215, 296
- Navigator 193, 208

## O

- OLTP 5
- online processing 2
- online security 5, 21
- OpenPGP 162–163
- OS390 environment 54, 56

## P

- PARSTD 44
- pass-through mode 214, 296
- password options 19–20
- password-based encryption (PBE) 148, 164
- PBE 148, 190, 192
- PCICA 7
- PCIXCC 7, 379
- PCOMM 298



PERM 44  
PFX 228, 233, 385  
  files 201, 212, 298, 301, 328, 385  
  format 201  
PGP 169–170  
PGP-based encryption 160  
PKE 101, 151, 188  
PKI 197  
  certificates 275  
port range 342  
POWER 3, 54  
PRIMARY.HTTPS 215–216  
private key 327  
public key 99  
public key infrastructure (PKI) 197  
public-key encryption  
  See PKE

## Q

QT 132, 135  
queue manager 356–357  
  remote 396, 403

## R

RACROUTE 12, 34, 253, 289, 292  
  logging 51  
  REQUEST=VERIFY 51  
Redbooks website 441  
  Contact us xv  
Remote Job Entry (RJE) 3  
resource classes 20  
RESSEC 14  
RJE 3  
RSA 379  
RSA key pair 96, 197, 222

## S

SAF 12, 15  
salt value 101  
SCICSTST 14  
SE 102  
SECSERV 15, 110  
secure FTP 8, 95, 102  
secure hash 75  
Secure Sockets Layer  
  See SSL  
secure Telnet 102  
security server 15, 61, 110  
  OS390 environment 15, 56  
security server partition 15–16  
security system settings 17  
self-signed certificates  
  definition of 198  
server authentication 378  
server certificate 380  
SIR command 17  
SITE command 289  
SKDMFST 54

SMF 52–53  
  80 records 52, 58  
  records 57–58  
SOAP 275, 277  
software-based encryption 145, 160  
SSL 72, 93, 197, 214, 226, 256, 296, 319, 378, 393  
  application 414  
  cipher suite 407  
  cipher suites 98, 218, 226, 242  
  function 415  
  licensed 390  
  native mode 214–215, 296  
  pass-through mode 214, 296  
  properties file, description of 209  
  session 236  
STDLABEL 44  
Support Element (SE) 102, 105  
SURROGAT 14  
SXBLOK 288–289  
symmetric cipher 95  
System Management Facility  
  See SMF  
System z  
  cryptographic solution 7  
  LPAR capabilities 1  
  operating systems 7

## T

tape encryption  
  hardware-based 126  
TCICSTRN 14  
TCP/IP 1–2, 12, 251, 321  
  security 8, 283  
  security exit 288  
  trace 218  
TCPIP service 224  
Telnet 293, 416  
  secure 102, 294  
Telnet 3270 293  
TLS 72, 251, 320  
  application 414  
  daemon 214, 296  
  function 415  
TLSD 215, 297  
Transport Layer Security  
  See TLS  
TS1120 7, 126, 130  
  Installation 134  
TS1130 7, 130

## U

UACC 36, 38–39  
universal access authority  
  See UACC  
user definition 23  
USERBG.PROC 21, 55  
USERID command 21

## V

Virtual Storage Access Method

*See* VSAM

Visual Basic 256, 260

VSAM 4, 52

    access 253

    backups 61, 131

    catalog 4

    cluster 73, 83, 149

    files 4, 13, 145, 252, 398

    programs 8

    restore 62

    space 4

VSE control file 13, 60

VSE/POWER 3, 54

    spool files 289

vsftpd server 335

## W

WebSphere MQ 11, 14, 346

## X

X.500 69

XDCT 14

XJCT 14

XML 273, 275

XPCT 14

XPPT 14

XTRAN 14

XTST 14

XUSER 14

## Z

z/VM LDAP server 72

z/VSE

    central functions 2

    connectors 3

    files 6, 34

    Interactive Computing and Control Facility (ICCF) 3

    Interactive Interface 3

    introduction 2

    libraries 6, 34

    members 6, 34

    sublibraries 6, 34

z/VSE Connector Client 3, 208, 252

z/VSE Connector Server 4, 199, 222, 256, 258, 294, 320

    level of security 254

z/VSE Navigator 193, 208

z/VSE script connector 257



**Redbooks**

**Security on IBM z/VSE**

SG24-7691-03

ISBN 0738456918



(0.5" spine)

0.475" x 0.873"

250 <-> 459 pages







SG24-7691-03

ISBN 0738456918

Printed in U.S.A.

Get connected

