IBM SecureWay Product
IBM 4758 PCI Cryptographic Coprocessor

PKCS #11 Support Program Installation Manual
for IBM 4758 Models 002 & 023 Release 2.3.1.0

21-MAY-01, 13:54

Note!

Before using this information and the product it supports, be sure to read the general information printed under Appendix E, “Notices” on page E-1.

Third Edition (February, 2001)

IBM does not stock publications at the address given below. This and other publications related to the IBM 4758 Coprocessor can be obtained in PDF format from the Library page at *http://www.ibm.com/security/cryptocards*.

Reader’s comments can be communicated to IBM by using the Comments and Questions Form located on the product Web site at *http://www.ibm.com/security/cryptocards*, or you can respond by mail to:

Department VM9A, MG81/204-3
IBM Corporation
8501 IBM Drive
Charlotte, NC 28262-8563
U.S.A.

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

Contents

About this Publication v
Audience v
Prerequisite Knowledge v
Organization of this Book vi
Typographic Conventions vi
Related Publications vi
General Interest vii
CCA Support Program Publications vii
Custom Software Publications vii
Cryptography Publications vii
Other IBM Cryptographic Product Publications ix
Summary of Changes x

Chapter 1. Installation Process Overview 1-1
Summary 1-1

Chapter 2. Obtaining Coprocessor Hardware and Software 2-1
Choosing Product Features 2-1
Ordering and Obtaining the IBM 4758 Hardware 2-2
Installing Your IBM 4758 Hardware 2-3
Downloading and Deciphering the Software 2-3

Chapter 3. Installing the Support Program 3-1
Support Program Components 3-1
Installing and Removing Host Software 3-1
Installing and Removing the Support Program for Windows NT and Windows 2000 3-2
Windows NT and Windows 2000 Requirements 3-2
Installing the Support Program 3-2
Removing the Support Program 3-3
Installing and Removing the Support Program for AIX 3-4
AIX Requirements 3-4
Installing the Support Program 3-4
Locating RS/6000 Coprocessor Hardware Errors 3-5
Removing the Support Program 3-5

Chapter 4. Loading Software into the Coprocessor 4-1
Loading Coprocessor Software 4-1
Validating the Coprocessor Segment Contents 4-5
How to Unload Coprocessor Software and Zeroize the PKCS #11 Node 4-5
Directories and Files 4-6

Chapter 5. Building Applications on Windows NT and Windows 2000 That Use the PKCS #11 API 5-1
Overview 5-1
PKCS #11 Directories 5-1
PKCS #11 Support Program API Extensions 5-1
Outbound Authentication 5-1
Compiling and Linking Application Programs 5-3

Appendix A. Overview of the PKCS #11 Application Download Process . A-1

Appendix B. Using CLU B-1

Appendix C. Device Driver Error Codes C-1

Appendix D. The IBM Root Public Key D-1

Appendix E. Notices E-1
License E-1
Copying and Distributing Softcopy Files E-2
Trademarks E-2

List of Abbreviations and Acronyms X-1

Glossary X-3

Index X-7

About this Publication

This installation manual describes the PKCS #11 Support Program feature Release 2.3.1.0 for the IBM® 4758 PCI Cryptographic Coprocessor Models 002 and 023. This feature includes device drivers, utilities, and the IBM PKCS #11 application and host files.

The feature is designed to be used with an AIX®, Windows NT®, or Windows 2000 operating system.

Use this manual to help with the following tasks:

- Obtain the support program through the IBM 4758 Web site *http://www.ibm.com/security/cryptocards*
- Load the software onto a host computer and into the coprocessor
- Use the utilities supplied with the support program to
  - Load the coprocessor function control vector
  - Intialize the coprocessor
- Link your application software to the PKCS #11 libraries

Audience

The audience for this publication includes:

- System administrators who install the software
- System and application programmers who determine how the software is to be used

Prerequisite Knowledge

Before you use this publication, familiarize yourself with the contents of the *IBM 4758 PCI Cryptographic Coprocessor General Information Manual*, located on the Library page of the *http://www.ibm.com/security/cryptocards* Web site.

Organization of this Book

- Chapter 1, "Installation Process Overview" summarizes the installation and the operation of the PKCS #11 Support Program.
- Chapter 2, "Obtaining Coprocessor Hardware and Software" describes how to order and obtain the PKCS #11 Support Program.
- Chapter 3, "Installing the Support Program" describes how to install the device drivers, utilities, and PKCS #11 host files onto the host computer.
- Chapter 4, "Loading Software into the Coprocessor" describes how to load the operating system and the PKCS #11 application into the cryptographic coprocessor.
- Chapter 5, "Building Applications on Windows NT and Windows 2000 That Use the PKCS #11 API" explains how to build applications that use PKCS #11 services, and how to link them to the PKCS #11 library.
- Appendix A, "Overview of the PKCS #11 Application Download Process" lists the steps a developer needs to perform during development of a PKCS #11 application.
- Appendix B, "Using CLU" describes the use of the Coprocessor Load Utility (CLU).
- Appendix D, "The IBM Root Public Key" lists the public exponent and modulus (in hex) for the public root key.
- Appendix E, "Notices" includes product and publication notices.
- A list of abbreviations, a glossary, and an index complete the manual.

Typographic Conventions

This publication uses the following typographic conventions:

- Commands that you enter verbatim onto the command line are presented in **bold** type.
- Variable information and parameters that you enter within commands, such as file names, are presented in *italic* type.
- The names of items that are displayed in graphical user interface (GUI) applications—such as pull-down-menu names, check boxes, radio buttons, and fields—are presented in **bold** type.
- Items displayed within the pull-down menus are presented in ***bold italic*** type.
- System responses in a non-GUI environment are presented in `monospace` type.
- Web addresses and file directory-locations are presented in *italic* type.

Related Publications

Check the Library page of the IBM 4758 Web site at *http://www.ibm.com/security/cryptocards* for the availability of these publications. From the Web site, you can download, view, and print publications available in the Adobe Acrobat portable document format (PDF).

General Interest

- *IBM 4758 PCI Cryptographic Coprocessor General Information Manual*
- *IBM 4758 PCI Cryptographic Coprocessor Installation Manual*

CCA Support Program Publications

- *IBM 4758 PCI Cryptographic Coprocessor CCA Support Program Installation Manual*
- *IBM 4758 CCA Basic Services Reference and Guide*

Custom Software Publications

- *IBM 4758 PCI Cryptographic Coprocessor Custom Software Developer's Toolkit Guide*
- *IBM 4758 PCI Cryptographic Coprocessor Custom Software Installation Manual*
- *IBM 4758 PCI Cryptographic Coprocessor Custom Software Interface Reference*
- *IBM 4758 PCI Cryptographic Coprocessor ICAT User's Guide*
- *IBM 4758 PCI Cryptographic Coprocessor CP/Q Operating System Overview*
- *IBM 4758 PCI Cryptographic Coprocessor CP/Q Operating System Application Programming Reference*
- *IBM 4758 PCI Cryptographic Coprocessor CP/Q Operating System C Runtime Library Reference*
- *IBM 4758 PCI Cryptographic Coprocessor CCA User Defined Extensions Programming Reference*
- *AMCC S5933 PCI Controller Data Book*, available from Applied Micro Circuits Corporation, 6290 Sequence Drive, San Diego, CA 92121-4358. Phone 1-800-755-2622 or 1-619-450-9333. The manual is available online as an Adobe Acrobat PDF file at *http://www.amcc.com/pdfs/pciprod.pdf*.

Cryptography Publications

The following publications describe cryptographic standards, research, and practices applicable to the PCI Cryptographic Coprocessor:

- "Application Support Architecture for a High-Performance, Programmable Secure Coprocessor," J. Dyer, R. Perez, S.W. Smith, and M. Lindemann, 22nd National Information Systems Security Conference, October 1999.
- "Validating a High-Performance, Programmable Secure Coprocessor," S.W. Smith, R. Perez, S.H. Weingart, and V. Austel, 22nd National Information Systems Security Conference, October 1999.
- "Building a High-Performance, Programmable Secure Coprocessor," S.W. Smith and S.H. Weingart, Research Report RC21102, IBM T.J. Watson Research Center, February 1998.
- "Using a High-Performance, Programmable Secure Coprocessor, S.W. Smith, E.R. Palmer, and S.H. Weingart, in *FC98: Proceedings of the Second International Conference on Financial Cryptography*, Anguilla, February 1998. Springer-Verlag LNCS. 1998. ISBN 3-540-64951-4

- “Smart Cards in Hostile Environments,” H. Gobioff, S.W. Smith, J.D. Tygar, and B.S. Yee, *Proceedings of the Second USENIX Workshop on Electronic Commerce*, 1996
- “Secure Coprocessing Research and Application Issues,” S.W. Smith, Los Alamos Unclassified Release LA-UR-96-2805, Los Alamos National Laboratory, August 1996.
- “Secure Coprocessing in Electronic Commerce Applications,” B.S. Yee and J.D. Tygar, in *Proceedings of the First USENIX Workshop on Electronic Commerce*, New York, July 1995.
- “Transaction Security Systems,” D.G. Abraham, G.M. Dolan, G.P. Double, and J.V. Stevens, in *IBM Systems Journal* Vol. 30 No. 2, 1991, G321-0103.
- “Trusting Trusted Hardware: Towards a Formal Model for Programmable Secure Coprocessors,” S.W. Smith and V. Austel, in *Proceedings of the Third USENIX Workshop on Electronic Commerce*, Boston, August 1998.
- “Using Secure Coprocessors,” B.S. Yee (Ph.D. Thesis), Computer Science Technical Report CMU-CS-94-149, Carnegie-Mellon University, May 1994.
- “Cryptography: It’s Not Just for Electronic Mail Anymore,” J.D. Tygar and B.S. Yee, Computer Science Technical Report, CMU-CS-93-107, Carnegie Mellon University, 1993.
- “Dyad: A System for Using Physically Secure Coprocessors,” J.D. Tygar and B.S. Yee, Harvard-MIT Workshop on Protection of Intellectual Property, April 1993.
- “An Introduction to Citadel—A Secure Crypto Coprocessor for Workstations,” E.R. Palmer, Research Report RC18373, IBM T.J. Watson Research Center, 1992.
- “Introduction to the Citadel Architecture: Security in Physically Exposed Environments,” S.R. White, S.H. Weingart, W.C. Arnold, and E.R. Palmer, Research Report RC16672, IBM T.J. Watson Research Center, 1991.
- “An Evaluation System for the Physical Security of Computing Systems,” S.H. Weingart, S.R. White, W.C. Arnold, and G.P. Double, Sixth Computer Security Applications Conference, 1990.
- “ABYSS: A Trusted Architecture for Software Protection,” S.R. White and L. Comerford, IEEE Security and Privacy, Oakland 1987.
- “Physical Security for the microABYSS System,” S.H. Weingart, IEEE Security and Privacy, Oakland 1987.
- *Applied Cryptography: Protocols, Algorithms, and Source Code in C, Second Edition*, Bruce Schneier, John Wiley & Sons, Inc. ISBN 0-471-12845-7 or ISBN 0-471-11709-9
- *ANSI X9.31 Public Key Cryptography Using Reversible Algorithms for the Financial Services Industry*
- *IBM Systems Journal* Volume 30 Number 2, 1991, G321-0103
- *IBM Systems Journal* Volume 32 Number 3, 1993, G321-5521
- *IBM Journal of Research and Development*, Volume 38 Number 2, 1994, G322-0191
- *USA Federal Information Processing Standard (FIPS):*

  - *Data Encryption Standard,* 46-1-1988
  - *Secure Hash Algorithm,* 180-1, May 31, 1994
  - *Cryptographic Module Security,* 140-1
- *Derived Test Requirements for FIPS PUB 140-1*, W. Havener, R. Medlock, L. Mitchell, and R. Walcott. MITRE Corporation, March 1995.
- *ISO 9796 Digital Signal Standard*
- *Internet Engineering Taskforce RFC 1321*, April 1992, MD5
- *Secure Electronic Transaction Protocol*, Version 1.0, May 31, 1997

IBM Research Reports can be obtained from:

IBM T.J. Watson Research Center
Publications Office, 16-220
P.O. Box 218
Yorktown Heights, NY 10598

Back issues of the *IBM Systems Journal* and the *IBM Journal of Research and Development* may be ordered by calling (914) 945-3836.

Other IBM Cryptographic Product Publications

The following publications describe products that utilize the IBM Common Cryptographic Architecture (CCA) application program interface (API).

- *IBM Transaction Security System General Information Manual*, GA34-2137
- *IBM Transaction Security System Basic CCA Cryptographic Services*, SA34-2362
- *IBM Transaction Security System I/O Programming Guide*, SA34-2363
- *IBM Transaction Security System Finance Industry CCA Cryptographic Programming*, SA34-2364
- *IBM Transaction Security System Workstation Cryptographic Support Installation and I/O Guide*, GC31-4509
- *IBM 4755 Cryptographic Adapter Installation Instructions*, GC31-4503
- *IBM Transaction Security System Physical Planning Manual*, GC31-4505

- *IBM Common Cryptographic Architecture Services/400 Installation and Operators Guide, Version 2*, SC41-0102
- *IBM Common Cryptographic Architecture Services/400 Installation and Operators Guide, Version 3*, SC41-0102
- *IBM ICSF/MVS General Information*, GC23-0093
- *IBM ICSF/MVS Application Programmer's Guide*, SC23-0098

Summary of Changes

Changes made to the second edition in October, 2000 include:

- Chapter 3—Updated the Support Program removal instructions.

Changes made to the third edition in December, 2000 include:

- Chapter 5—Added Outbound Authentication (OA) Manager API and updated subdirectories for compiling applications.
- Added support for the Windows 2000 operating system throughout the manual.

Changes made to the third edition in February, 2001 include:

- Chapter 3—Added Windows 2000 device driver information.

Chapter 1. Installation Process Overview

This chapter summarizes the installation procedures discussed in this manual and provides a checklist (see Table 1-1 on page 1-2) for you to use while installing the PCI Cryptographic Coprocessor and the PKCS #11 Support Program.

Summary

The PKCS #11 Support Program includes the following:

- The coprocessor operating system and the PKCS #11 application, which run on the coprocessor and provide support for the PKCS #11 application program interface (API).
- Device drivers and utility programs that run on the host in which the coprocessor is installed. These allow the host to interact with the coprocessor in order to load and configure the PKCS #11 application and request PKCS #11 services.

To obtain and install these components perform the following steps, which are described in this manual:

1. **Obtain the hardware and software**: Chapter 2, “Obtaining Coprocessor Hardware and Software” describes how to order the software from IBM, how to download it from the IBM 4758 Web site, and how to unpack the downloaded files.

   Note: Concurrent with the availability of support on the RS/6000® platform with AIX, IBM has updated the PKCS #11 Support Program distribution procedure. You no longer need to obtain license keys, but instead are guided through a registration process prior to downloading the software. This process is described in Chapter 2, “Obtaining Coprocessor Hardware and Software” on page 2-1.

2. **Install the host software**: Chapter 3, “Installing the Support Program” describes how to install the software onto the host in which the coprocessor is installed.

3. **Load the coprocessor software**: Chapter 4, “Loading Software into the Coprocessor” describes how to load into the coprocessor the CP/Q^{++} embedded operating system and the PKCS #11 application.

4. **Build applications to use with the PKCS #11 API**: Chapter 5, “Building Applications on Windows NT and Windows 2000 That Use the PKCS #11 API” describes how to build applications that use PKCS #11 servers and how to link them to the PKCS #11 library.

Table 1-1. Activity Checklist, PKCS #11 Support Program Installation

Step	Task	Reference	√
1	Decide which platform support package is appropriate to your setup: Platform: AIX ( ) Windows NT ( ) Windows 2000 ( )	"Choosing Product Features" on page 2-1	
2	Place an order with IBM or your IBM Business Partner. (OEM sales are processed by the IBM OEM Sales office.)	"Ordering and Obtaining the IBM 4758 Hardware" on page 2-2	
3	Receive the coprocessor hardware.		
4	Install the coprocessor hardware.	"Installing Your IBM 4758 Hardware" on page 2-3	
5	Download the support program.	"Downloading and Deciphering the Software" on page 2-3	
6	Install the software onto the host in which the coprocessor is installed.	Chapter 3, "Installing the Support Program"	
7	Load the coprocessor software.	Chapter 4, "Loading Software into the Coprocessor"	
9	Build a custom application to use with the PKCS #11 API (if desired).	Chapter 5, "Building Applications on Windows NT and Windows 2000 That Use the PKCS #11 API" on page 5-1	

Chapter 2. Obtaining Coprocessor Hardware and Software

The PKCS #11 Support Program feature is available for download from the Order page of the IBM 4758 Web site at *http://www.ibm.com/security/cryptocards*. The software is enciphered; to use it, you must place an order and then obtain the license keys necessary to decipher the support program files. This chapter describes how to:

- Choose the product features you need
- Order the hardware and software
- Download the software

Choosing Product Features

The coprocessor is manufactured in several models, each with different capabilities. Models 002 and 023 incorporate triple-DES and faster hardware than the earlier Models 001 and 013. Only Models 002 and 023 operate with the version 2 software. Model 002 includes advanced physical penetration detection and the product has been certified under FIPS 140-1 at level 4. Model 023 incorporates a different approach to physical penetration detection but is in other respects the same as the Model 002. Model 023 is certified under FIPS 140-1 at level 3.

Models 002 and 023 can be installed in personal computer PCI slots that accept a full-height, two-thirds-length, PCI board.

RS/6000 users can order IBM 4758 Model 023-class technology using feature codes as indicated in Table 2-1 on page 2-2. IBM 4758 Model 002 or Model 023 coprocessors may not be installed in IBM RS/6000 systems.

Review the Tested Systems page of the IBM 4758 Web site for server systems in which the IBM 4758 technology has been tested.

The battery kit contains two batteries and a temporary-battery tray. The shelf life of the batteries is nearly the same as the useful life of batteries mounted in an IBM 4758 that is continuously powered on. As a general guideline, plan to change the batteries every three to five years. The actual life of the batteries is anticipated to be in excess of five years. When you do change batteries, be sure that the replacements are fresh and have not been in inventory for a long period.

From the following table choose the coprocessor and (optional) battery kit:

Table 2-1. Order Numbers

Description	**Machine Type**	**Model No.**	**Feature Code**
Personal Computer and Netfinity Servers (You order an IBM 4758 machine type.)			
PCI Cryptographic Coprocessor FIPS 140-1, level 4, triple DES	4758	002	
PCI Cryptographic Coprocessor FIPS 140-1, level 3, triple DES	4758	023	
Replacement Battery Kit			1008
RS/6000 Systems (You order a feature for the indicated IBM RS/6000 machine type.)			
RS/6000 PCI Cryptographic Coprocessor	7025	F50 F80 H70	4958
RS/6000 PCI Cryptographic Coprocessor	7026	H50 H70 H80 M80	4958
RS/6000 PCI Cryptographic Coprocessor	7043	270	4958
RS/6000 PCI Cryptographic Coprocessor	7044	170 270	4958
RS/6000-SP PCI Cryptographic Coprocessor	9076	Refer to the IBM 4758 Web site.*	Refer to the IBM 4758 Web site.*
RS/6000-SP PCI Cryptographic Coprocessor (MES only)	9076	Refer to the IBM 4758 Web site.*	Refer to the IBM 4758 Web site.*
Note: * Current machines and feature codes can be found on the IBM 4758 Web site *http://www.ibm.com/security/cryptocards*.			

Ordering and Obtaining the IBM 4758 Hardware

To order the coprocessor hardware, contact your local IBM Representative or your IBM Business Partner, and order the models and features you have chosen.

Customers in the U.S.A. can contact IBM Direct at 1-800-IBM-CALL. Specifically mention “IBM 4758” so that you can discuss your order with the group that processes IBM 4758 orders.

Installing Your IBM 4758 Hardware

The IBM 4758 is installed in a manner similar to other PCI boards.

- Personal computer users should follow the process described in the *IBM 4758 PCI Cryptographic Coprocessor Installation Manual*.
- RS/6000 users should follow the process described in the *PCI Cryptographic Installation and Using Guide*. Note that the order of installation between hardware and the device driver is important in an AIX installation.

Important

Be certain that you never remove the coprocessor batteries except as outlined in the battery replacement procedure in the *IBM 4758 PCI Cryptographic Coprocessor Installation Manual*. The coprocessor is certified at the factory. If it ever detects tampering, or if battery power and system power are removed, the factory certification will be zeroized and the coprocessor will be rendered non-functional. There is no recovery from this situation.

If in handling the coprocessor you inadvertently cause a short circuit in the circuitry, a tamper event may occur. This is very unlikely, but be cautious when installing the coprocessor to keep the circuitry from contacting conductive portions of the host machine or adjacent boards.

Downloading and Deciphering the Software

To be sure you receive the latest version of the support program, wait until you have received and installed your coprocessor before you download the support program from the IBM 4758 Web site. At that time you should also check the Web site for any available fix packs.

Download the operating system feature you ordered from the Order page of the *http://www.ibm.com/security/cryptocards* Web site.

If you plan to use the support program on multiple host computers, you can copy the install images or the executable file to the other hosts.

Now you can install the support program; continue to Chapter 3, “Installing the Support Program.”

Chapter 3. Installing the Support Program

After downloading and deciphering the software as described in Chapter 2, "Obtaining Coprocessor Hardware and Software," follow the procedures in this chapter to install the PKCS #11 Support Program onto the host computer in which the coprocessor resides.

This chapter:

- Lists the support program components you are installing
- Lists system prerequisites for installing the software
- Describes how to install the software
- Describes how to uninstall the software

Support Program Components

The procedures in this chapter install the following support program components onto the host computer:

- The IBM 4758 PCI Cryptographic Coprocessor device drivers and related files
- Dynamic Load Libraries (DLLs) that allow an application on the host to use the PKCS #11 API
- The utilities and data files needed to load the CP/Q^{++} operating system and the PKCS #11 application into the coprocessor

Installing and Removing Host Software

For each operating system, the following sections:

- List the hardware and software requirements for the support program
- Describe how to install the support program
- Describe how to remove the support program

After you have installed the software as described in this chapter, you are ready to install software into the coprocessor; see Chapter 4, "Loading Software into the Coprocessor."

Installing and Removing the Support Program for Windows NT and Windows 2000

Windows NT and Windows 2000 Requirements

Before you install the support program, make sure your system meets the following requirements:

Hardware

An IBM-compatible PC with an IBM 4758 PCI Cryptographic Coprocessor installed. During installation of the software, the driver interacts with the coprocessor to arbitrate interrupt settings, DMA channels, and other system resources. For installation instructions regarding the coprocessor hardware, refer to the *IBM 4758 PCI Cryptographic Coprocessor Installation Manual*.

Software

Windows NT Version 4.0 or Windows 2000

Disk Space

Approximately 2 MB

Installing the Support Program

Important

The installation process modifies the system registry; it must be performed by a user with administrator privilege.

The PKCS #11 Support Program is shipped as an InstallShield package file. Run the file to install the PKCS #11 Support Program.

The installation process places the device drivers (cryptont.sys and cryptw2k.sys) in *%SystemRoot%/system32/drivers* and the DLLs that support communication with the card (cryptont.dll and cryptmsg.dll) in *%SystemRoot%/system32*. Details of the entries the installation process creates or modifies in the system registry can be found in the *IBM 4758 PCI Cryptographic Coprocessor Custom Software Interface Reference*. If you are running Windows 2000, switch to the *%SystemRoot%/system32/drivers* directory, delete cryptont.sys, and rename cryptw2k.sys to cryptont.sys. The host must be rebooted to complete the installation of the device driver.

The installation process places the Coprocessor Load Utility (CSUNCLU.EXE) in the *pkcs11/bin/nt* directory. CSUNCLU.EXE should be copied to a directory that is part of the PATH environment variable or the *pkcs11/bin/nt* directory should be added to the PATH environment variable.

The installation process places the PKCS #11 host files in the following directories:

pkcs11/bin/nt
pkcs11/lib/nt/msvcmasm
pkcs11/lib/nt/vacppmsm
pkcs11/include
pkcs11/src

The installation process places the DLL applications on the host used to obtain PKCS #11 services (CRYPTOKI.DLL) in the *pkcs11/bin/nt* directory; it should be moved to a directory that appears in the PATH environment variable if *pkcs11/bin/nt* is not in the PATH environment variable.

The other PKCS #11 host files are used to build custom applications that invoke PKCS #11 services. See Chapter 5, “Building Applications on Windows NT and Windows 2000 That Use the PKCS #11 API” on page 5-1 for details.

Removing the Support Program

The support program can be removed using the "Add/Remove Programs" utility in the Control Panel.

To remove the support program manually:

1. Remove *%SystemRoot%/system32/cryptont.dll*.
2. Remove *%SystemRoot%/system32/cryptmsg.dll*.
3. Remove *%SystemRoot%/system32/drivers/cryptont.sys*.
4. Delete the *pkcs11* subtree created when the Support Program was installed.
5. From the system prompt, enter **REGEDT32**; the Registry Editor starts.
6. Select the HKEY_LOCAL_MACHINE window.
7. Open the **System** folder.
8. Open the **CurrentControlSet** folder.
9. Open the **Services** folder.
10. Highlight the **cryptont** entry.
11. From the **Edit** menu, select ***Delete***; the **cryptont** entry is removed.
12. Open the **EventLog** folder.
13. Highlight the **cryptont** entry.
14. From the **Edit** menu, select ***Delete***; the **cryptont** entry is removed.
15. Close the Registry Editor.
16. Open the **Settings/Control Panel** folder and select **Add/Remove Programs**.
17. Select **PKCS11** from the list of applications.
18. Click **Add/Remove...**
19. After the removal process completes, close the Add/Remove Programs window. The removal process is complete.

Installing and Removing the Support Program for AIX

Important

The installation process requires root-level authority; it must be performed by a system administrator with that authority.

AIX Requirements

Before you install the support program, make sure your system meets the following requirements:

Hardware

An RS/6000 computer with an IBM 4758 PCI Cryptographic Coprocessor Models 002 or 023 installed. During installation of the software, the driver interacts with the coprocessor to arbitrate interrupt settings, DMA channels, and other system resources. For installation instructions regarding the coprocessor hardware, refer to the *IBM 4758 PCI Cryptographic Coprocessor Installation Manual*.

Software

AIX Version 4.3.3 or 5L (32-bit mode only)

Disk Space

4 MB in the */usr* file system.

Installing the Support Program

To install the support program:

1. Log on as **root**.
2. Enter the command **smitty cfgmgr**; you are prompted to enter the location of the software to be loaded.
3. Enter the location of the install images you obtained using the procedure described in "Downloading and Deciphering the Software" on page 2-3; the software is installed.
4. Press **F10** to exit **smitty**.
5. To confirm successful installation of the driver, enter the command **lsdev -C -l crypt0**; the system message should reflect status `Available`.
6. Enter the command **smitty install_latest**.
7. Enter the location of the install images you obtained using the procedure described in "Downloading and Deciphering the Software" on page 2-3;, the software is installed.
8. When requested, enter **csuf** as the package name.

   You may want to read or print */usr/lpp/csuf/README*;. This file contains current information about the support program.

   There are two install packages:

csuf.com	Common files across CCA and PKCS #11
csuf.pkcs11	PKCS #11 files

| **Note:** No additional configuration is needed.

Locating RS/6000 Coprocessor Hardware Errors

Errors occurring in the coprocessor hardware are placed in the AIX error log. To process and view the log, enter the command

```
errpt -a -N crypt0,libscc.a | more
```

Removing the Support Program

If your key storage files are located in the default directories, back up or save them before you remove the support program; removing the software deletes those key storage files located in the default directories.

To remove the support program:

1. Log on as **root**.
2. Enter the command **rmdev -dl crypt0**; the coprocessor device driver and related information are removed.
3. Enter the command **smitty install_remove**; you are prompted to enter the product names.
4. | Enter the product names **csuf.com**, **csuf.pkcs11**, and **devices.pci.14109f00.rte**.
5. Verify that the “REMOVE dependent software” value is **NO**. Also, verify that the “Preview Only” value is **NO**.
6. Press **Enter**.

Chapter 4. Loading Software into the Coprocessor

After installing the support program onto the host computer (as described in Chapter 3, “Installing the Support Program”) use the Coprocessor Load Utility (CLU) to load the operating system and the PKCS #11 application into the coprocessor.

If you obtain updates to the support program, use the CLU to reload the necessary program segments; you can also load software from other vendors using the CLU.

This chapter includes instructions for using the CLU to install and uninstall the software that runs within the coprocessor. Appendix B, “Using CLU” on page B-1 contains a detailed reference on the use of the CLU..

For an in-depth description of the code loading controls and the security considerations the coprocessor implements, refer to the research paper *Building a High-Performance Programmable, Secure Coprocessor* that is available on the IBM 4758 Web site Library page at *http://www.ibm.com/security/cryptocards*.

Loading Coprocessor Software

This section describes how to load the PKCS #11 application and the operating system into the coprocessor. In particular, it describes how to:

- Determine which software is currently loaded into the coprocessor
- Determine which CLU files need to be loaded into the coprocessor
- Download the requisite CLU files into the coprocessor

The discussion in this section assumes that the Coprocessor Load Utility (CSUNCLU.EXE) is in a directory that is part of the PATH environment variable.

| **Note:** On the the AIX operating system, the command for CLU is CSUFCLU.

Memory on the coprocessor is partitioned into three segments. Each segment has a state that determines what information is associated with the segment and how that information may be changed. The information associated with a segment usually includes a public key. In general, the coprocessor allows changes to be made to a segment’s state and associated information only if the command to make the changes has been signed by the corresponding private key.

To use the PKCS #11 application, segment 1 must contain basic hardware diagnostic routines and the software mechanisms that ensure nothing is loaded into the coprocessor without proper authorization. Segment 2 must hold the coprocessor’s embedded operating system (CP/Q^{++}). And segment 3 must contain the PKCS #11 application itself. The remainder of this section describes how to use CLU to achieve this goal.

Determine Which Software is in the Coprocessor

To determine the current state and content of each segment, use CLU’s ST command, for example:

```
csunclu \logfile-directory\41-00049.log ST
```

| **Note:** On the AIX operating system, the command for CLU is CSUFCLU.

Figure 4-1 on page 4-2 shows a typical response. The items in bold type are of particular interest and are discussed following the figure.

```
======================================================================
CSUNCLU V2.30 E0096.log ST 0    begun Fri Feb 9 09:52:16 2001

*********** Command ST started. ---- Fri Feb  9 09:52:16 2001

 *** VPD data;  PartNum = 04K9128
 *** VPD data;  EC Num = F72272A
 *** VPD data;  Ser Num = 41-E0096
 *** VPD data;  Description = IBM 4758-002 3.3V FIPS 140 LVL 4
 *** VPD data;  Mfg. Loc. = IBM041
 *** VPD data;  Flags = 2400300020000000
 *** ROM Status; PIC ver: 2100, ROM ver: 1102
 *** ROM Status; INIT: INITIALIZED
 *** ROM Status; SEG2: RUNNABLE , OWNER2: 2
 *** ROM Status; SEG3: RUNNABLE , OWNER3: 14
 *** Page 1 Certified: YES
 *** Segment 1 Image: CCA 2.31 & PKCS#11 SEGMENT-1 2000030610552310D00022000000000000000000000000
 *** Segment 1 Revision: 231
 *** Segment 1 Hash: 1BDF 675F F8C5 B38D 574D EAB7 4542 4523 F9A9 BF27
 *** Segment 2 Image: CCA 2.31 & PKCS#11 SEGMENT-2        2001020608492311D000000000000000000000000000
 *** Segment 2 Revision: 231
 *** Segment 3 Image PKCS #11 Application  200102081315          00000000000000000000000000000000
 *** Segment 3 Revision: 1
 *** Segment 3 Hash: 5602 76CE AB69 EC15 1251 BC55 4996 0D6C 3B9E D9AA
 *** Query Adapter Status successful ***
 Obtain Status ended successfully!
 *********** Command ST ended. ---- Fri Feb  9 09:53:33 2001

 *********** Command ST exited. ---- Fri Feb  9 09:53:38 2001
```

Figure 4-1. Typical CLU Status Response

Ser Num

The serial number of the coprocessor (for example, 41-00049).

Description

A statement that describes the coprocessor. Auditors should review this and other status information to confirm that an appropriate coprocessor is in use.

ROM Status

The coprocessor must always be in an INITIALIZED state. If the status is ZEROIZED, the coprocessor has detected a possible tamper event and is in an unrecoverable, non-functional state. (Unintended “tamper” events can be caused by improper handling of the coprocessor. Only remove the batteries when following the recommended battery changing procedure, maintain the coprocessor in the safe temperature range, and so on. Refer to the *IBM 4758 PCI Cryptographic Coprocessor Installation Manual*.)

ROM Status SEG2 / SEG3

Several status conditions for segment 2 and segment 3 exist including:

- UNOWNED: currently not in use, no content
- RUNNABLE: contains code and is in a generally usable state.

Owner identifiers are also shown. The PKCS #11 application is assigned identifier 2 for segment 2 and identifier 14 for segment 3. **Any other code identifier** indicates that the software is not the PKCS #11 product code. In all cases, be certain that the proper software is loaded in your coprocessor. Unauthorized or unknown software can represent a security risk.

Segment 1 Image

The name and description of the software loaded in segment 1. For a factory-fresh coprocessor, the name will include “FACTORY.” This image and associated validation key will need to be changed.

For a previously initialized coprocessor, the name will probably include “CCA” or “PKCS#11.” Be sure to observe the revision level.

Segment 2 and 3 Images

If these segments are RUNNABLE, observe the image name and the revision level. “PKCS11” in the image name means that the contents have been provided as part of the PKCS #11 Support Program. Be sure to observe the revision level.

Determine Which CLU Files to Load

The steps to take to load the PKCS #11 application and other required software into the coprocessor depend on the status and contents of the various segments, as follows.

Segment 1 State

If CLU’s ST command does not indicate segment 1 is in the INITIALIZED state or if page 1 is not certified, the PKCS #11 application cannot be loaded into the coprocessor without additional assistance from IBM.

If segment 1 is INITIALIZED and page 1 is certified, the states of segments 2 and 3 dictate how to proceed:

1. **Case 1 - Segment 2 UNOWNED**

   If CLU’s ST command indicates segment 2 is UNOWNED, the contents of segment 1 (as specified in the “Segment 1 Image” line) dictate how to proceed:

   a. **Coprocessor in Factory-Fresh State** - If software has never been loaded into the coprocessor (for example, if the coprocessor has just been removed from a factory-sealed package), the segment 1 image name will include “FACTORY.” In this case, load CR1rrrss.CLU into the coprocessor, for example:

   ```
   CSUNCLU \logfile-directory\41-00049.log PL \pkcs11\etc\CR1rrrss.CLU
   ```

   CR1rrrss updates the system software in segment 1.

   After this command has been performed, segment 1 has been loaded with the current version of the required system software. Proceed as directed in case 1c on page 4-4.

   b. **Segment 1 Downlevel** - If segment 1 contains a downlevel version or revision of CCA segment 1, load CE1rrrss.CLU into the coprocessor.

   Warning

   CE1rrrss.CLU updates the public key associated with segment 1. This key can only be updated a few times before the coprocessor runs out of memory in which to store the certificate chain connecting the segment 1 public key to the original key installed at the factory. Users should update the system software in a coprocessor as seldom as possible. Note that CE1rrrss.CLU need be loaded only once.

For example:

CSUNCLU \logfile-directory\41-00049.log PL \pkcs11\etc\CE1rrrss.CLU

CE1rrrss.CLU updates the system software in segment 1.

The README file specifies which version and revision of CCA segment 1 is current.

After this command has been performed, segment 1 has been loaded with the current version of the required system software. Proceed as directed in case 1c.

c. **Segment 1 Current** - If segment 1 contains the appropriate version and revision of CCA segment 1, load PNWrrrss.CLU into the coprocessor, for example:

   CSUNCLU \logfile-directory\41-00049.log PL \pkcs11\etc\PNWrrrss.CLU

   PNWrrrss.CLU loads the coprocessor operating system into segment 2 and the PKCS #11 application into segment 3. Segments 2 and 3 are now RUNNABLE. Proceed as directed for case 2.

   The README file specifies which version and revision of CCA segment 1 is current.

2. **Case 2 - Segment 2 RUNNABLE, Segment 3 RUNNABLE**

   If CLU's ST command indicates both segments 2 and 3 are RUNNABLE, the owner identifiers associated with segments 2 and 3 dictate how to proceed.

   a. **Segment 2 Owner ID 2 and Segment 3 Owner ID 14** - If the owner identifier associated with segment 2 is 2 and the owner identifier associated with segment 3 is 14, the PKCS #11 application has already been loaded into the coprocessor. You may wish to confirm the segment contents as described in "Validating the Coprocessor Segment Contents" on page 4-5.

      You may update the operating system and the PKCS #11 application by loading PEXrrrss.CLU into the coprocessor, for example:

      CSUNCLU \logfile-directory\41-00049.log PL \pkcs11\etc\PEXrrrss.CLU

   b. **Segment 2 Owner ID 2 and Segment 3 Owner ID not 14** - If the owner identifier associated with segment 2 is 2 and the owner identifier associated with segment 3 is not 14, load CRSrrrss.CLU into the coprocessor, for example:

      CSUNCLU \logfile-directory\41-00049.log PL \pkcs11\etc\CRSrrrss.CLU

      CRSrrrss.CLU relinquishes ownership of segment 2. It also removes the code from segments 2 and 3 and erases any information that the application in segment 3 has saved in the coprocessor's nonvolatile memory.

      If this command fails, further assistance from IBM is required. (The failure may indicate the public key associated with segment 2 has not been set to the expected value.)

      If this command succeeds, segment 2 is UNOWNED. Proceed as directed for case 1 on page 4-3.

   c. **Segment 2 Owner ID not 2** - If the owner identifier associated with segment 2 is not 2, it may not be possible to load the PKCS #11 application into the coprocessor. To do so requires the assistance of the owner of segment 2, who must supply a CLU file to surrender that ownership. If such a CLU file can be obtained and loaded, segment 2 will become UNOWNED and the instructions for case 1 on page 4-3 apply.

Validating the Coprocessor Segment Contents

During manufacture, each coprocessor generates an RSA keypair (the “device key”) and exports the public key. (The private key is stored in the card and never leaves it.) IBM uses a second RSA keypair key (the “class key”) to generate a certificate for the device public key. The certificate includes a digital signature of the device public key; the digital signature is produced using the class private key. The device key certificate is stored in the coprocessor’s nonvolatile memory.

The class key used for a particular coprocessor depends on several factors (including the model and operating voltage). Certificates for all class keys currently in use are shipped with the PKCS #11 Support Program (see “Directories and Files” on page 4-6 for details). These certificates are signed using IBM’s root keypair.

The CLU VA command essentially confirms that a coprocessor contains the software it claims it contains. In particular, the VA command

1. Uses IBM’s public root key (which is hardcoded into CLU[1] and can be found in Appendix D, “The IBM Root Public Key” on page D-1) to validate the class key certificate
2. Retrieves the device key certificate from the coprocessor and validates the device key certificate using the class key
3. Retrieves a copy of the device status information that has been signed with the device private key and validates the status information using the device public key

The README file describes the expected response from the VA command.

Sample VA commands are:

- For a 5V Model 002:

  CSUNCLU \logfile-directory\41-00049.log VA \pkcs11\etc\40H9951V.CLU

- For a 3.3V Model 023:

  CSUNCLU \logfile-directory\41-00049.log VA \pkcs11\etc\40H9858V.CLU

How to Unload Coprocessor Software and Zeroize the PKCS #11 Node

To remove the PKCS #11 application and operating system from the coprocessor and erase any token objects that have been created, load CRSrrrss.CLU into the coprocessor, for example:

CSUNCLU \logfile-directory\41-00049.log PL\pkcs11\etc\CRSrrrss.CLU

[1] Cautious users should ensure they have an unmodified copy of CSUNCLU.EXE.

Directories and Files

The *pkcs11/etc* directory contains files to be used as input to CLU including those listed as follows.

- CR1rrrss.CLU, which loads release rrr revision ss of IBM's system software into a coprocessor. The system software includes basic hardware diagnostic routines and the software mechanisms that ensure nothing is loaded into the coprocessor without proper authorization.

  CR1rrrss.CLU can only be loaded into an IBM 4758 in the factory-fresh state, that is, one in which the segment 1 image name reported by CLU's ST command includes "FACTORY."

- CE1rrrss.CLU, which updates the system software in a coprocessor.

  CE1rrrss.CLU loads release rrr revision ss of IBM's system software into an IBM 4758 into which system software has previously been loaded, that is, one in which the segment 1 image name reported by CLU's ST command includes "CCA" or "PKCS#11."

  Warning

  CE1rrrss.CLU updates the public key associated with segment 1. This key can only be updated a few times before the coprocessor runs out of memory in which to store the certificate chain connecting the segment 1 public key to the original key installed at the factory. Users should update the system software in a coprocessor as seldom as possible. Note that CE1rrrss.CLU need be loaded only once.

- PNWrrrss.CLU, which loads into a coprocessor a copy of release rrr revision ss of the operating system (CP/Q^{++}) and the PKCS #11 application.

  PNWrrrss.CLU can only be loaded into an IBM 4758 that contains release rrr revision ss of IBM's system software.

- PEXrrrss.CLU, which loads into a coprocessor a copy of release rrr revision ss of the operating system (CP/Q^{++}) and the PKCS #11 application.

  PEXrrrss.CLU can only be loaded into an IBM 4758 that already contains a copy of the operating system and the PKCS #11 application. This file is supplied so that users can easily upgrade an existing PKCS #11 installation. Loading PEXrrrss.CLU does not affect any token objects that have been created.

- CRSrrrss.CLU, which removes the operating system and application from an IBM 4758 into which the PKCS #11 application has been loaded. CRSrrrss.CLU essentially restores the coprocessor to the state it is in immediately after CR1rrrss.CLU or CE1rrrss.CLU has been loaded.

  CRSrrrss.CLU can only be loaded into an IBM 4758 into which the PKCS #11 application has been loaded. Any token objects stored on the coprocessor are destroyed.

- A number of files whose names are of the form *xxxxxxx*V.CLU, which contain the class key certificates used to validate the software in a coprocessor. The coprocessor's part number determines the proper file to use with the CLU VA command. The part number appears on a white label located on the side of the metal can that houses the coprocessor CPU furthest from the batteries. The first letters on the label are 11Y. For example, 40H9858V.CLU is the appropriate file to use with a 3.3V 4758 Model 023 (P/N 40H9858).

Chapter 5. Building Applications on Windows NT and Windows 2000 That Use the PKCS #11 API

This chapter includes an overview of the way in which applications obtain service from the PKCS #11 application program interface (API).

Source code for sample routines is shipped with the software. You can use the samples to test the coprocessor and the support program.

Overview

Applications issue service requests to the PCI Cryptographic Coprocessor by calling PKCS #11 functions, which are entry points in the PKCS #11 DLL (CRYPTOKI.DLL). The DLL in turn calls the coprocessor physical device driver (PDD). The hardware and software accessed through the API are themselves an integrated subsystem.

The PKCS #11 API is defined by RSA Laboratories. Refer to the *http://www.rsasecurity.com/rsalabs/pkcs/pkcs-11/* for specifications. The PKCS #11 Support Program defines several nonstandard extensions to the PKCS #11 API; see “PKCS #11 Support Program API Extensions” for details.

PKCS #11 Directories

- The *pkcs11/include* directory contains include (*.h*) files that define the constants, types, functions, and so on that are of interest to an application that uses the PKCS #11 API.
- The *pkcs11/lib* directories contain library (*.lib*) files that invoke the requisite DLL entry points to perform the requested PKCS #11 function. The *msvcmasm* subdirectory is used when building applications with MSVC++ and the *vacppmsm* subdirectory is used when building applications with VACPP.

PKCS #11 Support Program API Extensions

The PKCS #11 Support Program includes the following APIs that are not part of the PKCS #11 standard.

Outbound Authentication

| **Note:** Outbound Authentication is not supported by AIX.

The Outbound Authentication functions allow a coprocessor application to request services from the IBM PCI Cryptographic Coprocessor’s Outbound Authentication (OA) Manager, which supports cryptographic operations and data structures that allow the coprocessor application to authenticate itself to another agent and to engage in a wide range of cryptographic protocols. In particular, a coprocessor application can use these functions to:

- Prove to another agent that the coprocessor on which the application is running has not been tampered with

- Provide another agent a list of all the software that has ever been loaded on the coprocessor that could have revealed the application's secrets or compromised the authentication scheme
- Report in a manner that cannot be forged (unless the authentication scheme has been compromised) the status of the coprocessor, including its serial number and the identity of the software it contains
- Perform general cryptographic operations (encryption, decryption, signing, and verification) and engage in cryptographic protocols (for example, key exchange) using keys whose validity is assured by the authentication scheme

The Outbound Authentication interface is defined in */pkcs11/include/oa.h* and includes the following functions.

```
CK_RV C_GetOACertificates(CK_SLOT_ID  slot_id,
                          CK_BYTE_PTR pCertificates,
                          CK_ULONG_PTR)
```

C_GetOACertificates retrieves a certificate chain from the PCI Cryptographic Coprocessor identified by `slot_id`. The certificate chain links the key the coprocessor uses to sign nonces to IBM's root public key and thus ensures the validity of the information in the certificate chain and the authenticity of the key used to sign nonces.

If `pCertificates` is NULL, the number of bytes the certificate chain occupies is returned in `*pulLen`.

If pCertificates is not NULL, it must point to a buffer that is large enough to hold the certificate chain. `*pulLen` must contain the number of bytes the certificate chain occupies (for example, as returned by an earlier call to C_GetOACertificates with `pCertificates` set to NULL). On return, the buffer referenced by `pCertificates` contains the certificate chain.

The format of the certificate chain is rather complex. Sample code that parses and validates the chain is provided in */pkcs11/src/samples/oa*, and details on the contents of the chain can be found in Chapter 3 of the *IBM 4758 PCI Crytographic Coprocessor Custom Software Interface Reference*.

```
CK_RV C_GetOANonce(CK_SLOT_ID   slot_id,
                   CK_ULONG     ulUserRandom,
                   OA_NONCE_PTR pNonce,
                   CK_ULONG_PTR pulLen)
```

C_GetOANonce causes the PCI Cryptographic Coprocessor identified by `slot_id` to generate a digital signature based on the random number `ulUserRandonm`.

If pNonce is NULL, the number of bytes the signature occupies is returned in `*pulLen`.

If pNonce is not NULL, it must point to a buffer that is large enough to hold the signature. `*pulLen` must contain the number of bytes the signature occupies (for example, as returned by an earlier call to C_GetOANonce with `pNonce` set to NULL). On return, the buffer referenced by `pNonce` contains a random number generated on the coprocessor, the length of the signature, and the signature itself. The signature is a DSA signature generated by signing a message consisting of ulUserRandom concatenated with the random number generated on the coprocessor.[1]

Sample code that validates the signature on a nonce is provided in */pkcs11/src/samples/oa*.

Compiling and Linking Application Programs

The support program includes the C Language source code and the makefile for a sample program. The files reside in subdirectories of the */pkcs11/src* directory:

The following makefiles are provided to build sample programs with IBM VisualAge C++ for Windows:

/pkcs11/src/samples/simple/cryptibm.mak
/pkcs11/src/samples/tokens/tokeni.mak

The following makefiles are provided to build sample programs with Microsoft Visual C++:

/pkcs11/src/samples/simple/cryptmsvc.mak
/pkcs11/src/samples/tokens/tokenm.mak
/pkcs11/src/samples/oa/sampa.mak

The Outbound Authentication sample in the \pkcs11\src\samples\oa subdirectory uses RSA Laboratories' BSAFE library to perform certain cryptographic validation operations on the host.

[1] The coprocessor signs nonces with the current configuration key (and first generates a configuration key if necessary).

Appendix A. Overview of the PKCS #11 Application Download Process

1. Determine whether or not the coprocessor is empty, for example:

   CSUNCLU \logfile-directory\41-00049.log ST

   If coprocessor segment 1 is not in the INITIALIZED state or if page 1 is not certified, the PKCS #11 application cannot be downloaded into the coprocessor without additional assistance from IBM.

   If coprocessor segment 2 is UNOWNED, continue with step 2.

   If segment 2 is OWNED_BUT_UNRELIABLE or if the owner identifier associated with segment 2 is not 2, it may not be possible to load the PKCS #11 application into the coprocessor. To do so requires the assistance of the owner of segment 2, who must supply a CLU file to surrender that ownership.

   If the owner identifier associated with segment 2 is 2 and the owner identifier associated with segment 3 is not 14, continue with step 3.

   If the owner identifier associated with segment 2 is 2 and the owner identifier associated with segment 3 is 14, continue with step 4.

2. If coprocessor segment 2 is UNOWNED, the contents of segment 1 dictate how to proceed:

   - **Coprocessor in Factory-Fresh State** - If software has never been loaded into the coprocessor (for example, if the coprocessor has just been removed from a factory-sealed package), the segment 1 image name will include “FACTORY.” In this case, load CR1rrrss.CLU into the coprocessor, for example:

     CSUNCLU \logfile-directory\41-00049.log PL \pkcs11\etc\CR1rrrss.CLU

     CR1rrrss updates the system software in segment 1.

     After this command has been performed, segment 1 has been loaded with the current version of the required system software. Proceed as directed in “Segment 1 Current” on page A-2.

   - **Segment 1 Downlevel** - If segment 1 contains a downlevel version or revision of CCA segment 1, load CE1rrrss.CLU into the coprocessor, for example:

     CSUNCLU \logfile-directory\41-00049.log PL \pkcs11\etc\CE1rrrss.CLU

     Proceed to load PNWrrrss.CLU as indicated in “Segment 1 Current”.

     Warning

     CE1rrrss.CLU updates the public key associated with segment 1. This key can only be updated a few times before the coprocessor runs out of memory in which to store the certificate chain connecting the segment 1 public key to the original key installed at the factory. Users should update the system software in a coprocessor as seldom as possible. Note that CE1rrrss.CLU need be loaded only once.

- **Segment 1 Current -** If segment 1 contains the appropriate version and revision of CCA segment 1, load PNWrrrss.CLU into the coprocessor, for example:

  ```
  CSUNCLU \logfile-directory\41-00049.log PL \pkcs11\etc\POSrrrss.CLU
  ```

  Proceed to step 4.

3. If the owner identifier associated with segment 2 is 2 and the owner identifier associated with segment 3 is not 14, relinquish ownership of segment 2 by loading CRSrrrss.CLU into the coprocessor, for example:

   ```
   CSUNCLU \logfile-directory\41-00049.log PL \pkcs11\etc\CRSrrrss.CLU
   ```

   If this command fails, further assistance from IBM is required. (The failure may indicate the public key associated with segment 2 has not been set to the expected value.)

   If this command succeeds, segment 2 and segment 3 become UNOWNED. Proceed to step 2 on page A-1.

4. If the owner identifier associated with segment 2 is 243 and the owner identifier associated with segment 3 is 14, the PKCS #11 application has already been loaded into the coprocessor. You may wish to confirm the segment contents as described in “Validating the Coprocessor Segment Contents” on page 4-5.

   You may update the operating system and the PKCS #11 application by loading PEXrrrss.CLU into the coprocessor, for example:

   ```
   CSUNCLU \logfile-directory\41-00049.log PL \pkcs11\etc\PEXrrrss.CLU
   ```

This completes the download of the PKCS #11 application to the coprocessor.

Appendix B. Using CLU

The Coprocessor Load Utility (CSUNCLU.EXE) interacts with the coprocessor's ROM-based system software to update software in flash.[1] The Coprocessor Load Utility can also obtain information about the coprocessor, reset the coprocessor, or validate the software in the coprocessor.

Syntax

CSUNCLU *logfilename* {**PL** | **RS** | **SS** | **ST** | **VA**} [*coprocessornumber*] [*clufilename*]

where:

- *logfilename* is the name of a file to which CLU appends information about the operation and its results. The file is created if it does not exist. Path information must also be provided if the file is not in the current directory.

  It is strongly recommended that the coprocessor serial number be used as the log file name. (The serial number appears on the label on the bracket located at the end of the coprocessor.) This practice ensures a complete history of status and code changes for the contents of each coprocessor is available.

  CLU also appends log information in machine-readable form to a file with the same name as the log file name and the extension .MRL.

- The second argument specifies the operation CLU is to perform. Recognized values are as follows:

  – **PL** - Download a file containing software and/or commands to the coprocessor.
  – **RS** - Reset the coprocessor.
  – **SS** - Print information about every coprocessor installed in a host and the application each coprocessor contains.
  – **ST** - Print information about the coprocessor and the software it contains.
  – **VA** - Print and validate information about the coprocessor and the software it contains.

- More than one coprocessor may be installed in a host. *coprocessornumber* identifies the coprocessor with which CLU is to interact. The default is 0.

  The number assigned to a particular coprocessor depends on the order in which information about devices in the system is presented to the device driver by the host operating system. At the present time there is no way to tell *a priori* which coprocessor will be assigned a given number.

- *clufilename* is the name of the file containing software and commands to download to the coprocessor. Path information must also be provided if the file is not in the current directory. This name appears only if the **PL** or **VA** operation is specified.

If no arguments are provided CLU runs interactively and prompts for them.

[1] The syntax diagram in this appendix assumes the directory that contains the various utilities shipped with the support program is in the search path for executable files (that is, the PATH environment variable includes *pkcs11\bin\nt*).

Return Codes

When the utility finishes processing, it returns a value that can be tested in a script file or in a command file. The returned values are:

0 OK.

1 Command line parameters not valid.

2 Cannot access the coprocessor. Be sure that the coprocessor and its driver have been properly installed.

3 Check the utility log file for an abnormal condition report.

4 No coprocessor installed. Be sure that the coprocessor and its driver have been properly installed.

5 Invalid coprocessor number specified.

6 A data file is required with this command.

7 The data file specified with this command is incorrect or invalid.

Appendix C. Device Driver Error Codes

Each time that the coprocessor is reset, and the reset is not caused by a fault or tamper event, the coprocessor runs through “Miniboot,” its power-on self-test (POST), code-loading, and status routines. During this process the coprocessor attempts to coordinate with a host-system device driver. Coprocessor resets can occur as a result of power-on, a reset command sent from the device driver, or as a result of coprocessor internal activity such as completion of code updates.

The coprocessor can also reset if the coprocessor’s fault or tamper detection circuitry reset the coprocessor.

The coprocessor device driver monitors the status of its communication with the coprocessor and the coprocessor hardware status registers. Programs such as the Coprocessor Load Utility (CLU), and the CCA and PKCS #11 Support Programs code can receive unusual status in the form of a 4-byte return code from the device driver.

There are a very large number of possible 4-byte codes, all of which are of the form X'8xxxxxxx'. The most likely codes that may be encountered are described in Table C-1 on page C-2. If you encounter codes of the form X'8340xxxx' or X'8440xxxx', and the code is not in the following list, contact the IBM 4758 Support organization for advice via the question form on the IBM 4758 product Web site (*http://www.ibm.com/security/cryptocards*).

Table C-1. Device Driver Error Codes in the Class X'8xxxxxxx'

4-byte Return Code (hex)	**Reason**	**Considerations**
8040FFBF	External intrusion	Arises due to optional electrical connection to the coprocessor. This condition can be reset.
8040FFDA	Dead battery	The batteries have been allowed to run out of sufficient power, or have been removed. The coprocessor is zeroized and is no longer functional.
8040FFDB	Xray tamper	The coprocessor is zeroized and is no longer functional.
8040FFEB	Temperature tamper	High or low temperature has been exceeded. The coprocessor is zeroized and is no longer functional.
8040FFF3	Voltage tamper	The coprocessor is zeroized and is no longer functional.
8040FFF9	Mesh tamper	The coprocessor is zeroized and is no longer functional.
8040FFFE	Battery warning	Battery power is marginal. The battery changing procedure described in the IBM 4758 Installation Manual should be followed to replace the batteries.
804xxxxx (for example, 80400005)	General communication problem	Except for the prior X'8040xxxx' codes, there are additional conditions that arise in host-coprocessor communication. Determine that the host system in fact has a coprocessor. Try removing and reinserting the coprocessor into the PCI bus. Run the CLU status command (ST). If problems persists, contact IBM 4758 Support via the website.
8340xxxx	Miniboot-0 codes	This class of return code arises from the lowest-level of reset testing.
8340038F	Random number generation fault	Continuous monitoring of the random number generator has detected a possible problem. There is a small statistical probability of this event occurring without indicating an actual ongoing problem. The CLU status (ST) command should be run at least twice to determine if the condition can be cleared.
8440xxxx	Miniboot-1 codes	This class of return code arises from the replaceable POST and code-loading code.
844006B2	Invalid signature	The signature on the data sent from the CLU utility to Miniboot could not be validated by Miniboot. Be sure that you are using an appropriate file (for example, **CR1**xxxxx.CLU versus **CE1**xxxxx.CLU). If the problem persists, obtain the output of a CLU status report and forward this and a description of what you are trying to accomplish to Customer Support using the IBM 4758 website reporting process.

Appendix D. The IBM Root Public Key

As of the date of this document, the key IBM uses to sign the certificates for the class keys used with the IBM 4758 model 002/023 is a 1024-bit RSA key whose public exponent is 65537 (decimal) and whose modulus in hex is as follows:

```
80000000 00000000
00000000 00000010
0CACBAED FCEB4A2D
1FCE8B0F 42AA10DE
B9405685 C800156C
000D4635 811F34D4
375F17F0 3445EC7B
C2516182 20F75391
D0F91FE6 AA52CA9A
463FE87B F78FF842
A770EEC4 B8B07FD5
55BC54DF 194F3FC6
CE1B4936 EE0BAA1E
4E7E6D57 494E8334
26185CD3 6440ED2B
03963DBC 432DF717
```

The most significant byte of the modulus is 0x80 and the least significant byte is 0x17.

Appendix E. Notices

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM's product, program, or service may be used. Any functionally-equivalent product, program, or service that does not infringe any of IBM's intellectual property rights, or other legally protectable rights, may be used instead of the IBM product, program, or service. Evaluation and verification of operation in conjunction with other products, programs, or services, except those expressly designated by IBM, are the user's responsibility.

Licensors of this program who wish to have information about it for the purpose of enabling (i) the exchange of information between independently-created programs and other programs (including this one), and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Department MG39/201
8501 IBM Drive
Charlotte, NC 28262-8563, U.S.A.

Such information may be available—subject to appropriate terms and conditions—including, in some cases, the payment of a fee.

IBM may have patents or pending-patent applications covering subject matter 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 the IBM Director of Commercial Relations, IBM Corporation, Purchase, NY 10577.

License

You can obtain the files for the PKCS #11 Support Program feature by downloading from the product Web site at *http://www.ibm.com/security/cryptocards*.

- Feature Code 4396 identifies the Windows NT workstation software.

The PKCS #11 Support Program must be used in accordance with the IBM System Programs License Agreement.

Copying and Distributing Softcopy Files

For online versions of this book, we authorize you to:

- Copy, modify, and print the documentation contained on the media, for use within your enterprise, provided you reproduce the copyright notice, all warning statements, and other required statements on each copy or partial copy.
- Transfer the original unaltered copy of the documentation when you transfer the related IBM product (which may be either machines you own, or programs, if the program's license terms permit a transfer). You must, at the same time, destroy all other copies of the documentation.

You are responsible for payment of any taxes, including personal property taxes, resulting from this authorization.

THERE ARE NO WARRANTIES, EXPRESS OR IMPLIED, INCLUDING THE WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

Some jurisdictions do not allow the exclusion of implied warranties, so the above exclusion may not apply to you.

Your failure to comply with the terms above terminates this authorization. Upon termination, you must destroy your machine readable documentation.

Trademarks

The following terms are trademarks of the IBM Corporation in the United States or other countries or both:

AIX
IBM
RS/6000
SecureWay
System/390
VisualAge

Windows, Windows NT, and Windows 2000 are trademarks of the Microsoft Corporation in the United States, or other countries, or both.

BSAFE is a trademark of RSA Data Security, Inc.

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

List of Abbreviations and Acronyms

ANSI	american national standards institute
API	application program interface
ASCII	american national standard code for information interchange
CCA	common cryptographic architecture
CDMF	commercial data masking facility
CLU	coprocessor load utility
CP/Q++	control program/q with 4758 extensions
CV	control vector
DES	data encryption standard
DMA	direct memory access
FIPS	federal information processing standard
IBM	international business machines
ICSF	integrated cryptographic service facility
I/O	input/output
ISO	international organization for standardization
LU	logical unit
MB	megabyte
MAC	message authentication code
MD5	message digest 5 (hashing algorithm)
OEM	original equipment manufacturer
PC	personal computer
PCI	peripheral component interconnect
PDD	physical device driver
PDF	portable document format
PKA	public key algorithm
PKCS	public key cryptography standard
ROM	read only memory
RSA	rivest, shamir, and adleman (algorithm)
SCC	secure cryptographic coprocessor

Glossary

This glossary includes some terms and definitions from the *IBM Dictionary of Computing*, New York: McGraw Hill, 1994. This glossary also includes some terms and definitions taken from:

- The *American National Standard Dictionary for Information Systems*, ANSI X3.172-1990, copyright 1990 by the American National Standards Institute (ANSI). Copies may be purchased from the American National Standards Institute, 11 West 42 Street, New York, New York 10036. Definitions are identified by the symbol (A) following the definition.
- The *Information Technology Vocabulary*, developed by Subcommittee 1, Joint Technical Committee 1, of the International Organization for Standardization and the International Electrotechnical Commission (ISO/IEC JTC1/SC1). Definitions of published parts of this vocabulary are identified by the symbol (I) following the definition; definitions taken from draft international standards, committee drafts, and working papers being developed by ISO/IEC JTC1/SC1 are identified by the symbol (T) following the definition, indicating that final agreement has not yet been reached among the participating National Bodies of SC1.

A

access. In computer security, a specific type of interaction between a subject and an object that results in the flow of information from one to the other.

access control. Ensuring that the resources of a computer system can be accessed only by authorized users and in authorized ways.

access method. A technique for moving data between main storage and input/output devices.

american national standard code for information interchange (ASCII). The standard code, using a coded character set consisting of seven-bit characters (eight bits including parity check), that is used for information interchange among data processing systems, data communication systems, and associated equipment. The ASCII set consists of control characters and graphic characters. (A)

american national standards institute (ANSI). An organization consisting of producers, consumers, and general interest groups that establishes the procedures by which accredited organizations create and maintain voluntary industry standards for the United States. (A)

application program interface (API). A functional interface supplied by the operating system or by a separate program that allows an application program written in a high-level language to use specific data or functions of the operating system or the separate program.

authentication. (1) A process used to verify the integrity of transmitted data, especially a message. (T) (2) In computer security, a process used to verify the user of an information system or protected resource.

authorization. (1) In computer security, the right granted to a user to communicate with or make use of a computer system. (T) (2) The process of granting a user either complete or restricted access to an object, resource, or function.

authorize. To permit or give authority to a user to communicate with or make use of an object, resource, or function.

C

card. (1) An electronic circuit board that is plugged into an expansion slot of a system unit. (2) A plug-in circuit assembly.

CDMF algorithm. An algorithm for data confidentiality applications; it is based on the DES algorithm and possesses 40-bit key strength.

common cryptographic architecture (CCA) API. The application program interface described in the *IBM 4758 CCA Basic Services Reference and Guide*, SC31-8609.

coprocessor. (1) A supplementary processor that performs operations in conjunction with another processor. (2) A microprocessor on an expansion card that extends the address range of the processor in the host system, or adds specialized instructions to handle a particular category of operations; for example, an I/O coprocessor, math coprocessor, or a network coprocessor.

cryptographic coprocessor (IBM 4758). An expansion board that provides to a workstation a comprehensive set of cryptographic functions.

cryptographic node. A node that provides cryptographic services, such as key generation and digital signature support.

cryptography. (1) The transformation of data to conceal its meaning. (2) In computer security, the principles, means, and methods used to transform data.

D

data encrypting key. A key used to encipher, decipher, or authenticate data.

data encryption standard (DES). The National Institute of Standards and Technology (NIST) Data Encryption Standard, adopted by the U.S. government as Federal Information Processing Standards (FIPS) Publication 46 which allows only hardware implementations of the data encryption algorithm.

decipher. (1) To convert enciphered data into clear data. (2) Contrast with *encipher*.

direct memory access (DMA). The transfer of data between memory and input/output units without processor intervention.

driver. A program that contains the code needed to attach and use a device.

E

encipher. (1) To scramble data or to convert data to a secret code that masks the meaning of the data. (2) Contrast with *decipher*.

enciphered data. Data whose meaning is concealed from unauthorized users or observers.

expansion board. Synonym for *expansion card*.

expansion card. (1) A circuit board that a user can install in an expansion slot to add memory or special features to a computer. (2) Synonym for *card*.

expansion slot. One of several receptacles in a PC or RS/6000 machine into which a user can install an expansion card.

F

feature. A part of an IBM product that can be ordered separately.

federal information processing standard (FIPS). A standard that is published by the US National Institute of Science and Technology (NIST).

H

host computer. In regard to the PKCS #11 Support Program, the workstation into which the IBM 4758 PCI Cryptographic Coprocessor is installed.

I

inline code. In a program, instructions that are executed sequentially, without branching to routines, subroutines, or other programs.

integrated cryptographic service facility (ICSF). An IBM-licensed program that supports the cryptographic hardware feature in the MVS environment for the high-end System/390® processor.

interface. (1) A boundary shared by two functional units, as defined by functional characteristics, signal characteristics, or other characteristics as appropriate. The concept includes specification of the connection between two devices having different functions. (T) (2) Hardware, software, or both, that links systems, programs, and devices.

international organization for standardization (ISO). An organization of national standards bodies established to promote the development of standards to facilitate the international exchange of goods and services, and to foster cooperation in intellectual, scientific, technological, and economic activity.

K

key. In computer security, a sequence of symbols used with an algorithm to encipher or decipher data.

M

master key. In the IBM 4758's PKCS #11 Support Program implementation, the key used to encrypt keys to process other keys or data at the node.

multi-user environment. A computer system that supports terminals and keyboards for more than one user at the same time.

N

national institute of science and technology (NIST). Current name for the US National Bureau of Standards.

node. (1) In a network, a point at which one or more functional units connects channels or data circuits. (I) (2) The endpoint of a link or a junction common to two or more links in a network. Nodes can be processors, communication controllers, cluster controllers, or terminals. Nodes can vary in routing and other functional capabilities.

P

passphrase. In computer security, a string of characters known to the computer system and to a user; the user must specify it to gain full or limited access to the system and the data stored therein.

PKCS #11. RSA Laboratories' cryptographic token interface standard.

private key. (1) In computer security, a key that is known only to the owner and used with a public key algorithm to decipher data. Data is enciphered using the related public key. (2) Contrast with *public key*. (3) See also *public key algorithm*.

procedure call. In programming languages, a language construct for invoking execution of a procedure. (I) A procedure call usually includes an entry name and the applicable parameters.

profile. Data that describes the significant characteristics of a user, a group of users, or one-or-more computer resources.

public key. (1) In computer security, a key that is widely known and used with a public key algorithm to encipher data. The enciphered data can be deciphered only with the related private key. (2) Contrast with *private key*. (3) See also *public key algorithm*.

public key algorithm (PKA). (1) In computer security, an asymmetric cryptographic process that uses a public key to encipher data and a related private key to decipher data. (2) Contrast with *data encryption algorithm* and *data encryption standard algorithm*. (3) See also *RSA algorithm*.

Public-Key Cryptographic Standards (PKCS) #11. RSA Laboratories' cryptographic token interface standard.

R

read only memory (ROM). Memory in which stored data cannot be modified routinely.

RSA algorithm. A public key encryption algorithm developed by R. Rivest, A. Shamir, and L. Adleman.

S

security. The protection of data, system operations, and devices from accidental or intentional ruin, damage, or exposure.

system administrator. The person at a computer installation who designs, controls, and manages the use of the computer system.

T

token. (1) A string of characters treated as a single entity. (2) A particular message or bit pattern that signifies permission to transmit.

U

utility program. A computer program in general support of computer processes. (T)

V

verb. A function possessing an entry_point_name and a fixed-length parameter list. The procedure call for a verb uses the syntax standard to programming languages.

W

workstation. A terminal or microcomputer, usually one that is connected to a mainframe or a network, from which a user can perform applications.

Numerics

4758. IBM 4758 PCI Cryptographic Coprocessor.

Index

A

application download process, overview A-1
applications, building to use with the PKCS #11 API 5-1
auditor 4-2

B

batteries, coprocessor
- removal 2-3
- replacement kit 2-2

building applications to use with the PKCS #11 API 5-1

C

CLU (coprocessor load utility)
- commands B-1
- files, determining which to load 4-3—4-5
- overview 4-1
- return codes B-2
- software, determining which is loaded in the coprocessor 4-1
- syntax B-1
- using B-1

compile, application programs 5-3
components, support program 3-1
coprocessor
- installation 2-3
- load, software 4-1
- replacement kit, batteries 2-2

coprocessor load utility
- *See* CLU (coprocessor load utility)

coprocessor support program
- *See* support program

csuf.com 3-4
csuf.pkcs11 3-4

D

decipher, support program 2-3
device driver directory structure 3-2
directories and files 4-6
directories, PKCS #11 5-1
disk space, requirements 3-2
download process, PKCS #11 application A-1
download, support program 2-3

E

extensions, PKCS #11 Support Program API 5-1

F

features, product 2-2
files and directories
- coprocessor 4-6—4-7
- host 3-2—3-3

H

hardware, requirements 3-2
host install, support program
- *See* install host software

host uninstall, support program
- *See* uninstall host software

I

install host software
- AIX 3-4
- NT 3-2

installation, support program
- checklist 1-2
- into coprocessor 4-1
- onto host computer 3-1
- overview 1-1

installing the Support Program for Windows NT and Windows 2000 3-2

L

link to PKCS #11, application programs 5-3
load coprocessor software 4-1

M

makefile 5-3

O

order, support program 2-2
overview
- CLU 4-1
- installation, support program 1-1

overview of the development process A-1
owner command

P

PCI Cryptographic Coprocessor
- *See* coprocessor

PKCS #11 API, building applications to use with 5-1
PKCS #11 directories 5-1

PKCS #11 Support Program 1-1
PKCS #11 Support Program API Extensions 5-1
product
  features 2-2
public root key 4-5, D-1
publications, related vi

R
related publications vi
remove host software
  See uninstall host software
replacement kit, coprocessor batteries 2-2
requirements
  disk space 3-2
  hardware 3-2
  software 3-2
return codes, CLU B-2
root key, public 4-5, D-1

S
sample routine, C programming language
  makefile 5-3
  source code 5-3
segment 1 state 4-3
segment contents, validating 4-5
software load, coprocessor
  See load coprocessor software
software, requirements 3-2
support program
  components 3-1
  coprocessor load 4-1
  decipher 2-3
  download 2-3
  host install 3-2
  host uninstall 3-3
  overview, installation 1-1
syntax
  CLU B-1

U
uninstall host software
  AIX 3-5
  NT 3-3
  Windows 2000 3-3
unload coprocessor software 4-5
utilities
  CLU 4-1
  host 3-2

V
validating the coprocessor segment contents 4-5

Z
zeroize the PKCS #11 node 4-5