

THALES



nShield

User Guide for Unix-based OS

Version: 7.1
Date: 14 November 2012

Copyright 2012 Thales e-Security Limited. All rights reserved.

Copyright in this document is the property of Thales e-Security Limited. It is not to be reproduced, modified, adapted, published, translated in any material form (including storage in any medium by electronic means whether or not transiently or incidentally) in whole or in part nor disclosed to any third party without the prior written permission of Thales e-Security Limited neither shall it be used otherwise than for the purpose for which it is supplied.

CodeSafe, KeySafe, nCipher, nFast, nForce, nShield, payShield, and Ultrasign are registered trademarks of Thales e-Security Limited or nCipher Corporation Limited.

CipherTools, CryptoStor, CryptoStor Tape, keyAuthority, KeyVault, nCore, nethSM, nFast Ultra, nForce Ultra, nShield Connect, nToken, SafeBuilder, SEE, and Trust Appliance are trademarks of Thales e-Security Limited or nCipher Corporation Limited.

All other trademarks are the property of the respective trademark holders.

Information in this document is subject to change without notice.

Thales e-Security Limited makes no warranty of any kind with regard to this information, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Thales e-Security Limited shall not be liable for errors contained herein or for incidental or consequential damages concerned with the furnishing, performance or use of this material.

Commercial Computer Software - proprietary

This computer software and documentation is Commercial Computer Software and Computer Software Documentation, as defined in sub-paragraphs [a](1) and [a](5) of DFAR § 252.227-7014, "Rights in Noncommercial Computer Software and Noncommercial Computer Software Documentation". Use, duplication or disclosure by the Government is subject to the Thales standard US Terms And Conditions for the Product.

Patents

UK Patent GB9714757.3. Corresponding patents/applications in USA, Canada, South Africa, Japan and International Patent Application PCT/GB98/00142.

EMC compliance

The use of hand held or mobile radio equipment with a rated output power of 4W or more should not be permitted within a radius of 2m of this equipment.

FCC class A notice

This device complies with Part 15 of the FCC rules. Operation is subject to the following two conditions:

- 1 This device may not cause harmful interference, and
- 2 this device must accept any interference received, including interference that may cause undesired operation.

This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to Part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference in which case the user will be required to correct the interference at his own expense.

UL Listed Accessory

Some of the Thales modules are UL Listed Accessories. These may be identified by the UL Mark applied, as a label, to the back panel. These products should only be used with Listed ITE.

European class A notice

This device has been tested and found to comply with the requirements of the EMC directive 2004/108/EEC as a Class A product to be operated in a commercial environment at least 10m away from domestic television or radio. In a domestic environment this product may cause radio interference in which case the user may be required to take adequate measures.



Contents

Chapter 1: Introduction	10
Read this guide if ...	10
Conventions	11
Typographical conventions	11
CLI command conventions	11
Model numbers	12
Security World Software	13
Document version numbers	14
Further information	14
Contacting Support	14
Recycling and disposal information	14
 Chapter 2: Security Worlds	 16
Security	17
Smart cards	18
Remote Operator	18
Client cooperation feature	19
NIST SP800-131	19
FIPS 140-2 compliance	20
Common Criteria Compliance	21
Platform independence	22
Application independence	23
Flexibility	24
Using the Security World key: module-protected keys	24
Using Operator Card Sets: OCS-protected keys	24
Using softcard-protected keys	28
Scalability	28
Load sharing	29
Robustness	29
Backup and recovery	29
Replacing a hardware security device	30
Replacing the Administrator Card Set	30
Replacing an Operator Card Set or recovering keys to softcards	31

KeySafe and Security Worlds	32
Applications and Security Worlds	33
The nCipher PKCS #11 library and Security Worlds	33
Risks	34
Chapter 3: Support software installation	35
Before you install the software	35
Uninstall existing Software	35
Other preparatory tasks before software installation	40
Install the Security World Software	42
Installing on Solaris	43
Installing on AIX	44
Installing on HP-UX	45
Installing on Linux	46
Test the installation	48
After software installation and testing	50
Chapter 4: Software and module configuration	51
About user privileges	51
Setting up client cooperation	51
Useful utilities	53
Setting environment variables	56
Logging and debugging	56
Configuring Java support for KeySafe	57
Configuring the hardserver	57
Overview of hardserver configuration file sections	58
Enabling optional features on the module	60
Available optional features	61
Ordering additional features	65
Enabling features	66
Using multiple modules	68
Stopping and restarting the hardserver	70
Chapter 5: Creating and managing a Security World	71
Creating a Security World	71
Security World files	73
Security World options	74
Creating a Security World by using new-world	79
Creating a Security World with KeySafe	85
After you have created a Security World	93

Displaying information about your Security World	93
Displaying information about a Security World with nfkminfo	94
Adding or restoring a module to the Security World	94
Adding a module to a Security World with new-world	97
Transferring keys between Security Worlds	98
Security World migration	103
About the migration utility	103
Migrating keys to an SP800-131 Security World	104
Troubleshooting	109
Erasing a module from a Security World	111
Erasing a module with new-world	111
Erasing a module with KeySafe	112
Erasing a module with initunit	113
Deleting a Security World	113
Chapter 6: Managing card sets and softcards	115
Creating Operator Card Sets (OCSs)	116
Persistent Operator Card Sets	116
Time-outs	117
FIPS 140-2 level 3-compliant security worlds	117
Creating an Operator Card Set from the command line	118
Creating an Operator Card Set with KeySafe	120
Creating softcards	124
Creating a softcard with ppmk	125
Creating softcards with KeySafe	125
Erasing cards and softcards	128
FIPS 140-2 level 3-compliant Security Worlds	128
Erasing cards with KeySafe	128
Erasing cards from the command line	129
Erasing softcards	130
Viewing cards and softcards	131
Viewing card sets with KeySafe	131
Viewing card sets from the command line	132
Viewing softcards	133
Verifying the pass phrase of a card or softcard	134
Changing card and softcard pass phrases	135
Changing known pass phrases	136
Changing unknown or lost pass phrases	138
Replacing Operator Card Sets	140
Replacing OCSs with KeySafe	141
Replacing OCSs or softcards with rocs	143

Replacing the Administrator Card Set	151
Replacing an ACS with KeySafe	152
Replacing an Administrator Card Set with racs	154
Chapter 7: Application interfaces	155
Cryptographic Hardware Interface Library (CHIL)	155
Using keys	156
Generating keys	156
nCipher JCA/JCE CSP	156
Overview of the nCipherKM JCA/JCE CSP	156
Installing the nCipherKM JCA/JCE CSP	157
System properties	162
Compatibility	165
nCipher PKCS #11 library	165
Choosing functions	166
PKCS #11 library with Security Assurance Mechanism	172
Using the nCipher PKCS #11 library	174
nCipher PKCS #11 library environment variables	176
Checking the installation of the nCipher PKCS #11 library	187
How the nCipher PKCS #11 library protects keys	190
Restrictions on function calls in load-sharing mode	190
Vendor specific error codes	191
nShield native and custom applications	191
CodeSafe applications	192
Chapter 8: Remote Operator Card Sets	193
About Remote Operator	193
Configuring Remote Operator	194
Overview of configuring Remote Operator	194
Configuring modules for Remote Operator	194
Configuring hardservers for Remote Operator	195
Creating OCSs and keys for Remote Operator	197
Creating OCSs for use with Remote Operator	197
Loading Remote Operator Card Sets	198
Generating keys for use with Remote Operator	199
Configuring the application	199
Chapter 9: Working with keys	200
Generating keys	200
Generating keys on the command line	201
Generating keys with KeySafe	202
Generating NVRAM-stored keys	205

Importing keys	206
Importing keys from the command line	206
Importing keys with KeySafe	207
Listing supported applications with generatekey	210
Retargeting keys with generatekey	210
Viewing keys	211
Viewing keys with KeySafe	211
Viewing keys on the command line	212
Discarding keys	214
Restoring keys	214
Appendix A: Using KeySafe	215
Prerequisites for using KeySafe	215
Starting KeySafe	216
About the KeySafe window	217
Sidebar	218
Main panel area	221
Errors	224
Appendix B: Supplied utilities	226
Utilities for general operations	226
Hardware utilities	228
Test analysis tools	229
Security World utilities	229
CodeSafe utilities	233
PKCS #11	233
nShield Connect and netHSM utilities	234
Developer-specific utilities	235
Appendix C: Components on Security World Software DVD-ROMs	236
Security World Software component bundles	236
Standard component bundles	236
Additional component bundles	238
Security World for nShield User DVD-ROM	240
CipherTools DVD-ROM	241
CodeSafe DVD-ROM	242
Components required for particular functionality	242
KeySafe	243
PKCS #11 applications	243

Cryptographic Hardware Interface Library applications	243
nCipherKM JCA/JCE cryptographic service provider	243
nCipher SNMP monitoring agent	244
Appendix D: Environment variables	245
Appendix E: Logging, debugging, and diagnostics	249
Logging and debugging	249
Environment variables to control logging	249
Logging and debugging information for PKCS #11	253
Hardserver debugging	253
Debugging information for Java	254
Diagnostics and system information	255
nfdiag: diagnostics utility	256
nfkminfo: information utility	259
perfcheck: performance measurement checking tool	271
stattree: information utility	274
How data is affected when a module loses power and restarts	280
Appendix F: Hardserver configuration files	281
Hardserver configuration files	281
General hardserver configuration settings	282
server_settings	282
module_settings	284
server_remotecomms	284
server_startup	284
load_seemachine	285
slot_imports	286
slot_exports	287
Sections only in client configuration files	288
nethsm_imports	288
rfs_sync_client	289
remote_file_system	289
Appendix G: Cryptographic algorithms	291
Symmetric algorithms	291
Asymmetric algorithms	292
FIPS information	292
Appendix H: Key generation options and parameters	295
Key application type (APPNAME)	295
Key properties (NAME=VALUE)	296

Available key properties by action/application	300
Appendix I: Checking and changing module mode	302
Checking and changing nShield Solo (PCI and PCIe) module mode	302
Mode switch	302
Changing the module's mode	303
Appendix J: Upgrading firmware	308
Version Security Number (VSN)	308
Firmware on the DVD-ROM	308
Recognising firmware files	309
Using new firmware	309
Firmware installation overview	310
Upgrading both the monitor and firmware	310
Upgrading firmware only	312
After firmware installation	313
Appendix K: nCipher SNMP monitoring agent	314
Installing and activating the nCipher SNMP agent	315
Default installation settings	315
Do you already have an SNMP agent running?	315
Activating the nCipher SNMP agent	316
Further information	317
Protecting the nCipher SNMP installation	317
Configuring the nCipher SNMP agent	317
Using the nCipher SNMP agent with a manager application	319
Manager configuration	319
nCipher MIB module overview	319
MIB functionality	320
Administration sub-tree overview	321
Statistics sub-tree overview	331
Useful nCipher SNMP agent command-line switches	334
SNMP agent (snmpd) switches	334
Using the SNMP command-line utilities	336
Appendix L: Product returns	337
Glossary	338
Addresses	344



Chapter 1: Introduction

Read this guide if ...

Read this guide if you need to configure or manage:

- an nShield *hardware security module* (referred to as an *HSM* or *module*)
- any associated *Security World*. Thales hardware security devices use the Security World paradigm to provide a secure environment for all your hardware security device and key management operations.

All Thales hardware security devices support standard cryptography frameworks and integrate with many standards based products.

This guide assumes that:

- you are familiar with the basic concepts of cryptography and Public Key Infrastructure (PKI)
- you have read the *Quick Start Guide*.
- you have installed your nShield.

Note Throughout this guide, the term nShield refers generally to nShield Solo products. (nShield Solo products were formerly known as nShield.)

Note Throughout this guide, the term *Quick Start Guide* refers to the particular *Quick Start Guide* for your product.

Conventions

Typographical conventions

Note The word **Note** indicates important supplemental information.



If there is a danger of loss or exposure of key material (or any other security risk), this is indicated by a security triangle in the margin.



If there is a danger of damage to the hardware, this is indicated by a caution triangle in the margin. If you see this symbol on the product itself, see the *nShield Solo Quick Start Guide*.

Si une détérioration du matériel est possible, un triangle d'avertissement l'indique dans la marge. Si ce symbole apparaît sur le produit lui-même, reportez-vous à la partie correspondante du *nShield Solo Quick Start Guide*.

Besteht die Gefahr eines Hardware-Schadens, wird dies am Rand durch ein Warndreieck angezeigt. Falls Sie dieses Symbol auf dem Produkt selbst bemerken, schlagen Sie im zutreffenden Abschnitt des Installationshandbuchs (*nShield Solo Quick Start Guide*) nach.

Keyboard keys that you must press are represented like this: `Enter`, `Ctrl-C`.

Examples of onscreen text from graphical user interfaces, as well as names of files, command-line utilities, and other system items are represented in **boldface** text. Variable text that you either see onscreen or that you must enter is represented in *italic boldface*.

Examples of onscreen terminal display, both of data returned and of your input, are represented in a form similar to the following:

```
install
```

CLI command conventions

The basic syntax for a CLI command is:

```
command object <object_name> [parameter] [option] [modifier]
```

In this syntax, user-defined values are shown in *italics* and enclosed within the < > characters. Optional elements are shown enclosed within the [] characters. Mutually exclusive elements are separated by the | character.

Many system objects require the inclusion of a user-defined keyword value. For example, the **user** object is executed against a user-supplied *user_name*. Throughout this guide, all user-defined keyword values are shown in ***boldface italics***.

Each CLI command that you run performs an operation against the internal configuration of the appliance. The specific type of operation is specified by the first user-defined keyword value in the command string.

Model numbers

Model numbering conventions are used to distinguish different Thales hardware security devices. In the table below, *n* represents either any single-digit integer or the letter K (denoting 1000).

Model number	Used for
nCnnnnP-nnn, nCnnnnP-nKO	Thales nShield product line hardware security device with a PCI interface.
nCnnnnE-nnn	Thales nShield product line hardware security device with a PCI express interface.
nCnnnnN-nnn, nCnnnnN-nKO	netHSM
NH2047	nShield Connect 6000.
NH2040	nShield Connect 1500.
NH2033	nShield Connect 500.
NH2068	nShield Connect 6000+.
nC2023P-000	An nToken (PCI interface).
nC2021E-000, NCE2023E-000	An nToken (PCI express interface).
nC10nnP-nnn, nC10nnE-nnn	Any Thales nShield product line hardware security device that does not support key management (nFast module).
nC3nnnP-nnn, nC3nnnE-nnn, nC4nnnP-nnn, nC4nnnE-nnn, NH2047, NH2040, NH2033, NH2068.	Any Thales nShield product line hardware security device that supports key management (nShield PCI module, nShield PCIe module, nShield Connect).
nC30nnU-10, nC40nnU-10.	An nShield Edge module.

Note Throughout this guide, the term nShield refers generally to nShield Solo products. (nShield Solo products were formerly known as nShield.)

Security World Software

The *hardserver* software controls communication between applications and Thales nShield product line modules, which may be installed locally or remotely. It runs as a daemon on the host computer.

The Security World for nShield is a collection of programs and utilities, including the *hardserver*, supplied by Thales to install and maintain your Thales security system. For more information about the supplied utilities, see Appendix B: [Supplied utilities](#).

Default directories

The default locations for Security World Software and program data directories are summarized in the following table:

Directory name	Default path
nShield Installation	<code>/opt/nfast/</code>
Key Management Data	<code>/opt/nfast/kmdata/</code>
Dynamic Feature Certificates	<code>/opt/nfast/femcerts/</code>
Static Feature Certificates	<code>/opt/nfast/kmdata/features/</code>
Log Files	<code>/opt/nfast/log/</code>

Note Dynamic feature certificates must be stored in the directory stated above. The directory shown for static feature certificates is an example location. You can store those certificates in any directory and provide the appropriate path when using the Feature Enable Tool. However, you must not store static feature certificates in the dynamic features certificates directory. For more information about feature certificates, see [Enabling optional features on the module](#) on page 60.

The instructions in this guide refer to the locations of the software installation and program data directories by their names (for example, Key Management Data) or absolute paths (for example, `/opt/nfast/kmdata/`).

If the software has been installed into a non-default location, you must create a symbolic link from `/opt/nfast/` to the directory where the software is actually installed. For more information about creating symbolic links, see your operating system's documentation.

Utility help options

Unless noted, all the executable utilities provided in the **bin** subdirectory of your nShield installation have the following standard help options:

- **-h|--help** displays help for the utility

- **-v|--version** displays the version number of the utility
- **-u|--usage** displays a brief usage summary for the utility.

Document version numbers

The version number of this document is shown on the copyright page of this guide. Quote the version number and the date on the copyright page if you need to contact Support about this document.

Further information

This guide forms one part of the information and support provided by Thales. You can find additional documentation in the **document** directory of the DVD-ROM for your product.

If you have installed the CodeSafe or CipherTools developer products, the Java Generic Stub classes, nCipher KM JCA/JCE provider classes, and Java Key Management classes are supplied with HTML documentation in standard **Javadoc** format, which is installed in the appropriate **nfast/java** directory when you install these classes.

Release notes containing the latest information about your product are available in the **release** directory of your DVD-ROM or CD-ROM.

Note We strongly recommend familiarizing yourself with the information provided in the release notes before using any hardware and software related to your product.

If you would like to receive security advisories from Thales, you can subscribe to the low volume security-announce mailing list by emailing the word **subscribe** in the message body to **nShield-securityadvisories@thales-esecurity.com**.

Contacting Support

To obtain support for your product, visit <http://www.thales-esecurity.com/en/Support.aspx>.

Before contacting the Support team, click **Useful Information** and use the subtopics to see the information that the team requires.

Recycling and disposal information

In compliance with the WEEE (Waste Electrical and Electronic Equipment) directive for the recycling of electronic equipment, Thales provides a Takeback and Recycle program.

The program enables you to ship an obsolete or excess nShield product line hardware security device to Thales, who then dispose of the product in an environmentally safe manner. For further information or to arrange the safe disposal of your hardware security device, e-mail **recycling@thalessec.com**.



Chapter 2: Security Worlds

This chapter describes the *Security World* infrastructure we have developed for the secure life-cycle management of cryptographic keys. The Security World infrastructure gives you control over the procedures and protocols you need to create, manage, distribute and, in the event of disaster, recover keys.

A Security World provides you with the following features:

- security
- application independence
- platform independence
- flexibility
- scalability
- robustness.

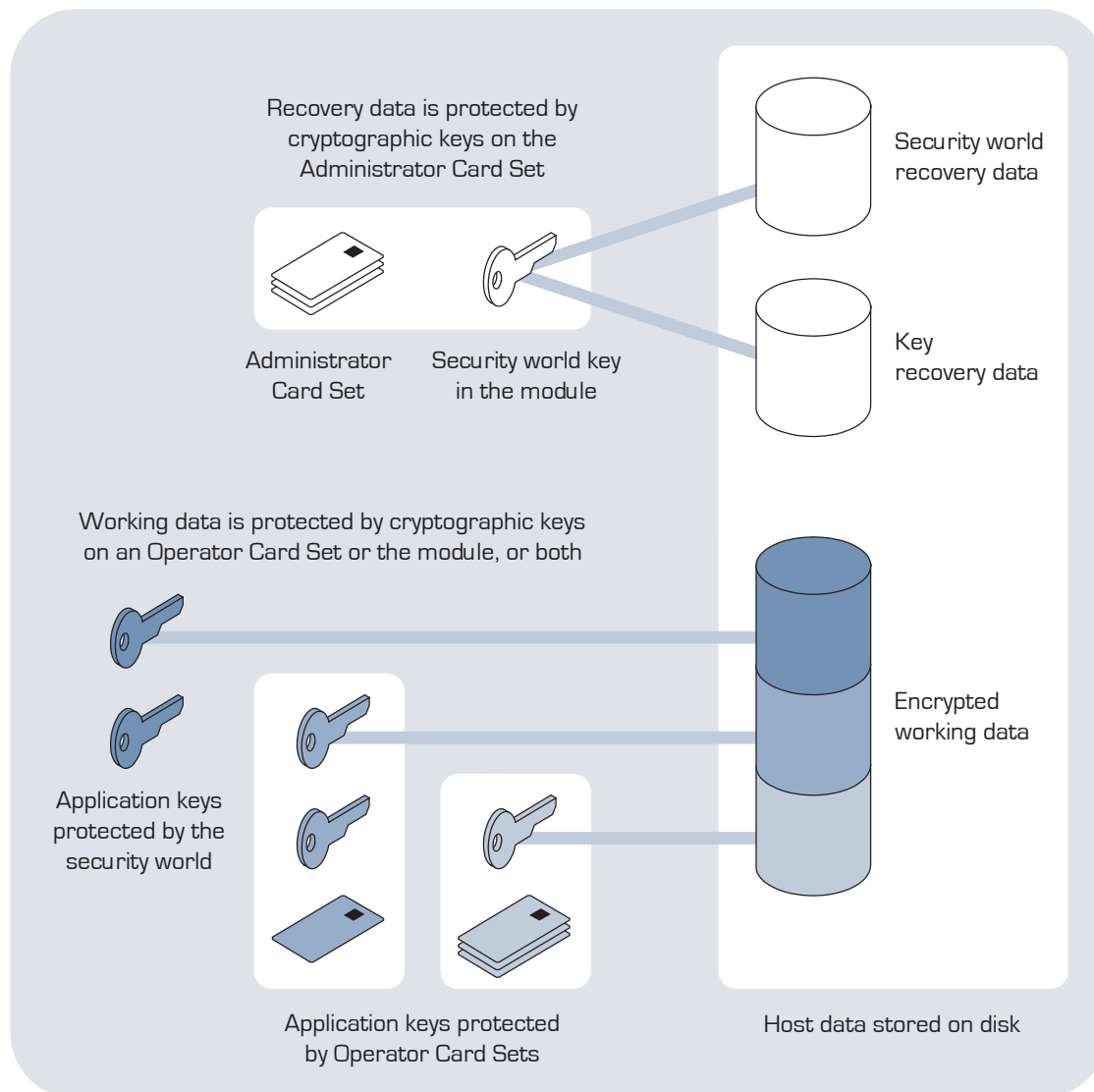
A Security World comprises:

- one or more Thales hardware security devices
- an *Administrator Card Set* (ACS), a set of Administrator smart cards used to control access to Security World configuration, as well as in recovery and replacement operations
- optionally, one or more *Operator Card Sets* (OCSs), a set or sets of Operator smart cards used to control access to application keys
- some cryptographic key and certificate data that is encrypted using the Security World key and stored on a host computer or computers.

You can add or remove cards, keys, and even hardware security devices at any time. These components are linked by the Security World key, which is unique to each world. To see how these components are related to one another, see Figure 1.

Distributing the keys used for different tasks within the Security World over different storage media means that the Security World can recover from the loss of any one component. It also increases the difficulties faced by an attacker, who needs to obtain all the components before gaining any information.

Figure 1 Key protection in a Security World



Security

We have designed the Security World technology to ensure that keys remain secure throughout their life cycle. Every key in the Security World is always protected by another key, even during recovery and replacement operations.

Because the Security World is built around Thales key-management modules, keys are only ever available in plain text on secure hardware.

Smart cards

The Security World uses:

- an *Administrator Card Set* (ACS) to control access to recovery and replacement functionality
- one or more *Operator Card Sets* (OCSs) to control access to application keys.

Note In FIPS 140-2 Level 3 Security Worlds, you require either the ACS or an OCS to authorize most operations, including the creation of keys and OCSs.

Each card set consists of a number of smart cards, N , of which a smaller number, K , is required to authorize an action. The required number K is known as the *quorum*.

Note The value for K should be less than N . We do not recommend creating card sets in which K is equal to N because an error on one card would render the whole card set unusable. If your ACS became unusable through such an error, you would have to replace the Security World and generate new keys.

An ACS is used to authorize several different actions, each of which can require a different value for K . All the card sets are distinct: a smart card can only belong to the ACS or to one OCS.

Each user can access the keys protected by the Security World and the keys protected by their OCS. They cannot access keys that are protected by other OCSs.

Operator Cards employ the Security World key to perform a challenge-response protocol with the hardware security device. This means that Operator Cards are only useable by a module that belongs to the same Security World.

Remote Operator

The Remote Operator feature is used to load a key protected by an OCS onto a machine to which you do not have physical access (for example, because it is in a secure area).

The Remote Operator feature enables the secure transmission of the contents of a smart card inserted into the slot of one module (the *attended module*) to another module (the *unattended module*). To transmit to a remote module, you must ensure that:

- the smart card is an Operator Card
- the attended and unattended modules are in the same Security World.

To achieve secure communication channels between the attended and unattended modules, the hardware uses an *impath* (an abbreviation of *intermodule path*), a secure protocol for communication over IP networks. The communication channels between the modules:

- are secure against both eavesdroppers and active adversaries
- can carry arbitrary user data as well as module-protected secrets, such as share data, that pass directly between modules.

See also [Compatibility issues](#) on page 19.

Client cooperation feature

The *client cooperation* feature allows client computers for nShield modules to automatically update the Security World and key data stored on the remote file system (RFS) used by the device. For more information, see [Setting up client cooperation](#) on page 51.

NIST SP800-131

You can create a Security World compliant with NIST (National Institute of Standards and Technology) SP800-131; see [Cipher suite](#) on page 75. For details about NIST SP800-131, see <http://csrc.nist.gov/publications/nistpubs/800-131A/sp800-131A.pdf>.

There is a migration tool available for transferring existing Security World data into an SP800-131 Security World; see [Migrating keys to an SP800-131 Security World](#) on page 104.

Compatibility issues

In order to comply with the latest encryption standards, Thales has adopted an enhanced NIST SP800-131A compliant encryption protocol between nShield Connect HSMs and their clients with Security World support software installed. In some cases, this change may impact the compatibility of network-attached HSMs in environments with mixed HSM deployments.

In most cases where versions of Security World support software of v11.50 or later are deployed in conjunction with v11.40 software or lower, no action is required. However, there are two cases in which it not compatible, meaning that communication cannot be established between the HSM and clients or hosts:

- v11.50 or higher clients communicating with a v11.40 or lower netHSM or nShield Connect, where the HSM client uses an nToken.

- v11.50 or higher netHSM or nShield Connect communicating with a Remote File System (RFS) using v11.40 or lower.

Release version	Image versions		Security World Software ¹ v11.40	Security World Software ¹ v11.50 and later
	nShield Connect	netHSM		
Up to 11.40	Up to image version 0.3.5.	Up to image version 2.11.1.	Supported.	For deployments with nTokens, please upgrade the nShield Connect netimage. As a less preferred option, you can downgrade the client-side software.
11.50 or later	Image 0.3.6 or later.	Image 2.12.3 or later.	RFS and client software upgrade required.	Supported.

¹ Previously known as nCipher Support Software, or nCSS.

FIPS 140-2 compliance

All Security Worlds are compliant with the Federal Information Processing Standards (FIPS) 140-2 specification. The default setting for Security Worlds complies with level 2 of FIPS 140-2.

A Security World that complies with the roles and services section of FIPS 140-2 level 2 does not require any authorization to create an OCS or an application key.

Note All Security Worlds, whether or not they comply with FIPS 140-2, rely on the security features of your operating system to control which users can write data to the host.

FIPS 140-2 level 3 compliance

When you create a Security World, you can choose whether the Security World is compliant with the roles and services section of either:

- FIPS 140-2 at level 2
- FIPS 140-2 at level 3.

The FIPS 140-2 level 3 option is included for those customers who have a regulatory requirement for compliance with FIPS 140-2 at level 3.

If you choose to create a Security World that complies with FIPS 140-2 level 3, the module initializes in strict FIPS mode, complying with the roles and services, key management, and self-test sections of FIPS 140-2 at level 3 as described in its validation certificate.

Before you can create or erase an OCS in a Security World that complies with FIPS 140-2 level 3, you must authorize the action with a card from the ACS or an OCS from that Security World.

Note A Security World only provides a complete level 3 compliant system when used with nShield modules that have the additional physical security coating required by the FIPS 140-2 level 3.

For more details about FIPS 140-2, see <http://csrc.nist.gov/publications/fips/fips140-2/fips1402.pdf>.

Common Criteria Compliance

The nShield module has been evaluated to Common Criteria (ISO in accordance with the Security Target *nShield Family of Hardware Security Modules Firmware Version 2.33.82*).

To ensure that the module has been configured to meet the requirements of this Security Target, you must:

- ensure that the firmware version matches the evaluated version
- create a Security World protected by a suitable ACS for which:
 - the total number of cards in the ACS (N) must be greater than 1
 - the quorum (K) must be less than the total number of cards (N)
 - the quorum (K) must be greater than half the total number (N).

Note ACSs for which K/N is $3/5$ or $4/7$ both meet the criteria for the Security Target. A $3/5$ ACS is sufficient in many cases. A $4/7$ ACS provides additional security.

- choose whether to enable the OCS/softcard replacement and pass phrase replacement options as appropriate to the data to be protected by the Security World (for a description of OCSs, see [Using Operator Card Sets: OCS-protected keys](#) on page 24)

- choose whether to enable the Remote Operator, Real-time clock (RTC), NVRAM, SEE, and Foreign Token operation options for the Security World as necessary.

Note Whether or not these options are enabled for the Security World does not affect the security of keys.

Note You do not need to create a FIPS 140-2 level 3 compliant Security World to meet the requirements of this Security Target.

After you have created the Security World, you must ensure that:

- each card of the ACS is held by a separate individual
- no separate individual can access more than 1 card of the ACS
- all the cards comprising the ACS are stored securely at all times.

If you are using a Security World that meets the requirements of the Security Target *nShield Family of Hardware Security Modules Firmware Version 2.33.82*, you must keep your Operator Cards safe at all times and must report the loss of any cards to an Administrator (that is, an individual holding Administrator Cards) immediately so that they can take appropriate action.

If you are using a Security World that meets the requirements of this Security Target, you are responsible for correctly generating any keys that you require. The Security World management utility **KeySafe** and the **generatekey** command-line utility both verify the security of keys by default. You must not override these security verification features when generating keys.

Platform independence

The Security World is completely platform independent. All key information is stored in a proprietary format that any computer supported by Security World Software can read, regardless of the native format used by that computer. This enables you to:

- safely move a Security World between platforms with differing native formats. For example, you can move a Security World between Windows and Unix-based platforms.
- include hosts running different operating systems in the same Security World.

Note When copying host data between computers using different operating systems or disk formats, use a mechanism that preserves the original data format and line endings [such as **.tar** file archives].

Application independence

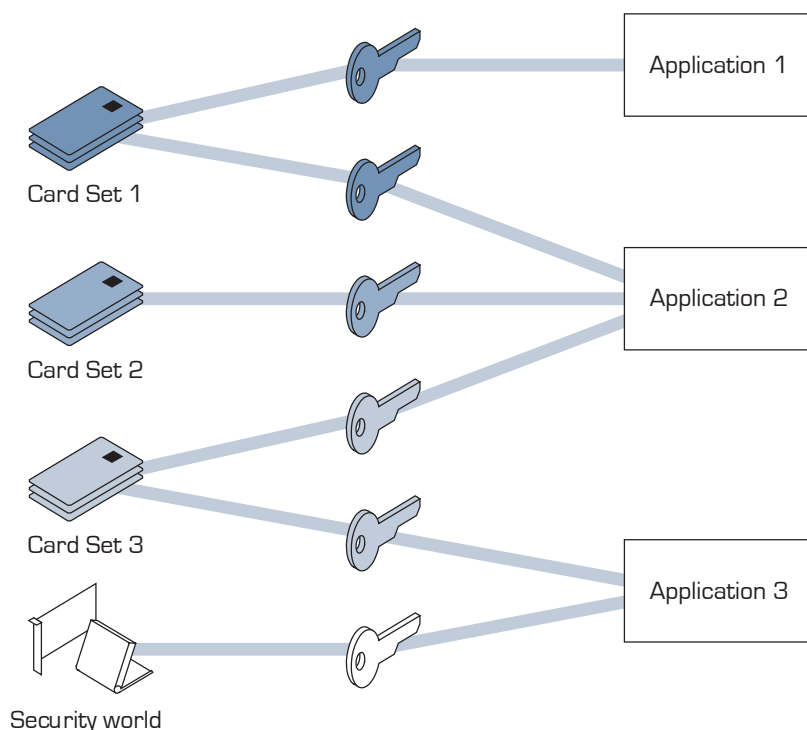
A Security World can protect keys for any applications correctly integrated with the Security World Software. Each key belongs to a specific application and is only ever used by that application. Keys are stored along with any additional data that is required by the application.

You do not need to specify:

- which applications you intend to use. You can add a key for any supported application at any time.
- how the key is used by an application. A Security World controls the protection for the key; the application determines how it is used.

Although keys belong to a specific application, OCSs do not. You can protect keys for different applications using the same OCS (see Figure 2).

Figure 2 Operator Card Sets, keys, and applications



In Figure 2:

- **Card Set 1** protects multiple keys for use with **Application 1** and **Application 2**
- **Card Set 2** protects a single key for use with **Application 2**
- **Card Set 3** protects multiple keys for use with **Application 2** and **Application 3**

- the *Security World* key protects a single key for use with **Application 3**.

Flexibility

Within a Security World, you can choose the level of protection for each application key that you create.

When you create a Security World, a cryptographic key is generated that protects the application keys and OCSs in that Security World. This *Security World* key can be a Triple DES (Data Encryption Standard) key or AES (Advanced Encryption Standard) key. The options are as follows:

Security World key / Cipher suite	Uses
Triple DES	1024-bit DSA key with SHA-1 hash
AES 256 [original]	1024-bit DSA key with SHA-1 hash
AES 256 [SP800-131 compliant]	3072-bit DSA 2 key with SHA-256 hash

Note To create a Triple DES Security World, you must use the **new-world** command-line utility.

Using the Security World key: module-protected keys

You can use the Security World key to protect an application key that you must make available to all your users at all times. This key is called a *module-protected key*. Module-protected keys:

- have no pass phrase
- are usable by any instance of the application for which they were created, provided that this application is running on a server fitted with a hardware security device belonging to the correct Security World.

This level of protection is suitable for high-availability Web servers that you want to recover immediately if the computer resets.

Using Operator Card Sets: OCS-protected keys

An OCS belongs to a specific Security World. Only a hardware security device within the Security World to which the OCS belongs can read, erase or format the OCS. There is no limit to the number of OCSs that you can create within a Security World.

An OCS stores a number of symmetric keys that are used to protect the application keys. These keys are of the same type as the Security World key.

Each card in an OCS stores only a fragment of the OCS keys. You can only re-create these keys if you have access to enough of their fragments. Because cards sometimes fail or are lost, the number of fragments required to re-create the key (K) are usually less than the total number of fragments (N).

To make your OCS more secure, we recommend that you make the value of K relatively large and the value of N less than twice that of K (for example, the values for K/N being 3/5 or 5/9). This practice ensures that if you have a set of K cards that you can use to recreate the key, then you can be certain that there is no other such card set in existence.

Note Some applications restrict K to 1.

Using Operator Card Sets to share keys securely

You can use OCSs to enable the same keys for use in a number of different hardware security devices at the same time, but you must leave one of the cards in each device.

To use OCS-protected keys across multiple modules, set:

- K to 1
- N equal to the number of the modules you want to use.

You can then insert a single card from the OCS into each hardware security device to authorize the use of that key.

To issue the same OCS-protected key to a set of users, set:

- K to 1
- N equal to the number of users.

You can then give each user a single card from the OCS, enabling those users to authorize the use of that key.

Note If you have created an OCS for extra security [in which K is more than half of N], you can still share the keys it protects simultaneously amongst multiple modules as long you have enough unused cards to form a K/N quorum for the additional hardware security devices. For example, with a 3/5 OCS, you can load keys onto 3 hardware security devices because, after loading the key on the first device, you still have 4 cards left. After loading the key on a second device, you still have 3 cards left. After loading the key onto a third device, you have only 2 cards left, which is not enough to create the quorum required to load the key onto a fourth device.

If a card becomes damaged, you can replace the whole OCS if you have authorization from the ACS belonging to that Security World.

Note You can only replace OCSs that were created by Security Worlds that have the OCS/softcard replacement option enabled. For more information, see [OCS and softcard replacement](#) on page 77.

Using Operator Card Sets for high availability

If you cannot risk the failure of a smart card, but some keys must remain accessible at all times, you can create a 1/2 OCS.

Use the first card as the working card and store the second card in a completely secure environment. If the working card fails, retrieve the spare second card from storage, and use it until you re-create a new set of 2 cards (see [Replacing an Operator Card Set or recovering keys to softcards](#) on page 31).

Note You can only replace OCSs that were created by Security Worlds that have the OCS/softcard replacement option enabled. For more information, see [OCS and softcard replacement](#) on page 77.

Using pass phrases for extra security

You can set individual pass phrases for some or all the cards in an OCS.

You can change the pass phrase for a card at any time provided that you have access to the card, the existing pass phrase, and a hardware security device that belongs to the Security World to which the card belongs. For more information, see [Changing card and softcard pass phrases](#) on page 135.

Note Some applications do not support the use of pass phrases.

We set no absolute limit on the length of pass phrases, although individual applications may not accept pass phrases longer than a specific number of characters. Likewise, the Security World does not impose restrictions on which characters you can use in a pass phrase, although some applications may not accept certain characters.

Using persistent Operator Card Sets

If you create a standard (non-persistent) OCS, you can only use the keys protected by that OCS while the last required card of the quorum remains loaded in the smart card reader of the hardware security device. The keys protected by this card are removed from the memory of the hardware security device as soon as the card is removed from the smart card reader.

Although this feature provides added security, it means that only a single user can load keys at any one time (because there is only a single smart card reader for the hardware security device).

If you create a *persistent* OCS, the keys protected by a card from that OCS persist after the card is removed from the smart card reader of the hardware security device. This enables:

- the use of the same smart card in several hardware security devices at the same time
- several users to load keys onto the same hardware security device at the same time.

The Security World Software maintains strict separation between the keys loaded by each user, and each user only has access to the keys protected by their OCS.

Keys protected by a persistent card are automatically removed from the hardware security device:

- when the application that loaded the OCS closes the connection to the hardware security device
- after a time limit that is specified when the card set is created
- when an application chooses to remove a key.

Note Some applications automatically remove a key after each use, reloading it only when required. Such applications do not benefit from persistent OCSs. The only way of sharing keys between hardware security devices for such applications is by having multiple smart cards in an OCS.

Although the hardware security device stores the key, the key is only available to the application that loaded it. To use keys protected by this card in another application, you must re-insert the card, and enter its pass phrase if it has one. Certain applications only permit one user at a time to log in, in which case any previously loaded persistent OCS used in that application is removed before the user is allowed to log in with a new OCS.

You can manually remove all keys protected by persistent cards by clearing the hardware security device. For example, you could:

- run the command **nopclearfail --clear --all**
- press the Clear button of the hardware security device

Any of these processes removes all keys protected by OCSs from the hardware security device, including the card in the reader. In such cases, all users of any applications using the hardware security device must log in again.

Persistence is an immutable property of the OCS. You can choose whether or not to make an OCS persistent at the time of its creation, but you cannot change a persistent OCS into a non-persistent OCS, or a non-persistent OCS into a persistent OCS.

A Security World can contain a mix of persistent and non-persistent card sets.

Using softcard-protected keys

If you want to use pass phrases to restrict key access but avoid using physical tokens (as required by smart-card protection), you can create a *softcard-protected key*.

A *softcard* is a file containing a logical token that you cannot load without a pass phrase. You must load the logical token to authorize the loading of any key that is protected by the softcard. Softcard files:

- are stored in the **kmdata/local** directory
- have names of the form **softcard_***hash* (where *hash* is the hash of the logical token share).

Softcard-protected keys offer better security than module-protected keys and better availability than OCS-protected keys. However, because softcard-protected keys do not require physical tokens to authorize key-loading, OCS-protected keys offer better security than softcard-protected keys.

The pass phrase of a softcard is set when you generate it, and you can use a single softcard to protect multiple keys. Softcards function as persistent 1/1 logical tokens, and after a softcard is loaded, it remains valid for loading its keys until its **KeyID** is destroyed.

Scalability

A Security World is scalable. You can add multiple hardware security devices to a server and share a Security World across multiple servers. You can also add OCSs and application keys at any time. You do not need to make any decisions about the size of the Security World when you create it.

To share a Security World across multiple servers:

- ensure each server has at least one hardware security device fitted
- copy the host data to each server or make it available on a shared disk
- use the recovery and replacement data with the ACS to load the required cryptographic keys securely onto every hardware security device.

To provide access to the same keys on every server, you must ensure that all changes to the data are propagated to the remaining servers. If your servers are part of a cluster, then the tools provided by the cluster should synchronize the data. If the servers are connected by a network, then they could all access the same copy of the data.

There is no risk of an attacker obtaining information by snooping on the network, as the data is only ever decrypted inside a hardware security module. Alternatively, you can maintain copies of the data on different servers.

You can configure nShield modules to:

- access the remote file system (RFS) used by the nShield Connect or netHSM automatically
- share Security World and key data stored in the **/opt/nfast/kmdata/local** directory.

Client hardware security devices that access data in this way are described as *cooperating clients*. For more information, see [Setting up client cooperation](#) on page 51.

Note We provide the **rfs-sync** command-line utility to synchronize the **kmdata** directory between a cooperating client and the remote file system it is configured to access. Run **rfs-sync** whenever a cooperating client is initialized, to retrieve data from the remote file system, and also whenever a client needs to update its local copy of the data [or, if the client has write access, to commit changes to the data].

Load sharing

If you have more than one hardware security device on your system and you load the same key onto each device, your applications (that have been integrated with the Security World Software) can make use of the load sharing features in the Security World Software to share the cryptography between them.

Note It is up to the application to implement load sharing. Some applications may not be able to make use of this feature.

Robustness

Cryptography must work 24 hours a day, 7 days a week, in a production environment. If something does go wrong, you must be able to recover without compromising your security. A Security World offers all of these features.

Backup and recovery

The Security World data stored on the host is encrypted using the Security World key.

You should regularly back up the data stored in the Key Management Data directory with your normal backup procedures. It would not matter if an attacker obtained this data because it is worthless without the Security World key, stored in your hardware security device, and the Administrator cards for that Security World.

When you create a Security World, it automatically creates recovery data for the Security World key. As with all host data, this is encrypted with the same type of key as the Security World key. The cryptographic keys that protect this data are stored in the ACS. The keys are split among the cards in the ACS using the same K/N mechanism as for an OCS. The ACS protects several keys that are used for different operations.

The cards in the ACS are only used for recovery and replacement operations and for adding extra hardware security devices to a Security World. At all other times, you must store these cards in a secure environment.

Note In strict FIPS 140-2 Level 3 Security Worlds, the ACS or an OCS is needed to control many operations, including the creation of keys and OCSs.

Replacing a hardware security device

If you have a problem with a hardware security device, you can replace it with a new hardware security device by using the ACS and the recovery data to load the Security World key securely. Use the same mechanism to reload the Security World key if you need to upgrade the firmware in the hardware security device or if you need to add extra hardware security devices to the Security World.

For information about replacing a hardware security device, see [Adding or restoring a module to the Security World](#) on page 94.

Replacing the Administrator Card Set

If you lose one of the smart cards from the ACS, or if the card fails, you must immediately create a replacement set using either:

- the KeySafe Replace Administrator Card Set option
- **racs** utility (see [Replacing the Administrator Card Set](#) on page 151).

The hardware security device does not store recovery data for the ACS. Provided that K is less than N for the ACS, the hardware security device can re-create all the keys stored on the device even if the information from one of the cards is missing.

The loss or failure of one of the smart cards in the ACS means that you must replace the ACS. However, you cannot replace the ACS unless you have:

- the required number of current cards

- access to their pass phrases.



Although replacing the ACS deletes the copy of the recovery data on your host, you can still use the old ACS with the old host data, which you may have stored on backup tapes and other hosts. To eliminate any risk this may pose, we recommend erasing the old ACS as soon as you create a new ACS.

Replacing an Operator Card Set or recovering keys to softcards

If you lose an Operator Card, you lose all the keys that are protected by that card. To prevent this, you have the option to store a second copy of the working key that the recovery key protects in a Security World. Similarly, you can recover keys protected by one softcard to another softcard.

Note The ability to replace an OCS is an option that is enabled by default during Security World creation (see [OCS and softcard replacement](#) on page 77). You can only disable the OCS replacement option during the Security World creation process. You cannot restore the OCS replacement option, or disable this option, after the creation of the Security World.

Note You can only recover keys protected by an OCS to another OCS, and not to a softcard. Likewise, you can only recover softcard-protected keys to another softcard, and not to an OCS.

To create new copies of the keys protected by the recovery key on a given card set, and to recover keys protected by one softcard to another softcard, use the **rocs** command-line utility.

The security of recovery and replacement data

Replacing OCSs and softcards requires authorization. To prevent the duplication of an OCS or a softcard without your knowledge, the recovery keys are protected by the ACS.

However, there is always some extra risk attached to the storage of any key-recovery or OCS and softcard replacement data. An attacker with the ACS and a copy of the recovery and replacement data could re-create your Security World. If you have some keys that are especially important to protect, you may decide:

- to issue a new key if you lose the OCS that protects the existing key
- turn off the recovery and replacement functions for the Security World or the recovery feature for a specific key.

You can only generate recovery and replacement data when you create the Security World or key. If you choose not to create recovery and replacement data at this point, you cannot add this data later. Similarly, if you choose to create recovery and replacement data when you generate the Security World or key, you cannot remove it securely later.

If you have not allowed recovery and replacement functionality for the Security World, then you cannot recover any key in the Security World (regardless of whether the key itself was created as recoverable).

The recovery data for application keys is kept separate from the recovery data for the Security World key. The Security World always creates recovery data for the Security World key. It is only the recovery of application keys that is optional.

KeySafe and Security Worlds

KeySafe provides an intuitive and easy-to-use graphical interface for managing Security Worlds. KeySafe manages the Security World and the keys protected by it. For more information about using KeySafe, see Appendix A: [Using KeySafe](#).

Note Most applications store only their long-term keys in the Security World. Session keys are short term keys generated by the application which are not normally loaded into the Security World.

Although you use KeySafe to generate keys, it is your chosen application that actually uses them. You do not need KeySafe to make use of the keys that are protected by the Security World. For example, if you share a Security World across several host computers, you do not need to install KeySafe on every computer. To manage the Security World from a single computer, you can install KeySafe on just that one computer even though you are using the Security World data on other machines.

KeySafe enables you to:

- create a Security World and its ACS, either FIPS 140-2 level 2 or level 3

Note This option provides compliance with the roles and services of the FIPS 140-2 level 3 standard. It is included for those customers who have a regulatory requirement for compliance.

- add a hardware security device to a Security World
- remove a hardware security device from a Security World
- replace an ACS
- create OCSs
- list the OCSs in the current Security World
- change the pass phrase on an Operator Card
- remove a lost OCS from a Security World

- replace OCSs
- erase an Operator Card
- add a new key to a Security World
- import a key into a Security World
- list the keys in the current Security World
- delete a key from a Security World.

KeySafe does not provide tools to back up and restore the host data or update hardware security device firmware, nor does KeySafe provide tools to synchronize host data between servers. These functions can be performed with your standard system utilities.

In addition to KeySafe, we also supply command-line utilities to manage the Security World; for more information about the supplied utilities, see Appendix B: [Supplied utilities](#). Current versions of these tools can be used interchangeably with the current version of KeySafe.

Applications and Security Worlds

A Security World can protect keys for a range of industry standard applications. For details of the applications that are currently supported, visit <http://www.thalesgroup.com/iss>.

We have produced Integration Guides for many supported applications. The Integration Guides describe how to install and configure an application so that it works with Thales hardware security devices and Security Worlds.

For more information about the Thales range of Integration Guides:

- visit <http://iss.thalesgroup.com/Resources.aspx>
- contact Support.

The nCipher PKCS #11 library and Security Worlds

Many applications use a PKCS (Public Key Cryptography Standard) #11 library to generate and manage cryptographic keys. We have produced an nCipher version of the PKCS #11 library that uses the Security World to protect keys.

Enabling a PKCS #11 based application to use nShield hardware key protection involves configuring the application to use the nCipher PKCS #11 library.

A PKCS #11 token created by the nCipher PKCS #11 library is a Security World OCS.

The current PKCS #11 standard only supports tokens that are part of a 1-of- N card set.

A Security World does not make any distinction between different applications that use the nCipher PKCS #11 library. Therefore, you can create a key in one PKCS #11 compliant application and make use of it in a different PKCS #11 compliant application.

Risks

Even the best-designed tools cannot offer security against every risk. Although a Security World can control which user has access to which keys, it cannot prevent a user from using a key fraudulently. For example, although a Security World can determine if a user is authorized to use a particular key, it cannot determine whether the message that is sent with that key is accurate.

A Security World can only manage keys that were created inside the Security World. Keys created outside a Security World, even if they are imported into the Security World, may remain exposed to a security risk.

Most failures of security systems are not the result of inherent flaws in the system, but result from user error. The following basic rules apply to any security system:

- Keep your smart cards safe.
- Always obtain smart cards from a trusted source: from Thales or directly from the smart card manufacturer.
- Never insert a smart card used with Thales key management products into an untrusted smart card reader.
- Never insert any untrusted smart card into your hardware security device.
- Never tell anyone your pass phrase.
- Never write down your pass phrase.
- Never use a pass phrase that is easy to guess.
- Only use the ACS in modules connected to trusted hosts.

Note If you have any doubts about the security of a key and/or Security World, replace that key and/or Security World with a newly generated one.



Chapter 3: Support software installation

This chapter provides a detailed overview of the processes required for installing Security World Software on the client or RFS of your nShield module.

After installing Security World Software, you must complete additional configuration and setup tasks (including Security World and OCS creation) before you can use your nShield module and support software to protect and manage your keys. For more information, see [After software installation and testing](#) on page 50.

For more information about the type of user that is required for different operations, see [About user privileges](#) on page 51.

Note If you are installing Security World Software for the first time, or want to complete a basic nShield installation quickly, see the *Quick Start Guide*.

Before you install the software

Before you install and configure the software, you must:

- install the module. For more information, see the *Quick Start Guide*.
- uninstall any existing older versions of Security World Software, as described in [Uninstall existing Software](#) on page 35
- complete any other necessary preparatory tasks, as described in [Other preparatory tasks before software installation](#) on page 40.

Uninstall existing Software

We recommend that you uninstall any existing older versions of Security World Software before you install new support software.

The automated Security World Software installers *do not* delete other components or any key data and Security World data that you have created. However, a manual installation using **.tar** files *does* overwrite existing data and directories.

The uninstaller removes only those files that were created during the installation. To remove key data or Security World data, delete the files in the Key Management Data directory.

Note Environment variables, if set, are stored in the configuration file `/etc/nfast.conf`. The uninstall process may not delete `/etc/nfast.conf`. If you are re-installing the support software over an old installation, examine this file, and remove any lines that are no longer relevant. For information about environment variables, see [Setting environment variables](#) on page 56 and [Environment variables](#) on page 245.

Note To remove an existing Security World before uninstalling the Security World Software, run the command `createocs -m# -s0 --erase` (where # is the module number) to erase the OCS before you erase the Security World or uninstall the Security World Software. After you have erased a Security World, you can no longer access or erase any cards that belonged to it.

Note The file `nCipherKM.jar`, if present, is located in the extensions folder of your local Java Virtual Machine. The uninstall process may not delete this file. Before reinstalling over an old installation, remove the `nCipherKM.jar` file. To locate the Java Virtual Machine extensions folder, see [Installing the nCipherKM JCA/JCE CSP](#) on page 157.



We recommend that you do not uninstall the Security World Software unless either you are certain it is no longer required or you intend to upgrade it.

To uninstall the Security World Software, follow the steps for your Unix-based platform: Solaris, AIX, HP-UX, or Linux.

Uninstalling on Solaris

To uninstall the Security World Software from Solaris:

- 1 Assume the nFast Administrator privileges or root privileges by running the command:

```
$ su -
```

- 2 Type your password, then press `[Enter]`.

- 3 To remove drivers, install fragments, and scripts and to stop services, run the command:

```
/opt/nfast/sbin/install -u
```

- 4 Remove the software package by running the command:

```
/usr/sbin/pkgrm
```

The command displays the list of components that were installed.

- 5 Select all the nShield packages (prefixed with the letters **NC**), then press **Enter**.
- 6 Follow the onscreen instructions, confirming the uninstallation of packages as prompted.
- 7 If you are not planning to re-install the product, delete the configuration file **/etc/nfast.conf** if it exists.

Note Do not delete the configuration file if you are planning to re-install the product.

If required, you can safely remove the module after shutting down all connected hardware.

Uninstalling on AIX

To uninstall the Security World Software from AIX:

- 1 Log in as a user with root privileges.
- 2 To remove drivers, install fragments, and scripts and to stop services, run the command:

```
/opt/nfast/sbin/install -u
```

- 3 Start the software management tool by running the command:

```
smit install_remove
```

- 4 For **SOFTWARE name**, select **List** to list all available file sets, and then select all those prefixed with **ncipher**.
- 5 Press **Enter** to confirm the selected file sets for uninstallation.

The Remove Installed Software panel is displayed.
- 6 Ensure that the **PREVIEW Only** option is set to **No** (or the removal operation does not occur), and press **Enter**.
- 7 When prompted to confirm that you are sure about the removal, press **Enter** again to start the uninstall process.

- 8 If you are not planning to re-install the product, delete the configuration file `/etc/nfast.conf` if it exists.

Note Do not delete the configuration file if you are planning to re-install the product.

If required, you can safely remove the module after shutting down all connected hardware.

Uninstalling on HP-UX

To uninstall the Security World Software from HP-UX:

- 1 Assume the nFast Administrator privileges or root privileges by running the command:

```
$ su -
```

- 2 Type your password, then press **Enter**.

- 3 To remove drivers, install fragments, and scripts and to stop services, run the command:

```
/opt/nfast/sbin/install -u
```

- 4 Remove the software packages by running the command:

```
/usr/sbin/swremove
```

The command displays the list of components that were installed.

- 5 Select all the packages.
- 6 Select **Remove** from the **Actions** menu.
- 7 When the analysis is complete, if there are no errors, click **OK**. If the installer reports any errors, click **Logfile** to display them.
- 8 When the uninstaller asks you to confirm that you want to remove this product, click **Yes**.
- 9 When the uninstallation is complete, click **Done**.
- 10 If you are not planning to re-install the product, delete the configuration file `/etc/nfast.conf` if it exists.

Note Do not delete the configuration file if you are planning to re-install the product.

If required, you can safely remove the module after shutting down all connected hardware.

Uninstalling on Linux

To uninstall the Security World Software from Linux:

- 1 Assume the nFast Administrator privileges or root privileges by running the command:

```
$ su -
```

- 2 Type your password, then press **Enter**.

- 3 To remove drivers, install fragments, and scripts and to stop services, run the command:

```
/opt/nfast/sbin/install -u
```

- 4 Delete all the files (including those in subdirectories) in **/opt/nfast** and **/dev/nfast/** by running the following commands:

```
rm -rf /opt/nfast
```

```
rm -rf /dev/nfast
```

Note Deleting all the files and subdirectories in **/opt/nfast** also deletes the **/opt/nfast/kmdata** directory. To be able to restore an existing Security World after deleting all the files in **/opt/nfast**, ensure you have made a backup of the **/opt/nfast/kmdata** directory in a safe location before deleting the original.

- 5 If you are not planning to re-install the product, delete the configuration file **/etc/nfast.conf** if it exists.

Note Do not delete the configuration file if you are planning to re-install the product.

- 6 Unless needed for a subsequent installation, remove the user **nfast** and, if it exists, the user **ncsnmpd**:
 - a Open the file **/etc/group** with a text editor.
 - b Remove the line that begins with the form:

```
nfast:x:n
```

In this line, *n* is an integer.

- c Open the file **/etc/passwd** with a text editor.
- d Remove the line that begins with the form:

```
nfast:x:...
```

- e If it exists, remove the line that begins with the form:

```
ncsnmpd:x:...
```

If required, you can safely remove the module after shutting down all connected hardware.

Other preparatory tasks before software installation

Perform any of the necessary preparatory tasks described in this section before installing the Security World Software.

Install perl

Ensure that you have installed **perl** on the client computer.

Install operating environment patches

Ensure that you have installed the latest recommended patches. See the documentation supplied with your operating environment for information.

Install Java with any necessary patches

Java software is available from <http://java.sun.com/>. If your security policy does not allow the use of downloaded software, these components can be obtained on removable media from Sun or from your operating system vendor.

The Security World Software supports JRE/JDK version 1.4.2, 1.5, and 1.6. The nCipherKM JCA/JCE CSP supports JRE/JDK 6.0

To use KeySafe, you must have installed JRE/JDK 1.4.2, 1.5, and 1.6. We recommend that you install Java before you install the Security World Software. The Java executable must be on your path.

To use the Security World Software's Java components, you may need to install patches supplied by your operating system manufacturer. Refer to the Sun documentation supplied with your Java installation.

Create a symbolic link to any alternate installation directory

If you want the Security World Software to be installed in a directory different from the default **/opt/nfast/** directory, you must create a symbolic link from **/opt/nfast/** to the directory where you want the software installed.

Identify software components to be installed

We supply standard component bundles that contain many of the necessary components for your installation and, in addition, individual components for use with supported applications. To be sure that all component dependencies are satisfied, you can install either:

- all the software components supplied
- only the software components you require.

During the installation process, you are asked to choose which bundles and components to install. Your choice depends on a number of considerations, including:

- the types of application that are to use the module
- the amount of disk space available for the installation

- your company's policy on installing software. For example, although it may be simpler to choose all software components, your company may have a policy of not installing any software that is not required.

Note You *must* install the **hwsp** bundle. If the **hwsp** bundle is not installed, your module cannot function.

Note The **nfdrv Kernel device drivers** component, required if you are using a Thales PCI or PCIe module, is installed as part of the **hwsp Hardware Support** bundle.

Note You *must* install the **nfdrv** component if you are using a Thales PCI or PCIe module.

We recommend that you always install the **ctls Core Tools bundle**. This bundle contains all the Security World Software command-line utilities, including:

- **generatekey**
- low level utilities
- test programs.

Note The Core Tools bundle includes the **tcsrc Tcl run time** component that installs a run-time Tcl installation within the nShield directories. This is used by the tools for creating the Security World and by **KeySafe**. This does not affect any other installation of Tcl on your computer.

Ensure that you have identified any optional components that you require before you install the Security World Software. For more information about optional components, see Appendix C: [Components on Security World Software DVD-ROMs](#).

Install the Security World Software

After determining requirements and completing preparatory tasks, you are ready to:

- 1 Install the Security World Software, as described in this section.
- 2 Test the installation to ensure the software has been installed correctly, as described in [Test the installation](#) on page 48.

To install the Security World Software, follow the steps that correspond to your Unix-based platform: Solaris, AIX, HP-UX, or Linux.

Note Before installing the Security World Software, ensure that you have uninstalled any previous installation as described in [Uninstall existing Software](#) on page 35.

Note In the following instructions, *disc-name* is the name of the mount point of the DVD-ROM.

Installing on Solaris

To install the Security World Software for Solaris:

- 1 Log in as a user with root privileges.
- 2 Place the DVD-ROM in the optical disc drive, and mount the drive.
- 3 To install the Security World Software server, run the command:

```
/usr/sbin/pkgadd -d /cdrom/disc-name/solaris/2_7/nfast/nfast.pkg
```

- 4 From the list of packages available for installation, select all the packages, then press **Enter**.
- 5 Run the install script by using the following command:

```
/opt/nfast/sbin/install
```

After the software is installed, you are returned to the shell prompt.

- 6 Add **/opt/nfast/bin** to your **PATH** system variable:

- if you use the Bourne shell, add these lines to your system or personal profile:

```
PATH=/opt/nfast/bin:$PATH
export PATH
```

- if you use the C shell, add this line to your system or personal profile:

```
setenv PATH /opt/nfast/bin:$PATH
```

Installing on AIX

To install the Security World Software for AIX:

- 1 Log in as a user with root privileges.
- 2 Place the DVD-ROM in the optical disc drive, and mount the drive.
- 3 Start the software management tool by running the command:

```
smit install_latest
```

- 4 Select **List** to display the input device or directory for the software, and select the location that contains the installation image.
- 5 For **SOFTWARE to install**, select **List**, and then select all the file sets. For information about the component bundles and the additional software supplied on your DVD-ROM, see Appendix C: [Components on Security World Software DVD-ROMs](#).
- 6 Press **Enter** to confirm the file set selection.

When additional installation options are displayed, leave the default settings enabled. Press **Enter** to confirm these settings, and then press **Enter** again to begin the installation.

- 7 After software installation is complete, run the install script with the following command:

```
/opt/nfast/sbin/install
```

- 8 Add **/opt/nfast/bin** to your **PATH** system variable:

- if you use the Bourne shell, add these lines to your system or personal profile:

```
PATH=/opt/nfast/bin:$PATH
export PATH
```

- if you use the C shell, add this line to your system or personal profile:

```
setenv PATH /opt/nfast/bin:$PATH
```

Installing on HP-UX

To install the Security World Software for HP-UX:

- 1 Log in as **root**.
- 2 Place the DVD-ROM in the optical disc drive, and mount the drive, using the **-o cdcase** option.
- 3 Open a terminal window, and start the software management tool by running a command of the form:

```
swinstall -s disc-name/hpux/ver/nfast/nfast.dep
```

In this example, *disc-name* is the mount point of the DVD-ROM and *ver* is the version of HP-UX (for example, use **11_23** for HP-UX version 11.23).

- 4 Select all the software bundles and components for installation. For information about the component bundles and the additional software supplied on your DVD-ROM, see Appendix C: [Components on Security World Software DVD-ROMs](#).
- 5 Select **Install** from the **Actions** menu.
- 6 When the installation analysis is complete, click **OK**. If the installer reports any errors, click **Logfile** to display them.
- 7 Click **Yes** to confirm you want to install.
- 8 The installer now installs the selected products. When it is complete, click the **Done** button.
- 9 Log in as **root**.
- 10 Run the install script by using the following command:

```
/opt/nfast/sbin/install
```

- 11 Add **/opt/nfast/bin** to your **PATH** system variable:

- if you use the Bourne shell, add these lines to your system or personal profile:

```
PATH=/opt/nfast/bin:$PATH
export PATH
```

- if you use the C shell, add this line to your system or personal profile:

```
setenv PATH /opt/nfast/bin:$PATH
```

Installing on Linux

To install the Security World Software for Linux:

- 1 Log in as a user with root privileges.
- 2 Place the DVD-ROM in the optical disc drive, and mount the drive.
- 3 Open a terminal window, and change to the root directory.
- 4 Extract the required **.tar** files to install all the software bundles by running commands of the form:

```
tar xf disc-name/linux/ver/nfast/bundle/file.tar
```

In this command, **ver** is the version of the operating system (for example, **libc6_3**), **bundle** is the directory name of a given bundle (for example, **hwsp** or **ctls**), and **file.tar** is the name of a **.tar** file within a bundle directory.

Note Some directories contain more than one **.tar** file

For information about the component bundles and the additional software supplied on your DVD-ROM, see Appendix C: [Components on Security World Software DVD-ROMs](#)

- 5 To use an nShield module with your Linux system, you must build a kernel driver. Thales supplies the source to the nCipher PCI kernel driver (**nfp**) and a makefile for building the driver as a loadable module.

The kernel level driver is installed as part of the **hwsp** bundle. To build the driver with the supplied makefile, you must have the correct headers installed for the kernel that you are running. They must be headers for the same version of the kernel and must contain the kernel configuration options with which your kernel was built. You must also have appropriate versions of **gcc**, **make**, and your C library's development package.

The configuration script looks for the kernel headers in the default directory **/lib/modules/`uname -r`/build/include**. If your kernel headers are located in a different directory, set the **KERNEL_HEADERS** environment variable so that they are in

\$KERNEL_HEADERS/include/. Historically, the headers have resided in **/usr/src/linux/include/**. If the headers for your kernel are not already installed, install them from your Linux distribution disk, or contact your kernel supplier.

Build the driver as a loadable kernel module. When you have ensured the correct headers are in place, perform the following steps to use the makefile:

- a Change directory to the nCipher PCI driver directory by running the command:

```
# cd /opt/nfast/driver/
```

- b Configure the source by running the command:

```
# ./configure
```

- c Make the driver by running the command:

```
# make
```

This produces a driver file that is automatically loaded as part of the normal installation process.

- 6 Run the install script by using the following command:

```
/opt/nfast/sbin/install
```

- 7 The first time the installer runs (before any users or groups have been created), it displays a message about creating the **nfast** user.

```
User 'nfast' or group 'nfast' do not exist. To create the 'nfast' user,
in group 'nfast', with home directory /opt/nfast please select one of the
following options:
```

- ```
 1) Let this script run groupadd to create the group 'nfast',
 then useradd to add the user 'nfast' to the group 'nfast'
 with home as '/opt/nfast'
 (expected to work with most Linux distributions).
 2) Exit the script so you can create the user and group manually.
```

```
 Rerun the install script when this is complete.
```

```
Please select option 1 or 2
```

---

To enable the installer to create the user, group, and directory for you, type 1: to create the user, group and directory yourself, type 2. Then press **Enter**.

## 8 The first time the installer runs, it also displays a message about creating the **ncsnmpd** user.

---

```
User 'ncsnmpd' or group 'ncsnmpd' do not exist. To create the 'ncsnmpd' user,
in group 'ncsnmpd', with home directory /opt/nfast please select one of the
following options:
```

- 1) Let this script run groupadd to create the group 'ncsnmpd',  
then useradd to add the user 'ncsnmpd' to the group 'ncsnmpd'  
with home as '/opt/nfast'  
(expected to work with most Linux distributions).
- 2) Exit the script so you can create the user and group manually.  
Rerun the install script when this is complete.

```
Please select option 1 or 2
```

---

To enable the installer to create the user, group, and directory for you, type 1: to create the user, group and directory yourself, type 2. Then press **Enter**.

## 9 Log in to your normal account.

## 10 Add **/opt/nfast/bin** to your **PATH** system variable:

- if you use the Bourne shell, add these lines to your system or personal profile:

---

```
PATH=/opt/nfast/bin:$PATH
export PATH
```

---

- if you use the C shell, add this line to your system or personal profile:

---

```
setenv PATH /opt/nfast/bin:$PATH
```

---

# Test the installation

To check that the software has been installed correctly:

## 1 Log in as a regular user and open a command window.



## 2 Run the command:

---

```
enquiry
```

---

A successful **enquiry** command returns output of the following form:

---

```
server:
enquiry reply flags none
enquiry reply level Six
serial number #####-####-####-####
mode operational
version #.#.#
speed index ###
rec. queue ##..##
...
version serial #
remote server port ####

Module ##:
enquiry reply flags none
enquiry reply level Six
serial number #####-####-####-####
mode operational
version #.#.#
speed index ###
rec. queue ##..##
...
rec. LongJobs queue ##
SEE machine type PowerPCSF
```

---

If the **mode** is **operational** the module has been installed correctly.

If the **mode** is **initialization** or **maintenance**, the module has been installed correctly, but you must change the mode to **operational**. For information about changing the module mode, see Appendix I: [Checking and changing module mode](#)

If the output from the **enquiry** command says that the module is not found, first restart your computer, then re-run the **enquiry** command.

## After software installation and testing

After you have successfully installed and tested the Security World Software (as described in this chapter or, more succinctly, in the *Quick Start Guide*), complete the following steps to finish preparing your module for use:

- 1 If necessary, perform additional software and module configuration tasks (described in Chapter 4: [Software and module configuration](#)):
  - Set up client configuration, as described in [Setting up client cooperation](#) on page 51
  - Set nShield specific environment variables, as described in [Setting environment variables](#) on page 56
  - Configure logging and debugging parameters, as described in [Logging and debugging](#) on page 56
  - Configure Java support for KeySafe, as described in [Configuring Java support for KeySafe](#) on page 57
  - Configure the hardserver, as described in [Configuring the hardserver](#) on page 57.
- 2 Create and configure a Security World, as described in [Creating a Security World](#) on page 71.
- 3 Create an OCS, as described in [Creating Operator Card Sets \(OCSs\)](#) on page 116.

**Note** For a brief explanation of how to create a basic Security World and OCS, see the *Quick Start Guide*.



## Chapter 4: Software and module configuration

This chapter describes software and module configuration tasks that you can choose to perform after the initial installation of Security World Software and hardware (as described in Chapter 3: [Support software installation](#) and the *Quick Start Guide*).

You must determine whether particular configuration options are necessary or appropriate for your installation. The additional configuration options described in this chapter can be performed either before or after the creation of a Security World (as described in [Creating a Security World](#) on page 71) and an OCS (as described in [Creating Operator Card Sets \(OCSs\)](#) on page 116).

### About user privileges

Cryptographic security does not depend on controlling user privileges or access but maintaining the integrity of your system from both deliberate or accidental acts can be enhanced by appropriate use of (OS) user privileges.

### Setting up client cooperation

You can allow any nShield module to access the remote file system (RFS) used by the nShield Connect or netHSM automatically and to share Security World and key data stored in the Key Management Data directory. Client hardware security devices that access data in this way are described as *cooperating clients*.

To configure client cooperation for hardware security devices that are not nShield Connect HSMs:

- 1 Configure the RFS used by your nShield Connect or netHSM to accept access by cooperating clients.

2 On each client that is to be a cooperating client, you must run the **rfs-sync** command-line utility with appropriate options:

- for clients that use a local **K<sub>NETI</sub>** for authorization (which is generated when the module is first initialized from factory state) and which are to be given write access to the RFS, run the command:

---

```
rfs-sync --setup rfs_IP_address
```

---

- for clients that do not have a local **K<sub>NETI</sub>** and require write access, run the command:

---

```
rfs-sync --setup --no-authenticate rfs_IP_address
```

---

**Note** The **rfs-sync** utility uses lock files to ensure that updates are made in a consistent fashion. If an **rfs-sync --commit** operation (the operation that writes data to the remote file system) fails due to a crash or other problem, it is possible for a lock file to be left behind. This would cause all subsequent operations to fail with a lock time-out error.

The **rfs-sync** utility has options for querying the current state of the lock file, and for deleting the lock file; however, we recommend that you do not use these options unless they are necessary to resolve this problem. Clients without write access cannot delete the lock file.

For more information about the **rfs-sync** utility, see [rfs-sync](#) on page 54.

To remove a cooperating client so the RFS no longer recognizes it, you must:

- know the IP address of the cooperating client that you want to remove

- manually update the **remote\_file\_system** section of the hardserver configuration file by removing the following entries for that particular client:

---

```
remote_ip=client_IP_addressremote_esn=
keyhash=00
native_path=/opt/nfast/kmdata/local
volume=kmdata-local
allow_read=yes
allow_write=yes
allow_list=yes
is_directory=yes
is_text=no
```

---

and

---

```
remote_ip=client_IP_addressremote_esn=
keyhash=00
native_path=/opt/nfast/kmdata/localsync-store
volume=kmdata-backup
allow_read=yes
allow_write=yes
allow_list=yes
is_directory=yes
is_text=no
```

---

## Useful utilities

### anonkneti

To find out the ESN and the hash of the **K<sub>NETI</sub>** key for a given IP address, use the **anonkneti** command-line utility. A manual double-check is recommended for security.

## rfs-sync

This utility synchronises the **kmdata** between a cooperating client and the remote file system it is configured to access. It should be run when a cooperating client is initialised in order to retrieve data from the remote file system and also whenever a client needs to update its local copy of the data or, if the client has write access, to commit changes to the data.

### Usage

---

```
rfs-sync [-U|--update] [-c|--commit] [-s|--show] [--remove] [--setup setup_options] ip_address
```

---

### Options

#### **-U, --update**

These options update local key-management data from the remote file system.

**Note** If a cooperating client has keys in its **kmdata/local** directory that are also on the remote file system, if these keys are deleted from the remote file system and then **rfs-sync --update** is run on the client, these keys remain on the client they are until manually removed.

#### **-c, --commit**

These options commit local key-management data changes to the remote file system.

#### **-s, --show**

These options display the current synchronisation configuration.

**--setup**

This option sets up a new synchronisation configuration. Specifics of the configuration can be altered using *setup\_options* as follows:

**-a, --authenticate**

These set-up options specify use of KNETI authentication. This is the default.

**--no-authenticate**

This set-up option specifies that KNETI authentication should not be used.

**-m, --module=*module***

These options select which module to use for KNETI authorisation. The default is module 1. This option can only be used with the **--authenticate** option.

**-p, --port=*port***

These options specify the port on which to connect to the remote file system. The default is 9004.

***ip\_address***

This option specifies the IP address of the remote file system.

**--remove**

This option removes the synchronisation configuration.

A client can use **rfs-sync --show** to display the current configuration, or **rfs-sync --remove** to revert to a standalone configuration. Reverting to a standalone configuration leaves the current contents of the Key Management Data directory in place.

The **rfs-sync** command also has additional administrative options for examining and removing lock files that have been left behind by failed **rfs-sync --commit** operations. Using the **--who-has-lock** option displays the task ID of the lock owner. As a last resort, you can use the **rfs-sync** command-line utility to remove lock files. In such a case, the **--kill-lock** option forcibly removes the lock file.

**Note** The lock file can also be removed via menu item 3-3-2, **Remove RFS Lock**: this executes the **rfs-sync --kill-lock** command.

## Setting environment variables

This section describes how to set Security World Software-specific environment variables. You can find detailed information about the environment variables used by Security World Software in Appendix D: [Environment variables](#).

You can set Security World Software-specific environment variables in the file `/etc/nfast.conf`. This file is not created by the installation process: you must create it yourself.

**Note** Ensure that all variables are exported as well as set.

## Logging and debugging

**Note** The current release of Security World Software uses controls for logging and debugging, that differ from those used in previous releases. However, settings you made in previous releases to control logging and debugging are still generally supported in the current release, although in some situations the output is now formatted differently.

The Security World Software generates logging information that is configured through a set of four environment variables:

- **NFLOG\_FILE**
- **NFLOG\_SEVERITY**
- **NFLOG\_DETAIL**
- **NFLOG\_CATEGORIES**

**Note** If none of these logging environment variables are set, the default behavior is to log nothing, unless this is overridden by any individual library. If any of the four logging variables are set, all unset variables are given default values.

Detailed information about controlling logging information by means of these environment variables is supplied in Appendix E: [Logging, debugging, and diagnostics](#).

Some components of the Security World Software generate separate debugging information which you can manage differently. If you are setting up the module to develop software that uses it, you should configure debugging before commencing software development.



## Configuring Java support for KeySafe

To use KeySafe, you must add the **nfjava**, **kmjava**, and **keysafe** classes to your Java class path after Security World Software installation is complete. For more information about KeySafe, see Appendix A: [Using KeySafe](#).

## Configuring the hardserver

The hardserver handles secure transactions between the modules connected to the host computer and applications that run on the host computer. In addition, the hardserver controls any remote slots that the module uses and loads any SEE (Secure Execution Engine) machines that are to run on the module.

The hardserver can handle transactions for multiple modules. This does not require configuration of the hardserver. For more information, see [Using multiple modules](#) on page 68.

The hardserver must be configured to control:

- the way the hardserver communicates with remote modules
- the way the hardserver communicates with local modules
- the import and export of remote slots
- the loading of SEE machines on to the module when the hardserver starts up.

The hardserver configuration file defines the configuration of the hardserver. By default, it is stored in the **/opt/nfast/kmdata/config** directory, and a default version of this file is created when the Security World Software is installed. See [Overview of hardserver configuration file sections](#) on page 58 for an overview of the hardserver configuration file, and see Appendix F: [Hardserver configuration files](#) for detailed information about the various options available through it.

**Note** In some previous releases of the Security World Software, hardserver configuration was controlled by environment variables. The use of these variables has been deprecated. If any of these environment variables are still set, they override the settings in the configuration file.

You must load the configuration file for the changes to the configuration to take effect.

To configure the hardserver, follow these steps:

- 1 Save a copy of the configuration file **/opt/nfast/kmdata/config/config** so that the configuration can be restored if necessary.

- 2 Edit the configuration file `/opt/nfast/kmdata/config/config` to contain the required configuration. (See Appendix F: [Hardserver configuration files](#) for descriptions of the options in the configuration file.)
- 3 Run the **cfg-reread** command-line utility to load the new configuration.

**Note** If you changed the `server_startup` section of the hardserver configuration file, you must restart the hardserver instead of running **cfg-reread**. For more information, see [Stopping and restarting the hardserver](#) on page 70.

- 4 Test that the hardserver is configured correctly by running the **enquiry** command-line utility.  
Check that a module with the correct characteristics appears in the output.
- 5 Test that the client has access to the Security World data by running the **nfkminfo** command-line utility.

Check that a module with the correct ESN appears in the output and has the state **0x2 Usable**.

## Overview of hardserver configuration file sections

### Configuring remote module connections

A *remote module* is a module that is not connected directly to the host computer but with which the hardserver can communicate. It can be one of the following:

- a network-connected Thales module that is configured to use the host computer as a client computer.
- a module to which an attended remote slot is imported for the hardserver's unattended local module.

You configure the hardserver's communications with remote modules in the **server\_remotecomms** section of the hardserver configuration file. This section defines the port on which the hardserver listens for communications from remote modules. You need to edit this section only if the default port (9004) is not available.

For detailed descriptions of the options in this section, see [server\\_remotecomms](#) on page 284.

For information about configuring the Remote Operator feature (remote slots), as opposed to remote modules, see Chapter 8: [Remote Operator Card Sets](#).

### Hardserver settings

You configure the hardserver's settings in the **server\_settings** section of the configuration file.

This section defines how connections and hardserver logging are handled. These settings can be changed while the hardserver is running.

For detailed descriptions of the options in this section, see [server\\_settings](#) on page 282.

## Module settings

You configure the module's settings in the **module\_settings** section of the configuration file.

This section defines the settings for the module that can be changed while the hardserver is running.

For detailed descriptions of the options in this section, see [module\\_settings](#) on page 284.

## Hardserver start-up settings

You configure the hardserver's start-up settings in the **server\_startup** section of the configuration file.

This section defines the sockets and ports used by the hardserver. You need to change this section only if the default ports for privileged or unprivileged connections (9000 and 9001) are not available.

For detailed descriptions of the options in this section, see [server\\_startup](#) on page 284.

## SEE machines

You configure the hardserver to load SEE machines on start-up in the **load\_seemachine** section of the configuration file. The SEE Activation feature must be enabled on the module, as described in [Enabling optional features on the module](#) on page 60.

This section defines the SEE machines and optional user data to be loaded, as well any other applications to be run in order to initialize the machine after it is loaded.

For detailed descriptions of the options in this section, see [load\\_seemachine](#) on page 285.

For information about SEE machines, see [CodeSafe applications](#) on page 192.

## Remote slots

You configure remote slots in the **slot\_imports** and **slot\_exports** sections of the configuration file. These sections define the slots that are imported to or exported from the module.

For detailed descriptions of the options in these sections, see [slot\\_imports](#) on page 286 and [slot\\_exports](#) on page 287.

The Remote Operator feature must be enabled on the module, as described in [Enabling optional features on the module](#) on page 60.

## Remote file system

Each client's remote file system is defined separately in the **remote\_file\_system** section of the configuration file with a list of modules that are allowed to access the file system on the given client. For information about setting up client cooperation, see [Setting up client cooperation](#) on page 51.



The **remote\_file\_system** section is updated automatically when the **rfs-setup** utility is run. Do not edit the **remote\_file\_system** section manually.

As a reference, for detailed descriptions of the options in this section, see [remote\\_file\\_system](#) on page 289.

## Enabling optional features on the module

nShield modules support a range of optional features. Optional features must be enabled with a certificate that you order from Thales. You can order the features when you purchase a module, or you can obtain the certificate at a later date and use the Feature Enable Tool to enable the features.

For more information about:

- ordering optional features, see [Ordering additional features](#) on page 65
- feature-enabling procedures, see [Enabling features](#) on page 66.

The firmware checks to confirm whether any features that it attempts to use are enabled. It normally does this when it authorizes the commands or command options that relate to a specific feature.

Most features are *static*; that is, they are enabled by means of a switch in the **EEPROM** of the module. A static feature remains enabled when the module is reinitialized.

Some optional features are *dynamic*; that is, they are enabled by means of a software switch in the volatile memory of the module. A dynamic feature must be enabled again if the module is reinitialized.

After you have enabled features on a module, you must clear the module to make them available. Clear the module by running the command **nopclearfail --clear --all** or by pressing the module's Clear switch).

**Note** If you are enabling the Remote Operator feature, you must enable it on the module that is to be used as the unattended module. For information about Remote Operator, see Chapter 8: [Remote Operator Card Sets](#).

## Available optional features

### Elliptic Curve

Cryptography based on elliptic curves relies on the mathematics of random elliptic curve elements. It offers better performance for an equivalent key length than either RSA or Diffie-Hellman public key systems. Using RSA or Diffie-Hellman to protect 128-bit AES keys requires a key of at least 3072 bits. The equivalent key size for elliptic curves is only 256 bits. Using a smaller key reduces storage and transmission requirements.

Elliptic curve cryptography is endorsed by the US National Security Agency and NIST (the National Institute of Standards and Technology), and by standardization bodies including ANSI, IEEE and ISO.

Thales modules incorporate hardware that supports elliptic curve operations for ECDH (Elliptic curve Diffie-Hellman) and ECDSA (Elliptic Curve Digital Signature Algorithm) keys.

### Elliptic Curve activation

All nShield HSMs require specific activation to utilize the elliptic curve features. Thales uses an activator smart card to enable this feature. Refer to [Enabling features with a smart card](#) on page 66 for instructions on how to enable the EC feature. Additionally it is possible to activate the elliptic curve feature without a physical smart card. In this case the certificate details can be provided by email and entered locally. Refer to [Enabling features without a smart card](#) on page 67. Contact Sales if you require an EC activation.

Thales modules with elliptic curve activation support *MQV* (*Menezes-Qu-Vanstone*) modes.

## Elliptic Curve support on the nShield product line

The following table details the range of nShield HSMs and the level of elliptic curve support that they offer.

| HSM module type                                                              | Elliptic Curve support    |                               | Elliptic Curve offload acceleration <sup>3</sup>    |                               |
|------------------------------------------------------------------------------|---------------------------|-------------------------------|-----------------------------------------------------|-------------------------------|
|                                                                              | Named curves <sup>2</sup> | Custom curves <sup>1, 5</sup> | Named curves                                        | Custom curves <sup>1, 5</sup> |
| nShield PCI 500.                                                             | Yes.                      | Yes.                          | Yes. <sup>2</sup> .                                 | Yes.                          |
| nShield PCI 2000, 4000; nShield Edge <sup>2</sup> .                          | Yes.                      | Yes.                          | No.                                                 | No.                           |
| nShield PCIe 500 and 6000; nShield Connect 500, 1500 and 6000 <sup>2</sup> . | Yes.                      | Yes.                          | No.                                                 | No.                           |
| nShield 6000+ PCIe, nShield Connect 6000+ <sup>2</sup> .                     | Yes.                      | Yes.                          | Yes, Prime curves are accelerated <sup>2, 4</sup> . | Yes.                          |

<sup>1</sup>Accessed via nCore and PKCS #11 APIs.

<sup>2</sup>Both Prime and Binary named curves are supported. Refer to [Named Curves](#) on page 63, below, which lists the most commonly supported elliptic curves.

<sup>3</sup>Offload acceleration refers to offloading the elliptic curve operation from the main CPU for dedicated EC hardware acceleration.

<sup>4</sup>Binary curves are supported, but are not hardware offload accelerated.

<sup>5</sup>Includes support for Brainpool curves. See [Brainpool curves](#) on page 63, below, for a list of Brainpool curves.

nShield software / API support required to use elliptic curve functions

|                                | Security World Software for nShield                                    | CipherTools                                                            | CodeSafe                                                               |
|--------------------------------|------------------------------------------------------------------------|------------------------------------------------------------------------|------------------------------------------------------------------------|
| Elliptic curve supported / API | Microsoft CNG, PKCS#11, Java Cryptographic Engine [JCE] <sup>1</sup> . | Microsoft CNG, PKCS#11, Java Cryptographic Engine [JCE] <sup>1</sup> . | Microsoft CNG, PKCS#11, Java Cryptographic Engine [JCE] <sup>1</sup> . |

<sup>1</sup>To enable Java elliptic curve functionality, Thales recommends using the PKCS #11 library instead of the JCE (Java Cryptographic Extension) programming interface.

To demonstrate the accelerated performance of elliptic signing and verify operations, run the **perfcheck** utility. See [perfcheck: performance measurement checking tool](#) on page 271.

## Named Curves

This table lists the supported named curves that are pre-coded in Thales module firmware.

| Curve      | Offload acceleration: PCI 500 | Offload acceleration: nShield 6000+ and nShield Connect 6000+ |
|------------|-------------------------------|---------------------------------------------------------------|
| NISTP192   | Yes                           | Yes                                                           |
| NISTP224   | Yes                           | Yes                                                           |
| NISTP256   | Yes                           | Yes                                                           |
| NISTP384   | Yes                           | Yes                                                           |
| NISTP521   | —                             | Yes                                                           |
| NISTB163   | Yes                           | —                                                             |
| NISTB233   | Yes                           | —                                                             |
| NISTB283   | Yes                           | —                                                             |
| NISTB409   | Yes                           | —                                                             |
| NISTB571   | —                             | —                                                             |
| NISTK163   | Yes                           | —                                                             |
| NISTK233   | Yes                           | —                                                             |
| NISTK283   | Yes                           | —                                                             |
| NISTK409   | Yes                           | —                                                             |
| NISTK571   | —                             | —                                                             |
| ANSIB163v1 | Yes                           | —                                                             |
| ANSIB191v1 | Yes                           | —                                                             |
| SECP160r1  | Yes                           | Yes                                                           |

## Custom curves

Thales modules also allow the entry of custom elliptic curves which are not pre-coded in firmware. If the curve is Prime, it may benefit from hardware acceleration if supported by the nShield HSM (see [nShield software / API support required to use elliptic curve functions](#) on page 62, above).

Custom curves are supported by nCore and PKCS #11 APIs.

### Brainpool curves

Brainpool curves are supported as a custom curve type. The following Brainpool curves are supported.

| Curve size in bits | Curve-ID        | Curve-ID (twisted representation) |
|--------------------|-----------------|-----------------------------------|
| 160                | brainpoolP160r1 | brainpoolP160t1                   |
| 192                | brainpoolP192r1 | brainpoolP192t1                   |

| Curve size in bits | Curve-ID        | Curve-ID (twisted representation) |
|--------------------|-----------------|-----------------------------------|
| 224                | brainpoolP224r1 | brainpoolP224t1                   |
| 256                | brainpoolP256r1 | brainpoolP256t1                   |
| 320                | brainpoolP320r1 | brainpoolP320t1                   |
| 384                | brainpoolP384r1 | brainpoolP384t1                   |
| 512                | brainpoolP512r1 | brainpoolP512t1                   |

Further information on using elliptic curves

For more information on how to use elliptic curves, see the following sections:

- PKCS #11:
  - Mechanisms supported by PKCS #11: [Mechanisms](#) on page 168
- CNG:
  - Supported algorithms for CNG:
  - Key exchange for CNG:
- Symmetric and asymmetric algorithms: [Cryptographic algorithms](#) on page 291
- Using **generatekey** options and parameters to generate ECDH and ECDSA keys: [Key generation options and parameters](#) on page 295

Note To enable Java elliptic curve functionality, use the PKCS #11 library instead of the JCE (Java Cryptographic Extension) programming interface.

## Secure Execution Engine (SEE)

The SEE is a unique secure execution environment. The SEE features available to you are:

|                             |                                                                                                                                                                                                                                                                                                                                   |
|-----------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| SEE Activation (EU+10)      | This SEE feature is provided with the CodeSafe developer product to enable you to develop and run SEE applications. The CodeSafe developer product is only available to customers in the Community General Export Area (CGEA, also known as EU+10). Contact Thales to find out whether your country is currently within the CGEA. |
| SEE Activation (Restricted) | This SEE feature is provided with specific products that include an SEE application. This feature enables you to run your specific SEE application and is available to customers in any part of the world.                                                                                                                        |

For more information about the SEE, see the *CodeSafe Developer Guide*.



## Remote Operator support

Many Thales customers keep critical servers in a physically secure and remote location. The Security World infrastructure, however, often requires the physical presence of an operator to perform tasks such as inserting cards. Remote Operator enables these customers to remotely manage servers running Security World Software using a secure Thales communications protocol over IP networks.

The Remote Operator feature must be enabled on the module installed in the remote server. Remote Operator cannot be enabled remotely on an unattended module.

For more information about using Remote Operator, see Chapter 8: [Remote Operator Card Sets](#).

## ISO smart card Support (ISS)

ISS, also called Foreign Token Open (FTO) allows data to be read to and written from ISO 7816 compliant smart cards in a manner prescribed by ISO7816-4. ISS allows you to develop and deploy a security system that can make full use of ISO 7816 compliant smart cards from any manufacturer.

## Korean algorithms

This feature enables the following mechanisms:

- Korean Certificate-based Digital Signature Algorithm (KCDSA), which is a signature mechanism.

KCDSA is used extensively in Korea as part of compliance with local regulations specified by the Korean government. For more information about the KCDSA, see the *nCore API Documentation*.

- SEED, which is a block cipher.
- ARIA, which is a block cipher.
- HAS160, which is a hash function.

## Ordering additional features

When you have decided that you require a new feature, you can order it for your from Sales. Before you call Sales, you collect information about your as follows:

- If possible, make a note of the serial number. This can be found on the circuit board of nShield modules.

- Run the **enquiry** command and note down the Electronic Serial Number of the module.

You must provide the ESN number to order a new feature.

If you prefer, you can include this information in an e-mail to Sales. You can use the Feature Enable Tool to save the ESN details to a file. For more information about using the Feature Enable Tool, see [Enabling features](#) on page 66.

When your order has been processed, you receive a Feature Enabling Certificate in one of the following ways:

- Thales e-mails you the Feature Enabling Certificate.
- Thales sends you a smart card that contains the Feature Enabling Certificate.

The Feature Enabling Certificate contains the information that you need to enable the features you have ordered.

For more information, including pricing of features, telephone or e-mail your nearest Sales representative using the contact details from this guide, or the Thales web site (<http://iss.thalesgroup.com/Resources.aspx>).

## Enabling features

### Viewing enabled features

The **Feature Enable Tool** can be used to view the status of modules connected to the host or to confirm that a feature has been successfully enabled on all modules connected to the host. To view the status of features, run the tool without a smart card.

### Enabling features with a smart card

When it is launched, the **Feature Enable Tool** automatically scans the smart card readers of all modules attached to a host computer for any Feature Enabling smart cards present in the smart card readers, including imported slots.

To enable a new feature with a Feature Enabling smart card from Thales:

- 1 Insert the Feature Enabling card from Thales into a slot available to the module to be updated.
- 2 Run the **fet** command-line utility to start the **Feature Enable Tool**.

A message is displayed if the features are enabled successfully. If you do not see this message confirming a successful upgrade, see [Enabling features without a smart card](#) on page 67.

## Enabling features without a smart card

The **Feature Enable Tool** can also obtain the Feature Enabling Certificate information supplied by Thales from a file or from the keyboard.

When you run the **Feature Enable Tool** without a Feature Enabling smart card in any module slot, a message similar to the following is displayed. There is a line for the features on each module, and a list of options. In this example, only one module (**ESN E51C-D135-83F8**) is attached to the host.

---

```

 Feature Enable Tool
 =====
 payShield Activation
 | ISO Smart Card Support
 | | Remote Operator
 | | | Korean Algorithms
 | | | | SEE Activation (EU+10)
 | | | | | SEE Activation (Restricted)
 | | | | | | CodeSafe SSL
 | | | | | | | Elliptic Curve algorithms
 | | | | | | | | Elliptic Curve MQV
 | | | | | | | | | Accelerated ECC
Mod Electronic | | | | | | | | |
No. Serial Number
1 E51C-D135-83F8 -- YES YES NO YES YES YES YES YES NO NO
0. Exit Feature Enable Tool.
1. Read FEM certificate(s) from a smart card or cards.
2. Read FEM certificate from a file.
3. Read FEM certificate from keyboard.
4. Write table to file.
Enter option :
```

---

If you select option 1, you are prompted to insert a Feature Enabling smart card into a module slot and then press **Enter**. The **Feature Enable Tool** then behaves in exactly the same way as if it were run with the Feature Enabling cards already in the slots.

If you select option 2, you are prompted for the location and file name of the Feature Enabling certificate. When you supply these, you are prompted for the module number and the feature.

If you select option 3, you are prompted for the module number and the feature, and are then asked to enter the certificate one line at a time, followed by a period (".") on a line by itself. You can also cut and paste a Feature Enabling certificate from an e-mail.

If you select option 4, the table of enabled features is written to a file. You can include this file when you request new features from Thales. See [Ordering additional features](#) on page 65.

## Using multiple modules

The hardserver can communicate with multiple modules connected to the host. By default, the server accepts requests from applications and submits each request to the first available module. The server can share load across buses, which includes the ability to share load across more than one module.

If your application is multi-threaded, you can add additional modules and expect performance to increase proportionally until you reach the point where cryptography no longer forms a bottleneck in the system.

## Identifying modules

Modules are identified in two ways:

- by serial number
- by **ModuleID**.

You can obtain the **ModuleIDs** and serial numbers of all your modules by running the **enquiry** command-line utility.

### Serial Number

The serial number is a unique 12-digit number that is permanently encoded into each module. Quote this number in any correspondence with Support.

### ModuleID

The **ModuleID** is an integer assigned to the module by the server when it starts. The first module it finds is given a **ModuleID** of 1, the next is given a **ModuleID** of 2, and this pattern of assigning **ModuleID** numbers continues for additional modules.

The order in which buses are searched and the order of modules on a bus depends on the exact configuration of the host. If you add or remove a module, this can change the allocation of **ModuleIDs** to all the modules on your system.

You can use the **enquiry** command-line utility to identify the PCI bus and slot number associated with a module.

All commands sent to nShield modules require a **ModuleID**. Many Security World Software commands, including all acceleration-only commands, can be called with a **ModuleID** of 0. Such a call causes the hardserver to send the command to the first available module. If you purchased a developer kit, you can refer to the developer documentation for information about the commands that are available on nShield modules.

In general, the hardserver determines which modules can perform a given command. If no module contains all the objects that are referred to in a given command, the server returns an error status.

However, some key-management operations must be performed together on the same module. In such cases, your application must specify the **ModuleID**.

To be able to share OCSs and keys between modules, the modules must be in the same Security World.

## Adding a module

If you have a module installed, you can add further modules without reinstalling the server software.

However, we recommend that you always upgrade to the latest server software and upgrade the firmware in existing modules to the latest firmware.

**Note** Before you install new hardware, check the release notes on the DVD-ROM supplied with your new module for information about specific compatibility issues, new features, and bug fixes.

- 1 Install the module hardware. Refer to the *Quick Start Guide* for information on installing Thales hardware.
- 2 Run the script: `/opt/nfast/sbin/install`
- 3 Add the module to the Security World. Refer to [Adding or restoring a module to the Security World](#) on page 94.

## Module fail-over

The Security World Software supports fail-over: if a module fails, its processing can be transferred automatically to another module provided the necessary keys have been loaded. Depending on the mode of failure, however, the underlying bus and operating system may not be able to recover and continue operating with the remaining devices.

To maximize uptime, we recommend that you fit any additional nShield modules for failover on a bus that is physically separate from that of the primary modules.

## Stopping and restarting the hardserver

If necessary, you can stop the hardserver on the client by running the following command:

---

```
/opt/nfast/sbin/init.d-ncipher stop
```

---

Similarly, you can restart the hardserver on the client by running the following command:

---

```
/opt/nfast/sbin/init.d-ncipher start
```

---



## Chapter 5: Creating and managing a Security World

This chapter describes how to create and manage a Security World. You must create a Security World before using the nShield module to manage keys.

You normally create a Security World after installing and configuring the module and its support software. For more information, see:

- Chapter 3: [Support software installation](#)
- Chapter 4: [Software and module configuration](#).

You create a Security World with a single nShield module. If you have more than one module, select one module with which to create the Security World, then add additional modules to the Security World after its creation. For more information, see [Adding or restoring a module to the Security World](#) on page 94.

**Note** To use the module to protect a different set of keys, you can replace an existing Security World with a new Security World.

For more information about the type of user that is required for different operations, see [About user privileges](#) on page 51.

### Creating a Security World

Before you start to create a Security World:

- The modules that you wish to add to the Security World must be in pre-initialization mode. For more information about module modes, see Appendix I: [Checking and changing module mode](#).



On some internal modules, you must access the initialization link or switch on the module. With such modules, always shut down your computer and turn off the power before opening the case. Replace the case before reconnecting the power.

- You must be logged in to the host computer as **root** or as a user in the group **nfast**. For more information, see [Hardserver start-up settings](#) on page 59 and [server\\_startup](#) on page 284.

- If you have installed the Security World Software in a directory other than **/opt/nfast/**, you must have created a symbolic link from **/opt/nfast/** to the directory in which the software is actually installed.
- Before configuring the Security World, you should know:
  - the security policy for the module
  - the number and quorum of Administrator Cards and Operator Cards to be used.

To help you decide on the Security World you require, see [Security World options](#) on page 74.

- You must have enough smart cards to form the Security World's card sets.

When you have finished creating a Security World, you must restart the module in the operational state.

The process of creating a Security World:

- erases the module
- creates a new module key for this Security World
- creates a new ACS to protect this module key
- stores the Security World information on the computer's hard disk. The information is encrypted using the secrets stored on the ACS.



Any Operator Cards created in a previous Security World cannot be used in a new Security World. If you are replacing a Security World, you must erase all the Operator Cards created in the previous Security World before you create the new Security World [see [Erasing cards and softcards](#) on page 128].

You can use the following tools to create a Security World:

- the **new-world** command-line utility, as described in [Creating a Security World by using new-world](#) on page 79

Note You can also use **new-world** to add a module to an existing Security World.

- KeySafe, as described in [Creating a Security World with KeySafe](#) on page 85

You can use KeySafe and the **new-world** utility to create a Security World for which you can specify different **K**values for the ACS.



## Security World files

The Security World infrastructure stores encrypted key material and related data in files on the host. For multiple hosts to use the same Security World, the system administrator must ensure that these files are copied to all the hosts and updated when required.

### Location of Security World files

Security World files are created or updated in the directory specified by the environment variable **NFAST\_KMLOCAL** on the host. By default, this is **/opt/nfast/kmdata/local**.

**Note** By default, the Key Management Data directory, and sub-directories, inherit permissions from the user that creates them. Installation of the Security World Software must be performed by a user with Administrator rights that allow read and write operations, and the starting and stopping of applications.

### Files and operations

Security World operations create or modify files in the **/opt/nfast/kmdata/local** directory as follows:

| Operation               | creates/modifies    | the file(s) ...                                                           |
|-------------------------|---------------------|---------------------------------------------------------------------------|
| Create a Security World | creates             | <b>world</b><br>[for each module in the Security World] <b>module_ESN</b> |
| Load a Security World   | creates or modifies | [for each module in the Security World] <b>module_ESN</b>                 |
| Replace an ACS          | modifies            | <b>world</b>                                                              |
| Create an OCS           | creates             | <b>card_HASH</b><br><b>cards_HASH_NUMBER</b>                              |
| Create a softcard       | creates             | <b>softcard_HASH</b>                                                      |
| Generate a key          | creates             | <b>key_APPNAME__IDENT</b>                                                 |
| Recover a key           | modifies            | <b>key_APPNAME</b> [for each key that has been recovered]                 |

In this table:

- **ESN** is the electronic serial number of the module on which the Security World is created
- **IDENT** is the identifier given to the card set or key when it is created
- **NUMBER** is the number of the card in the card set
- **APPNAME** is the name of the application by which the key was created.

The **IDENT** of a card set is a 40-character string that represents the hash of the card set's logical token. The **IDENT** of a key is either user supplied or a hash of the key's logical token, depending on the application that created the key.

## Required files

The following files must be present and up to date in the **/opt/nfast/kmdata/local** directory, or the directory specified by the **NFAST\_KMLOCAL** environment variable, for a host to use a Security World:

- **world**
- a **module\_ESN** file for each module that this host uses
- a **cards\_IDENT** file for each card set that is to be loaded from this host
- a **card\_IDENT\_NUMBER** file for each card in each card set that is to be loaded from this host
- a **key\_APPNAME\_IDENT** file for each key that is to be loaded from this host.

These files are not updated automatically. You must ensure that they are synchronized whenever the Security World is updated on the module.

## Security World options

Decide what kind of Security World you need before you create it. Depending on the kind of Security World you need, you can choose different options at the time of creation. For convenience, Security World options can be divided into the following groups:

- basic options, which must be configured for all Security Worlds
- recovery and replacement options, which must be configured if the Security World, keys, or pass phrases are to be recoverable or replaceable
- SEE options, which only need be configured if you are using the nCipher Secure Execution Engine (SEE)
- options relating to the replacement of an existing Security World with a new Security World.

Security World options are highly configurable at the time of creation but, so that they will remain secure, not afterwards. For this reason, we recommend that you familiarize yourself with Security World options, especially those required by your particular situation, before you begin to create a Security World.

## Security World basic options

When you create a Security World, you must always configure the basic options described in this section.

### Cipher suite

You must decide whether to use a cipher suite that uses Triple DES, AES (standard), or AES (SP800-131 compliant) Security World keys. The Security World keys are generated during Security World creation, and protect the application keys and OCSs in the created Security World.

**Note** Due to the additional primality checking required by SP800-131, Security World generation and key generation operations will take longer in SP800-131 compliant Security Worlds.

To create a Triple DES Security World, you must use the **new-world** command-line utility.

### K and N

You must decide the total number of cards ( $N$ ) in a Security World's ACS and must have that many blank cards available before you start to create the Security World. You must also decide how many cards from the ACS must be present ( $K$ ) when performing administrative functions on the Security World.

**Note** We recommend that you do not create ACSs for which  $K$  is equal to  $N$ , because you cannot replace such an ACS if even 1 card is lost or damaged.

In many cases, it is desirable to make  $K$  greater than half the value of  $N$  (for example, if  $N$  is 7, to make  $K$  4 or more). Such a policy makes it harder for a potential attacker to obtain enough cards to access the Security World. Choose values of  $K$  and  $N$  that are appropriate to your situation.

The total number of cards used in the ACS must be a value in the range 1 – 64.

### FIPS 140-2 level 3 compliance

By default, Security Worlds are created to comply with the roles and services, key management, and self-test sections of the FIPS 140-2 standard at level 2. However, you can choose to enable compliance with the FIPS 140-2 standard at level 3.

**Note** This option provides compliance with the roles and services of the FIPS 140-2 level 3 standard. It is included for those customers who have a regulatory requirement for compliance.

If you enable compliance with FIPS 140-2 level 3 roles and services, authorization is required for the following actions:

- generating a new OCS
- generating or importing a key, including session keys
- erasing or formatting smart cards (although you can obtain authorization from a card you are about to erase).

In addition, you cannot import or export private or symmetric keys in plain text.

### UseStrongPrimes Security World setting

When creating a Security World, the default setting for UseStrongPrimes depends on the FIPS level:

- *FIPS 140-2 level 3*: UseStrongPrimes is *on*, meaning that the Security World always generates RSA keys in a manner compliant with FIPS 186-3.
- *FIPS 140-2 level 2*: UseStrongPrimes is *off*, meaning that the Security World leaves the choice of RSA key generation algorithm to individual clients.

Enabling UseStrongPrimes increases the RSA key generation time by approximately 10 times. If you want to use a different UseStrongPrimes setting from its default setting, you must use the **new-world** command-line utility to create the Security World.

The **nfkminfo** utility shows the status of the Security World. Running the **enquiry** utility for a module shows the status of the world in relation to that module.

### Remote Operator

To use a module without needing physical access to present Operator Cards, you must enable the Remote Operator feature on the module. For more information, see [Enabling optional features on the module](#) on page 60.

By default, modules are initialized into Security Worlds with remote card set reading enabled. If you add a module for which remote card reading is disabled to a Security World for which remote card reading is enabled, the module remains disabled.

## OCS and softcard replacement

By default, Security Worlds are created with the ability to replace one OCS or softcard with another. This feature enables you to transfer keys from the protection of the old OCS or softcard to a new OCS or softcard.

**Note** You can replace an OCS with another OCS, or a softcard with another softcard, but you cannot replace an OCS with a softcard or a softcard with an OCS. Likewise, you can transfer keys from an OCS to another OCS, or from a softcard to another softcard, but you cannot transfer keys from an OCS to a softcard or from a softcard to an OCS.

You can choose to disable OCS and softcard replacement for a Security World when you create it. However, in a Security World without this feature, you can never replace lost or damaged OCSs; therefore, you could never recover the keys protected by lost or damaged OCSs, even if the keys themselves were generated as recoverable (which is the default for key generation).



OCS and softcard replacement cannot be enabled after Security World creation without reinitializing the Security World and discarding all the existing keys within it.

For an overview of Security World robustness and OCS or softcard replacement, see [Replacing an Operator Card Set or recovering keys to softcards](#) on page 31. For details about performing OCS and softcard replacement operations, see [Replacing Operator Card Sets](#) on page 140 and [Replacing the Administrator Card Set](#) on page 151.

## Pass phrase replacement

By default, Security Worlds are created so that you cannot replace the pass phrase of a card or softcard without knowing the existing pass phrase.

However, you can choose to enable pass phrase replacement at the time you create a Security World. This option makes it possible to replace a the pass phrase of a card or softcard even if you do not know the existing pass phrase. Performing such an operation requires authorization from the Security World's ACS.

For details about performing pass phrase replacement operations, see [Changing unknown or lost pass phrases](#) on page 138.

## Nonvolatile memory (NVRAM) options

Enabling nonvolatile memory (NVRAM) options allows keys to be stored in the module's NVRAM instead of in the Key Management Data directory of the host computer. Files stored in the module's non-volatile memory have Access Control Lists (ACLs) that control who can access

the file and what changes can be made to the file. NVRAM options are relevant only if your module's firmware supports them, and you can store keys in your module's NVRAM only if there is sufficient space.

**Note** When the amount of information to be stored in the NVRAM exceeds the available capacity, you can instead store this data in a blob encrypted with a much smaller key that is itself then stored in the NVRAM. This functionality allows the amount of secure storage to be limited only by the capacity of the host computer.

## Security World SEE options

You must configure **SEE options** if you are using the nCipher Secure Execution Engine (SEE). If you do not have SEE installed, the SEE options are irrelevant.

### SEE debugging

SEE debugging is disabled by default, but you can choose whether to enable it for all users or whether to make it available only through use of an ACS. In many circumstances, it is useful to enable SEE debugging for all users in a development Security World but to disable SEE debugging in a production Security World. Choose the SEE debugging options that best suit your situation.

### Real-time clock (RTC) options

Real-time clock (RTC) options are relevant only if you have purchased and installed the CodeSafe Developer kit. If so, by default, Security Worlds are created with access to RTC operations enabled. However, you can choose to control access to RTC operations by means of an ACS.

## Security World replacement options

Options relating to Security World replacement are relevant only if you are replacing a Security World.

If you replace an existing Security World, its **kmdata/local** directory is not overwritten but renamed **kmdata/local\_N** (where *N* is an integer assigned depending on how many Security Worlds have been previously saved during overwrites). A new Key Management Data directory is created for the new Security World. If you do not wish to retain the **kmdata/local\_N** directory from the old Security World, you must delete it manually.

## Creating a Security World by using new-world

Follow the directions in this section to create a Security World from the command line with the **new-world** utility.

### Running the new-world command-line utility

Open a command window and type the command:

---


```
new-world --initialize [--factory] [--no-remoteshare-cert] [--strict-fips-140-2-level-3]
[--no-strict-rsa-keygen|--strict-rsa-keygen] [--no-recovery] --cipher-suite=CIPHER-SUITE [--nso-
timeout=TIMEOUT] [--module MODULE] --acs-quorum=K/N [--reduced-features] FEATURES
```

---

In this command:

| Option                                                      | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|-------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>--initialize</b>                                         | <p>This option tells <b>new-world</b> to create a new Security World, replacing any existing <b>kmdata</b> directory.</p> <p><b>Note</b> Replacing an existing Security World in this way does not delete the Security World's host data and recovery and replacement data, but renames the existing <b>kmdata/local</b> directory in which these reside as <b>kmdata/local_</b><i>N</i> (where <i>N</i> is an integer assigned depending on how many Security Worlds have been previously saved during overwrites).</p>                                                                                                                                         |
| <b>--factory</b>                                            | This option tells <b>new-world</b> to erase a module, restoring it to factory state.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| <b>--no-remoteshare-cert</b>                                | This option tells <b>new-world</b> not to make the module a target for remote shares.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| <b>--strict-fips-140-2-level-3</b>                          | <p>This options tells <b>new-world</b> to create a Security World that conforms to the FIPS 140-2 requirements for roles and services at level 3. If you do not specify this flag, <b>new-world</b> creates a Security World that complies with FIPS 140-2 requirements for level 2.</p> <p><b>Note</b> This option provides compliance with the roles and services of the FIPS 140-2 level 3 standard. It is included for those customers who have a regulatory requirement for compliance.</p>                                                                                                                                                                 |
| <b>--no-strict-rsa-keygen</b><br><b>--strict-rsa-keygen</b> | <p>If you are using the <b>--strict-fips-140-2-level-3</b> flag to create a FIPS 140-2 level 3 Security World, you can use the <b>--no-strict-rsa-keygen</b> flag to <i>disable</i> UseStrongPrimes. Otherwise it is enabled by default.</p> <p>If you are creating a FIPS 140-2 level 2 Security World, you can use the <b>--strict-rsa-keygen</b> flag to <i>enable</i> UseStrongPrimes. Otherwise, it is disabled by default.</p> <p>If UseStrongPrimes is on, the Security World always generates RSA keys in a manner compliant with FIPS 186-3. Otherwise, the Security World leaves the choice of RSA key generation algorithm to individual clients.</p> |



| Option                              | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|-------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>--no-recovery</b>                | <p>This option tells <b>new-world</b> to disable the ability to recovery or replace OCSs and softcard (which is otherwise enabled by default). This is equivalent to setting <b>!r</b>, where the <b>!</b> operator instructs the system to turn off the specified feature <b>[r]</b>.</p> <p>By default, <b>new-world</b> creates key recovery and replacement data that is protected by the cryptographic keys on the ACS. This option does not give Thales or any other third party access to your keys. Keys can only be recovered if authorization from the ACS is available. We recommend that you leave OCS and softcard recovery and replacement functionality enabled.</p> <div style="display: flex; align-items: flex-start;">  <div style="margin-left: 10px;"> <p>We recommend that you do not disable the recovery and replacement option.</p> <p>If you set the <b>--no-recovery</b> option, you can never replace lost or damaged OCSs generated for that Security World. Therefore, you could never recover any keys protected by lost or damaged OCSs, even if the keys themselves were generated as recoverable (which is the default for key generation).</p> <p>OCS and softcard replacement cannot be enabled after Security World creation without reinitializing the Security World and discarding all the existing keys within it.</p> </div> </div> |
| <b>--cipher-suite= CIPHER-SUITE</b> | <p>This option specifies the Cipher suite and type of key that is used to protect the new Security World. <i>CIPHER-SUITE</i> can be one of the following:</p> <ul style="list-style-type: none"> <li>• <b>DLf1024s160mDES3</b>, which uses a Triple DES key.</li> <li>• <b>DLf1024s160mRijndael</b>, which uses an AES key.</li> <li>• <b>DLf3072s256mRijndael</b>, which uses an AES key to create a Security World compliant with SP800-131.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <b>--nso-timeout= TIMEOUT</b>       | <p>This option allows you to specify the time-out (<b>TIMEOUT</b>) for new Security Worlds. By default, an integer given for <b>TIMEOUT</b> is interpreted in seconds, but you can supply values for <b>TIMEOUT</b> in the form <b>Ns</b>, <b>Mh</b>, or <b>Nd</b> where <b>N</b> is an integer and <b>s</b> specifies second, <b>h</b> specifies hours, and <b>d</b> specifies days.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| <b>--module= MODULE</b>             | <p>This option specifies the module to use (by its <b>ModuleID</b>). If you have multiple modules, <b>new-world</b> initializes them all together.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |

| Option                                     | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|--------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>--acs-quorum=<math>K/N</math></code> | <p>In this option, <math>K</math> specifies the minimum number of smart cards needed from the ACS to authorize a feature. You can specify lower <math>K</math> values for a particular feature. All the <math>K</math> values must be less than or equal to the total number of cards in the set. If a value for <math>K</math> is not specified, <b>new-world</b> creates an ACS that requires a single card for authorization.</p> <p><b>Note</b> Some applications do not have mechanisms for requesting that cards be inserted. Therefore any OCSs that you create for use with these applications must have <math>K=1</math>.</p> <p><math>N</math> specifies the total number of smart cards to be used in the ACS. This must be a value in the range 1 – 64. If a value for this option is not specified, <b>new-world</b> creates an ACS that contains a single card.</p> <p><b>Note</b> We recommend that you do not create an ACS for which the required number of cards is equal to the total number of cards because you will not be able to replace the ACS if even a single card is lost or damaged.</p> <p>This option only takes effect if you are creating a new Security World.</p> |
| <code>--reduced-features</code>            | <p>This option instructs <b>new-world</b> to use a reduced default feature set when creating a Security World. A Security World created with the <code>--reduced-features</code> option has no pass phrase recovery; no NVRAM, RTC, or FTO; and no NSO delegate keys. However, such a reduced-features Security World can perform many operations faster than more fully featured Security Worlds.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |

**Note** The `--km-type` option, which was used in previous releases to select the Security World module key type, is now deprecated in favor of the `--cipher-suite` option. If you use both options simultaneously, the **new-world** utility displays an error and exits.

## new-world command-line utility features

Features for the Security World can be specified on the command line.

Security world features are selected by 'feature expressions'. A feature expression is a comma-separated list of 'feature terms'. Each term consists of a feature name, optionally preceded by either a double dash --, an exclamation point !, or **no-** to turn off the feature, and optionally followed by an equals sign = and the quorum of cards from the ACS required to use the feature. The default quorum is taken from the **K** argument of the **--acs-quorum** option.

**Note** The ! character is interpreted by some shells as history expansion and must be escaped with a backslash, \!. The dash may be interpreted as being the start of an command-line option unless you have used the **-f** option or specified a module without including the **-m** flag.

**Note** If you set the **!fto** flag, that is, turn off FTO, you will not be able to use smart cards to import keys.

**Note** To use extended debugging for the module, you must set the **dseeall** flag.

The following feature names are available:

| Feature name   | Description                                                                                                                                                                   |
|----------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>m</b>       | This feature makes it possible to add new modules into the Security World. This feature cannot be disabled.                                                                   |
| <b>r</b>       | This feature enables OCS and softcard replacement; see <a href="#">OCS and softcard replacement</a> on page 77 and <a href="#">Replacing Operator Card Sets</a> on page 140.  |
| <b>p</b>       | This feature enables pass phrase replacement; see <a href="#">Pass phrase replacement</a> on page 77 and <a href="#">Changing card and softcard pass phrases</a> on page 135. |
| <b>nv</b>      | This feature specifies that ACS authorization is needed to enable nonvolatile memory (NVRAM) allocation.                                                                      |
| <b>rtc</b>     | This feature specifies that ACS authorization is needed to set the real-time clock (RTC); (see <a href="#">Real-time clock (RTC) options</a> on page 78).                     |
| <b>dsee</b>    | This feature specifies that that ACS authorization is needed to enable SEE World debugging.                                                                                   |
| <b>dseeall</b> | This feature enables SEE World debugging for all users.                                                                                                                       |
| <b>fto</b>     | This feature specifies that ACS authorization is needed to enable foreign token operations [FTO].                                                                             |

The following features remain available for use on presentation of the standard ACS quorum, even if turned off using the ! operator:

- **p**
- **nvram**
- **rtc**

- **fto**

Setting the quorum of one these features to **0** has the same effect as turning it off using the **!** operator.

The pass phrase replacement (**p**) and **dseeall** features are turned off by default; the other options are turned on by default.

**Note** The nonvolatile memory and SEE world debugging options are relevant only if you are using the Secure Execution Engine. If you have bought the CodeSafe Developer Kit, refer to the *CodeSafe Developer Guide* for more information.

**Note** To use extended debugging for the module, you must set the **dseeall** flag.



The **dseeall** option is designed for testing purposes only. Do not enable this feature on production Security Worlds as it may enable SEE applications to leak security information.

For example, the following features:

---

```
m=1, r, -- -p, nv=2, rtc=1
```

---

create a Security World for which:

- a single card from the ACS is required to add a new module
- the default number is required to replace an OCS
- pass phrase replacement is not enabled
- two cards are required to allocate nonvolatile memory
- one card is required to set the real-time clock (applies to SEE only).

## new-world command-line utility output

If **new-world** cannot interpret the command line, it displays its usage message and exits.

If you attempt to set a quorum for a feature that you have disabled or if you attempt to set a quorum too high, **new-world** displays an error and exits.

If the module is not in the pre-initialization mode, **new-world** advises you that you must put the module in this mode and waits until you have changed the module mode before continuing. For more information, see Appendix I: [Checking and changing module mode](#)

**new-world** provides the following reasons why the module is not ready for programming:

| Output                            | Action to take                                                                                                                                                                                                                                    |
|-----------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Not in initialization mode</b> | <ol style="list-style-type: none"> <li>1 Move the switch to the initialization mode position.</li> <li>2 Clear the module by running the command <b>nopclearfail --clear --all</b>.</li> <li>3 Run the <b>new-world</b> command again.</li> </ol> |
| <b>Not in maintenance mode</b>    | <ol style="list-style-type: none"> <li>1 Move the switch to the maintenance mode position.</li> <li>2 Clear the module by running the command <b>nopclearfail --clear --all</b>.</li> <li>3 Run the <b>new-world</b> command again.</li> </ol>    |
| <b>Not in operational mode</b>    | <ol style="list-style-type: none"> <li>1 Move the switch to the operational mode position.</li> <li>2 Run the command <b>nopclearfail --clear --all</b>.</li> <li>3 Run the <b>new-world</b> command again.</li> </ol>                            |

Note If the module is in the initialization mode, **new-world** prompts you for smart cards and pass phrases as required.

When **new-world** has initialized the module, restart the module in the operational mode, as described in [Putting a module in operational mode](#) on page 306.

## Creating a Security World with KeySafe

To create a Security World with KeySafe:

- 1 Start KeySafe. (For an introduction to KeySafe and information on starting the software, see Appendix A: [Using KeySafe](#).)
- 2 Check the Module Status tree to ensure that you have a module in the initialization mode (**PreInitMode**). See Appendix I: [Checking and changing module mode](#).
- 3 Click the **Modules** menu button, which takes you to the Module Operations panel.
- 4 Click the **Initialize Security World** navigation button.

- 5 If KeySafe finds existing Security World data, it takes you to the Existing Security World panel to confirm whether or not you want to delete this data by overwriting the existing Security World:



**Note** Replacing an existing Security World in this way does not delete the Security World's host data and recovery and replacement data, but renames the existing **kmdata/local** directory in which these reside as **kmdata/local\_#** (where **#** is an integer assigned depending on how many Security Worlds have been previously saved during overwrites).

To overwrite an existing Security World, click the **Overwrite Security World** button on the Existing Security World panel. KeySafe then takes you to the Initialize Security World panel.

- 6 If KeySafe does not find existing Security World data, it takes you to the Initialize Security World panel.
- 7 Enter the total number of cards (**N**) that you wish to have in the ACS. This must be a value in the range 1 – 64.

- 8 Enter the number of Administrator Cards that are needed to authorize any action. This number (***K***) must be less than or equal to the total number of cards (***N***).

Note We recommend that you do *not* create an ACS for which ***K*** is equal to ***N***, because you cannot replace such an ACS even if only 1 card is lost or damaged.

- 9 Select the Cipher suite for the Security World—that is, whether the Security World key is to be an AES key (original) or AES key (SP800-131).
- 10 Select whether or not you want the new Security World to comply with the roles and services, key management, and self-test sections of the FIPS 140-2 standard at level 3.

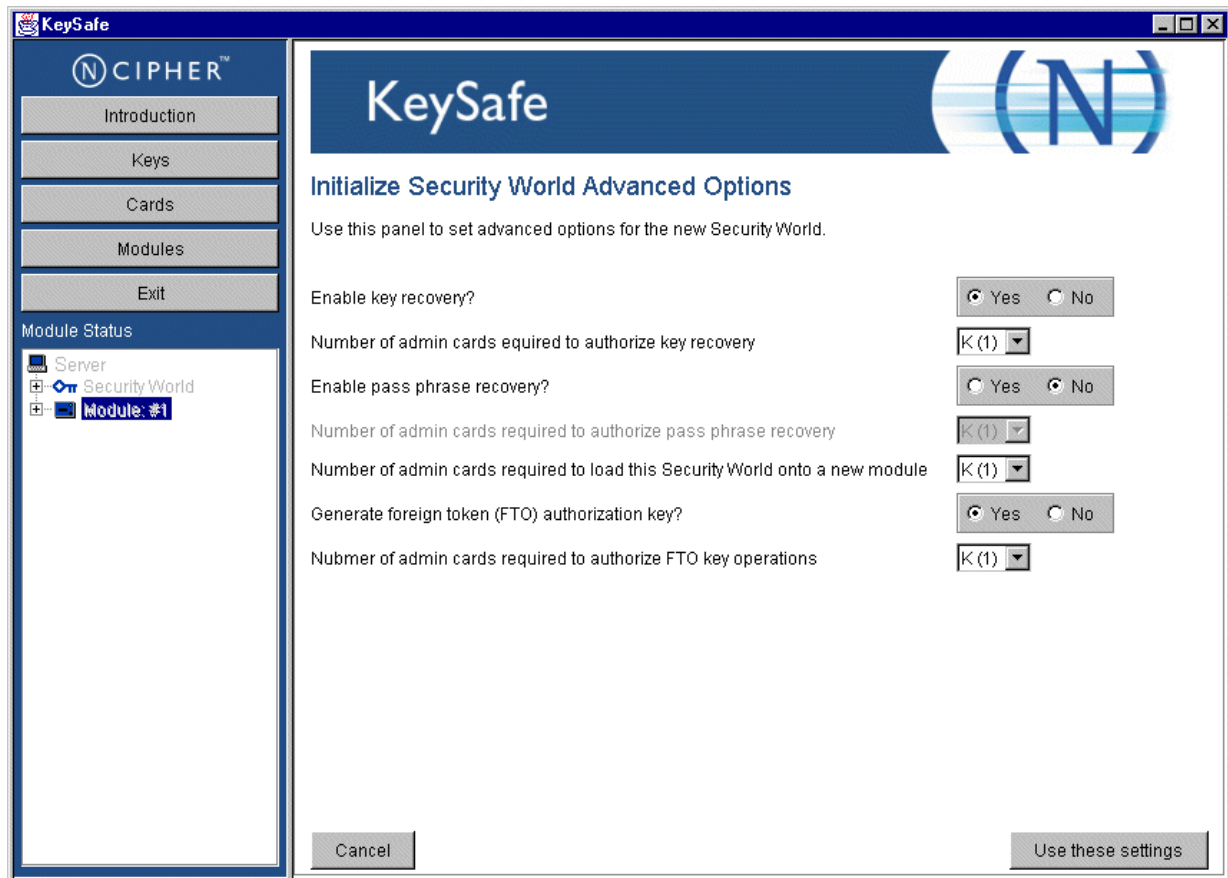
Note This option provides compliance with the letter of the FIPS 140-2 level 3 standard, but does not improve the security of your keys. It is included for those customers who have a regulatory requirement for compliance.

- 11 To configure SEE options, proceed to the **SEE options** panel, and follow these steps:
  - a Select whether or not you want to generate a real-time clock (RTC) authorization key, and the number of Administrator Cards required to change RTC details. This option is not applicable unless you have purchased and installed the CodeSafe Developer kit.
  - b Select whether or not you want to generate a nonvolatile memory (NVRAM) authorization key, and the number of Administrator Cards required to change NVRAM details. This option is not applicable unless you have purchased and installed the CodeSafe Developer kit.
  - c Choose the parameters for SEE debugging and the number of Administrator Cards required to change SEE debugging settings. These are:
    - **Restricted**
    - **Generate Authorization Key**
    - **No Access Control.**

Click the **Use these settings** button to return to the **Initialize Security World** panel.

Note If you wish to use SEE, you must have ordered and enabled it as described in [Enabling optional features on the module](#) on page 60.

- 12 KeySafe always generates recovery and replacement data for the Security World key. However, if you want to be able to recover Operator Cards and application keys, you must proceed to the **Initialize Security World Advanced Options** panel:



Use the **Initialize Security World Advanced Options** panel to create key recovery and pass phrase replacement data that is protected by the cryptographic keys on the ACS. This does not give Thales, or any other third party, access to your keys. Keys can only be recovered by using the Administrator Cards.



We recommend that you do not disable the recovery and replacement option.

If you disable OCS and softcard replacement, you can never replace lost or damaged OCSs generated for that Security World. Therefore, you could never recover any keys protected by lost or damaged OCSs, even if the keys themselves were generated as recoverable [which is the default for key generation].

OCS and softcard replacement cannot be enabled after Security World creation without reinitializing the Security World and discarding all the existing keys within it.

- a Select whether or not you want to enable OCS and softcard replacement, and the number



of Administrator Cards required to authorize key recovery.

- b Select whether or not you want to enable pass phrase replacement, and the number of Administrator Cards required to authorize pass phrase replacement.
- c Specify the number of Administrator Cards required to authorize loading this Security World on to another module.

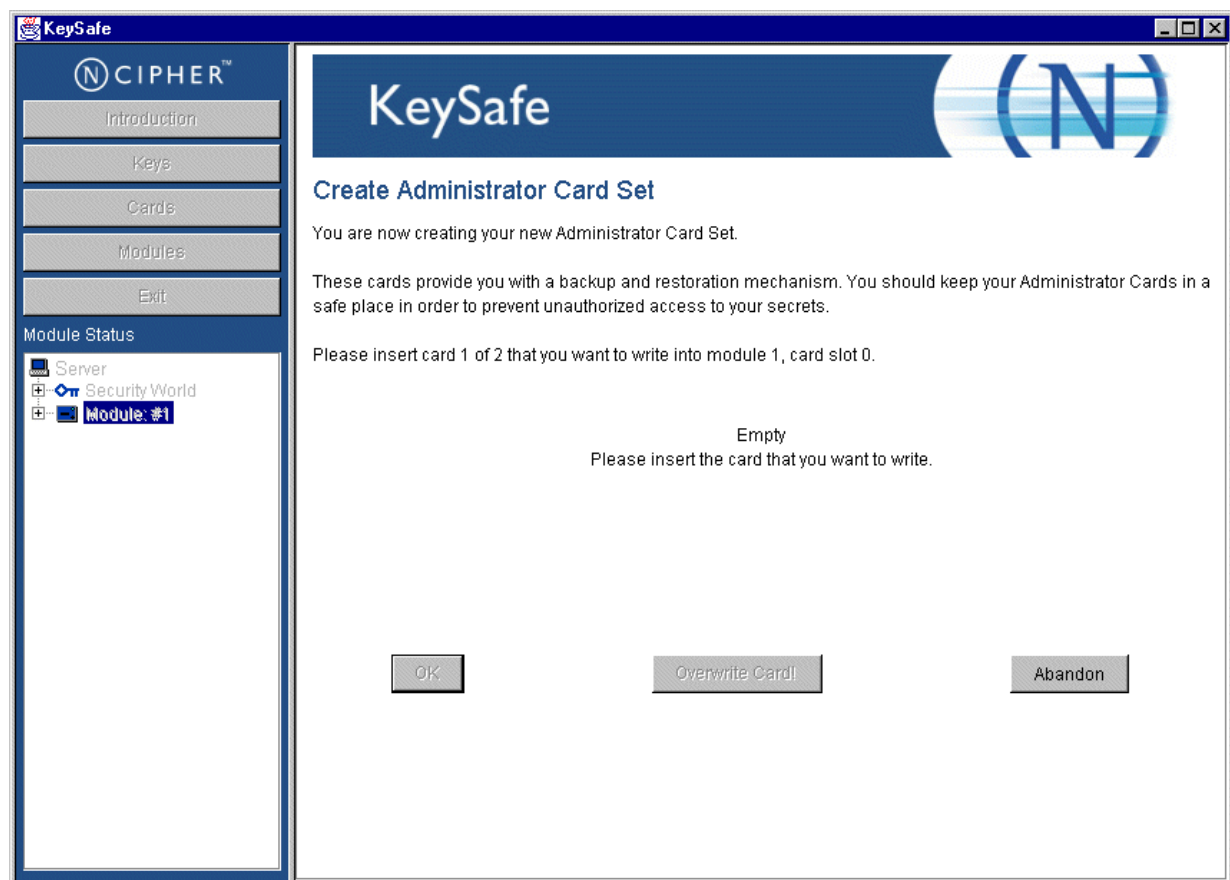
Click the **Use these settings** button to return to the Initialize Security World panel.



If you select **No** for the **Do you want to enable key recovery?** option, you cannot replace lost or damaged Operator Card Sets and, therefore, cannot access keys that are protected by such cards. This feature cannot be enabled later without reinitializing your Security World and discarding all your existing keys.

**Note** To use extended debugging from the module, you must set **SEEDebugging** to **Unrestricted**.

- 13 Click the **Initialize Security World!** button. This takes you to the Create Administrator Card Set panel:

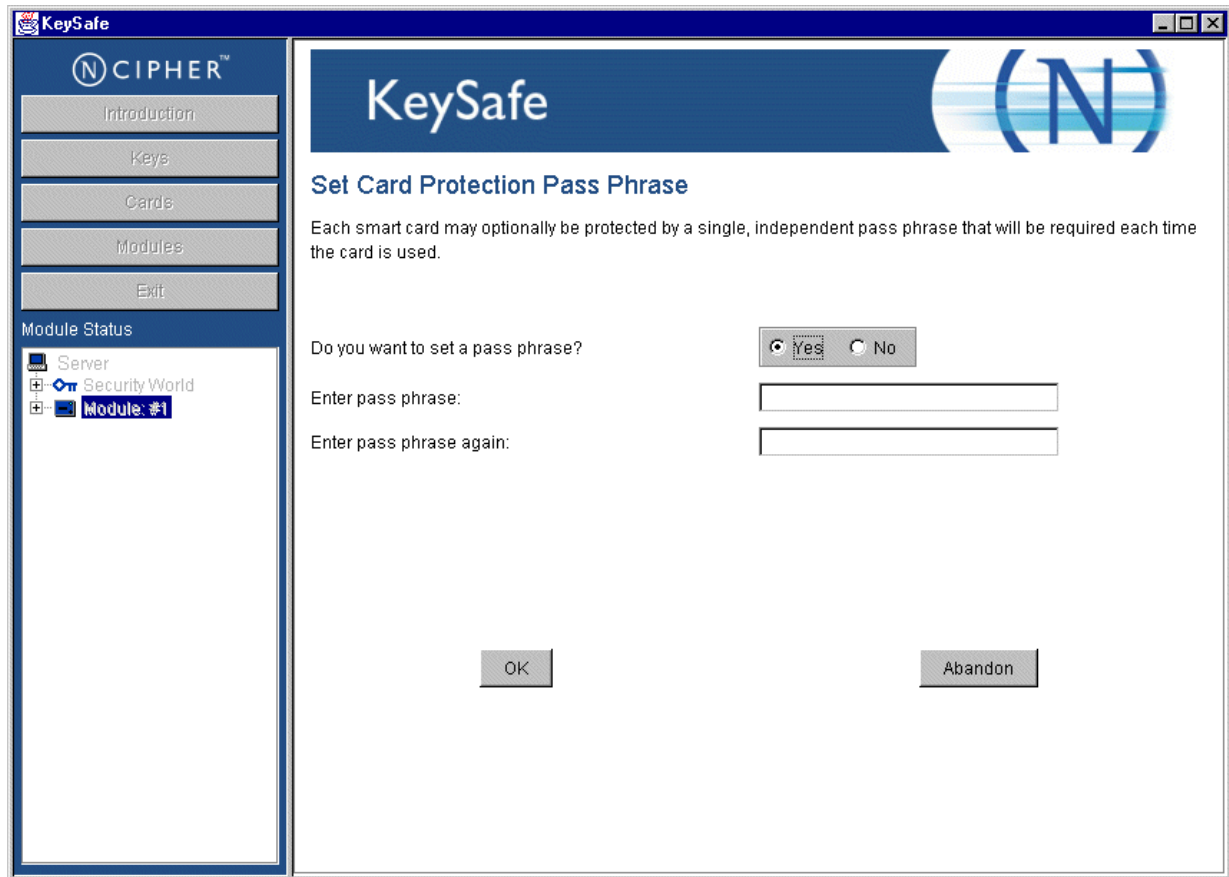


KeySafe prompts you to insert the cards that are to form the ACS.

- 14 Insert a blank card and click the **OK** button. If you insert a non-blank card that KeySafe can erase, the **Erase Card!** button is enabled, giving you the option of overwriting that card.

**Note** When creating a card set, KeySafe recognizes a card that belongs to the set before the card set is complete. If you accidentally insert an already written card from the card set you are in the process of creating, KeySafe warns you of this.

15 KeySafe takes you to the Set Card Protection Pass Phrase panel:



To set a pass phrase for this Administrator Card:

- a Select the **Yes** option.
- b Enter the same pass phrase in both text fields.
- c Click **OK**.

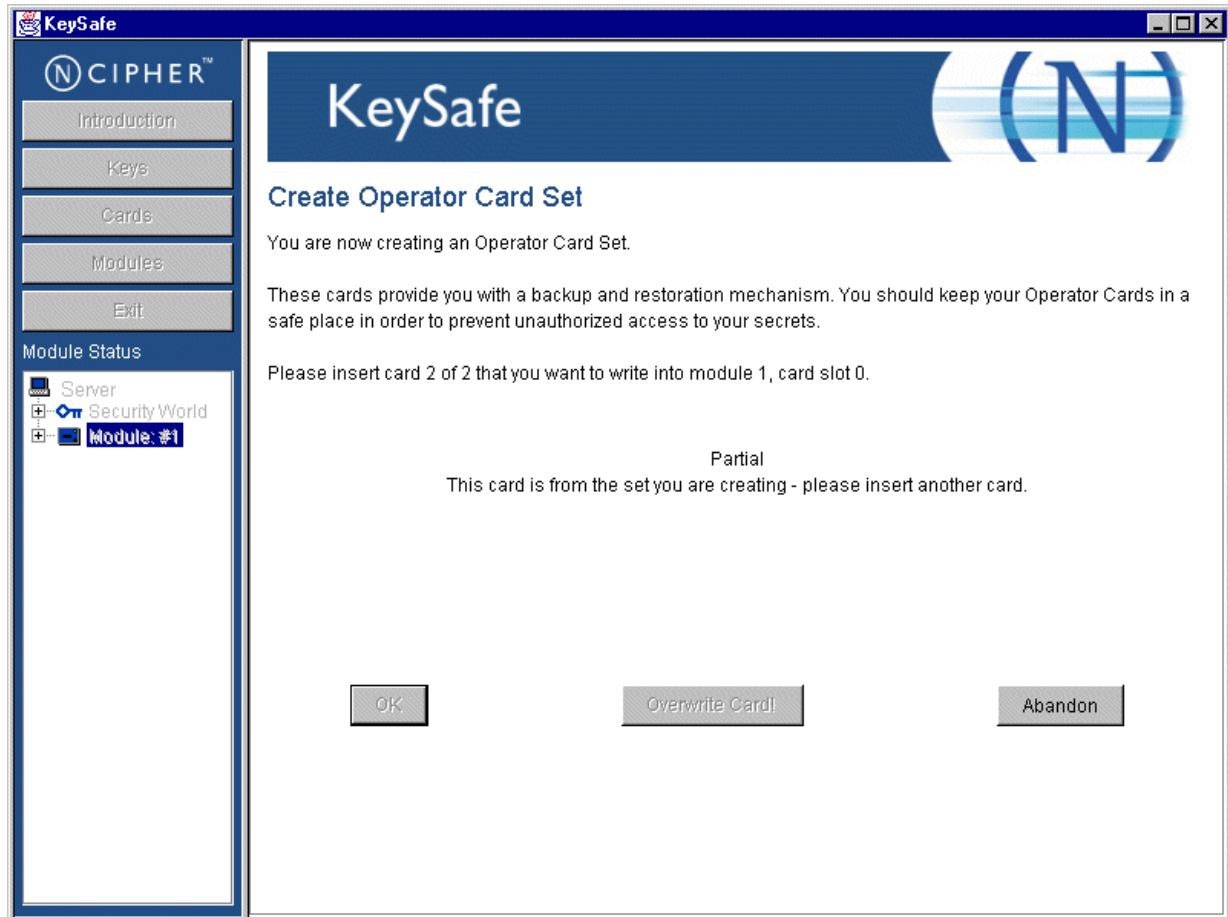
A given pass phrase is associated with a specific card, so each card can have a different pass phrase. You can change these pass phrases at any time by using the KeySafe Examine/Change Card option (available from the Card Operations panel) or the **cardpassphrase** command-line utility.

If you do not want to set a pass phrase for this Administrator Card:

- d Select the **No** option.
- e Click **OK**.

16 KeySafe then prompts you for the next card (if any).

**Note** When creating a card set, KeySafe recognizes a card that belongs to the set before the card set is complete. If you accidentally insert a card to be written again after it has already been written, you see the following warning:



17 After you have created all the Administrator Cards, KeySafe creates a new module key.

When initialization is complete, KeySafe displays a message: **Security world successfully initialized**. Click the **OK** button, and KeySafe returns you to its introduction panel.

18 After you have added a module to the Security World, restart the module in the operational mode (as described in [Putting a module in operational mode](#) on page 306).

If you have more than one module on this server, you can use the KeySafe **Reprogram Module** option to incorporate a new module into your Security World (or to restore an existing module after a firmware upgrade), as described in [Adding or restoring a module to the Security World](#) on page 94. Alternatively, you can also incorporate a new module into your Security World by using the **new-world** command-line utility; see [Adding a module to a Security World with new-world](#) on page 97.

## After you have created a Security World

Store the ACS in a safe place.



If you lose more than  $N$  minus  $K$  of these Administrator Cards you cannot restore the Security World or lost Operator Cards. For example, if you have a 2/3 ACS and you lose more than one card, you cannot restore the Security World. If you have created an Administrator card set where  $K = N$ , then the loss of one card stops you from being able to restore the Security World.

To prevent this situation from occurring, replace lost or damaged cards from the ACS as soon as you discover the loss or damage. For more information, see [Replacing the Administrator Card Set](#) on page 151.



The security of the keys that you create within this Security World is wholly dependent on the security of these smart cards.

The Security World host data is stored in the directory to which the **NFAST\_KMLOCAL** environment variable points (see [Security World files](#) on page 73). The data in this directory is encrypted. Ensure that this directory is backed up regularly. Check the file permissions for this directory. Ensure that the nFast Administrator role, and any user that you want to be able to create Operator Cards or keys, has write permission for this directory. All other valid users must have read permission.

**Note** By default, the Key Management Data directory, and sub-directories, inherit permissions from the user that creates them. Installation of Security World Software must be performed by a user with Administrator rights that allow read and write operations, and the starting and stopping of applications.

The module can now be used to create Operator Cards and keys for the new Security World.

## Displaying information about your Security World

To display information about the status of your Security World, use the **nfkminfo** command-line utility. For more information, see [Displaying information about a Security World with nfkminfo](#) on page 94.

To view details of Operator Cards in a Security World, use KeySafe.

## Displaying information about a Security World with `nfkminfo`

To display information about a Security World from the command line, run the command:

---

```
nfkminfo -w|--world-info [-r|--repeat] [-p|--preload-client-id]
```

---

In this command, the **-w|--world-info** option specifies that you want to display general information about the Security World. This option is set by default, so you do not need to include it explicitly.

Optionally, the command can also include the following:

| Option                        | Description                                                                                                                                                                   |
|-------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-r --repeat</b>            | This option repeats the information displayed. There is a pause at the end of each set of information. The information is displayed again when you press <code>Enter</code> . |
| <b>-p --preload-client-id</b> | This option displays the preloaded client ID value, if any.                                                                                                                   |

To output a detailed list of Security World information, run **nfkminfo** with the **-w|--world-info** option (with or without the other options). For a description of the fields in this list, and more information about using **nfkminfo**, see [nfkminfo: information utility](#) on page 259.

## Adding or restoring a module to the Security World

When you have created a Security World, you can add additional modules to it. These additional modules can be on the same host computer as the original module or on any other host. The modules may have previously been removed from the same Security World, that is, the Security World can be restored on a module by adding the module to the Security World again.

You can also restore a module to a Security World to continue using existing keys and Operator Cards:

- after you upgrade the firmware
- if you replace the module.

**Note** The additional modules can be any nShield modules.

To add a module to a Security World, you must:

- have installed the additional module hardware, as described in the *Quick Start Guide*.

**Note** After installing additional module hardware and restarting host machine, you must stop and then restart the hardserver as described in [Stopping and restarting the hardserver](#) on page 70. This ensures that the added module is recognized and accessible.

- have a copy of the Security World data on this host. This is the host data written by KeySafe or **new-world** when you created the Security World. This data is stored in the **local** directory within the Key Management Data directory.

**Note** If the Key Management Data directory is not in the default location, ensure that the **NFAST\_KMDATA** environment variable is set with the correct location for your installation.

- be logged in to the host computer as root; see [Hardserver start-up settings](#) on page 59 and [server\\_startup](#) on page 284.
- have started the module in pre-initialization state. For more information about module modes, see Appendix I: [Checking and changing module mode](#).
- possess a sufficient number of cards from the ACS and the appropriate pass phrases.

Adding or restoring a module to a Security World:

- erases the module
- reads the required number of cards (**K**) from the ACS so that it can re-create the secret
- reads the Security World data from the computer's hard disk
- uses the secret from the ACS to decrypt the Security World key
- stores the Security World key in the module's nonvolatile memory.

After adding a module to a Security World, you cannot access any keys that were protected by a previous Security World that contained that module.

**Note** It is not possible to program a module into two separate Security Worlds simultaneously.

Initialization removes any data stored in a module's nonvolatile memory (for example, data for an SEE program or NVRAM-stored keys). To preserve this data, you must back it up before initializing the module and restore it after the module has been reprogrammed. We provide the **nvrn-backup** utility to enable data stored in nonvolatile memory to be backed up and restored.

In order to continue using existing keys and Operator Cards, you must reprogram the module:

- after you upgrade the firmware
- if you replace the module
- if you need to add a module to an existing Security World.



## Adding a module to a Security World with new-world

- 1 Open a command window and type the command:

---

```
new-world [-l|--program] [-S|--no-remoteshare-cert] [-m|--module=MODULE]
```

---

In this command:

### **-l|--program**

This option tells **new-world** to add a module to an existing Security World (in the Key Management Data directory). If you have multiple modules available, you can use the **-m|--module=MODULE** option to specify a module. If you do not specify a module **new-world** adds all available modules to the Security World.

### **-S|--no-remoteshare-cert**

These options tell **new-world** not to make the module a target for remote shares.

### **-m|--module=MODULE**

This option specifies the module to use (by its **ModuleID**). If you have multiple modules and do not specify a module, **new-world** adds all available modules to the existing Security World.

If **new-world** cannot find the key-management data, it displays the message:

---

```
new-world: no existing world to load.
```

---

If you intend to initialize the module into a new Security World, run **new-world** with the **-i** option.

If the module is not in the pre-initialization state, **new-world** displays an error and exits. For more information about module modes, see Appendix I: [Checking and changing module mode](#).

If the module is in the pre-initialization state, **new-world** prompts you for cards from the Security World's ACS and to enter their pass phrases as required.

- 2 After **new-world** has reprogrammed the module, restart the module in the operational state. For more information about module modes, see Appendix I: [Checking and changing module mode](#).

- 3 Store the ACS in a safe place.

**Note** If any error occurs [for example, if you do not enter the correct pass phrases], the module is reset to the factory state. The module does not form part of the Security World unless you run **new-world** again.

## Transferring keys between Security Worlds

You use the command-line utilities **mk-reprogram** and **key-xfer-im** are used to transfer keys between Security Worlds.

**Note** To transfer existing Security World data into an SP800-131 Security World, use the **migrate-world** command-line utility instead; see [Migrating keys to an SP800-131 Security World](#) on page 104.

To transfer keys between Security Worlds:

- 1 Ensure that the source Security World (from which you will transfer keys) has the necessary features enabled:
  - OCS and/or softcard replacement.
  - Key recovery or module-protected keys.

**Note** The destination Security World does not need to have these options enabled.

- 2 Move the Security World files (by default, these are the files in the **/opt/nfast/kmdata/local** Security World directory) for the source and destination Security Worlds into directories named **source** and **destination** respectively. For more information, see [Security World files](#) on page 73.

- 3 Decide which Security World to use as the *working Security World* when running the **key-xfer-im** command-line utility.

The working Security World can be any Security World that is not compliant with FIPS 140-2 level 3. If both your source and destination security are compliant FIPS 140-2 level 3, you can create a temporary, dummy Security World that is not compliant with FIPS 140-2 level 3 to use as the working Security World.

The following table shows various possible configurations:

| Source Security World | Destination Security World | Working Security World | Additional module to be added (mk-reprogram) | ACL export change options (key-xfer-im) |
|-----------------------|----------------------------|------------------------|----------------------------------------------|-----------------------------------------|
| non-FIPS              | non-FIPS                   | source                 | destination                                  | --                                      |
| non-FIPS              | non-FIPS                   | destination            | source                                       | --                                      |
| non-FIPS              | FIPS                       | source                 | destination                                  | --export-delete                         |
| FIPS                  | non-FIPS                   | destination            | source                                       | --export-add                            |
| FIPS                  | FIPS                       | Dummy                  | source and destination                       | --                                      |

Note In this table, "FIPS" refers to Security Worlds that are compliant with FIPS 140-2 level 3 and "non-FIPS" refers to Security Worlds that are not compliant with FIPS 140-2 level 3.

- 4 Ensure that you have a quorum for each relevant card set:
  - ACS for the source and destination Security Worlds (and for the dummy Security World, if needed when both the source and destination are compliant with FIPS 140-2 level 3).
  - OCSs for the destination Security World.

Note If necessary, create OCSs for the destination Security World. For more information, see [Creating Operator Card Sets \(OCSs\)](#) on page 116.

- 5 Add the HSM to the working Security World. For more information, see [Adding or restoring a module to the Security World](#) on page 94.

- 6 Program the module key (or keys) from Security Worlds that are not the working Security World into the working Security World by running a command of the form:

---

```
mk-reprogram --owner <working_security_world_dir> add <non-working_security_world_dir>
```

---

In this command, *<working\_security\_world\_dir>* is the Security World directory of the working Security World and *<non-working\_security\_world\_dir>* is the working directory for the Security World that is not the working Security World. In the following example, the working Security World is the destination Security World:

---

```
mk-reprogram --owner /opt/nfast/destination/local add /opt/nfast/source/local
```

---

Supply any appropriate ACS cards (and pass phrases) as prompted.

If you are using a dummy Security World to transfer keys between two Security Worlds that are compliant with FIPS 140-2 level 3, you must run `mk-reprogram` twice: once to transfer module keys from the source and destination Security Worlds to the dummy Security World, as in the following example commands:

---

```
mk-reprogram --owner /opt/nfast/Dummy/local add /opt/nfast/source/local
mk-reprogram --owner /opt/nfast/Dummy/local add /opt/nfast/destination/local
```

---

- 7 Transfer a key (or keys) by running the **key-xfer-im** command-line utility:

---

```
key-xfer-im SOURCE-KMDATA-LOCAL DESTINATION-KMDATA-LOCAL NEW-PROTECT KEY-FILE [KEY-FILE ...] [NEW-PROTECT
KEY-FILE [KEY-FILE ...]]
```

---

In this command:

**<SOURCE-KMDATA-LOCAL>**

The full path name of the **kmdata** file for the source Security World.

**<DESTINATION-KMDATA-LOCAL>**

The full path name of the **kmdata** file for the target Security World.

**<NEW-PROTECT>**

The protection for the key in the target Security World.

You must specify either **--module** or **--cardset**. If you specify **--cardset**, it must be followed by the key hash for the destination card set.

In addition, you can also specify options to configure the key protection further:

**--export-leave**

This option is used to leave the key's list of operations requiring authorization from the Administrator Card Set (ACS)= the same. This is the default.

**--export-add**

This option is used to add export to the key's list of operations requiring authorization from the ACS. This option is available only when exporting keys from a strict FIPS 140-2 level 3 Security World into a non-strict FIPS 140-2 level 3 Security World.

**--export-delete**

This option is used to remove export from the key's list of operations requiring authorization from the ACS. This option is available only when exporting keys from a non-strict FIPS 140-2 level 3 Security World into a strict FIPS 140-2 level 3 Security World.

**--aclbase-recovery**

This option is used to base the Access Control List (ACL) of the exported key on the ACL in the recovery key blob. This is the default.

**--aclbase-working**

This option is used to base the ACL of the exported key on the ACL in the working key blob.

**<KEY-FILE>**

This is the full path of source **kmdata** file for the key. The module must have module keys from both worlds: program it into one world with **new-world** and use **mk-reprogram** to add the other module key. If transferring between Strict FIPS-140 level 3 and non-strict worlds, the module's owning world must be non-strict.

The following example command demonstrates transfer of a module key between source and destination Security Worlds that are both compliant with FIPS 140 2 level 3:

---

```
key-xfer-im /opt/example/source/local /opt/example/destination/local --module
/opt/example/source/local/key_pkcs11_ua753157d8a9b86e943c5e4a6c100963f26839749a
```

---

The following example command demonstrates transfer of a card set key from a source Security World that is compliant with FIPS 140 2 level 3 to a destination Security World that is not:

---

```
key-xfer-im /opt/example/destination/local /opt/example/Destination/local --cardset
1234578..cardsethash...abcdef --export-add
/opt/example/source/local/key_pkcs11_ua753157d8a9b86e943c5e4a6c100963f26839749a
```

---

The following example command demonstrates transfer of a card set key from a source Security World that is not compliant with FIPS 140 2 level 3 to a destination Security World that is :

---

```
key-xfer-im /opt/example/source/local /opt/example/destination/local --cardset
1234578..cardsethash...abcdef --export-delete
/opt/example/source/local/key_pkcs11_ua753157d8a9b86e943c5e4a6c100963f26839749a
```

---

Supply appropriate cards and pass phrases for the ACS (source Security World) and OCSs (destination Security World) as prompted.

**Note** You could use any directory instead of the **example** directory shown in these example commands.

- 8 If necessary, copy the Security World files to the Security World directory (by default, **/opt/nfast/kmdata/local**), and then add the HSM to the destination Security World. For more information, see [Adding or restoring a module to the Security World](#) on page 94.

9 Check that keys have been transferred. For example:

- Check the Security World directory (by default, **/opt/nfast/kmdata/local**) to see if the key was successfully exported.
- Verify that a module-protected key was imported successfully by running a command of the form:

---

```
preload --module-prot exit
```

---

**Note** You can also use the preload command-line utility to load other types of keys, thereby also verifying that they have been imported successfully.

## Security World migration

The current version of the Security World for nShield enables you to create a Security World that fully complies with NIST Recommendations for the Transitioning of Cryptographic Algorithms and Key Sizes (SP800-131). The software also includes a **migrate-world** command-line utility that you can use for migrating existing keys into the new Security World. We recommend that where compliance with SP800-131 is required, you create a new Security World and new keys within that world. This utility is provided as a convenience for customers who require compliance with SP800-131 and who need to continue using existing keys.

**Note** If your current Security World or keys are non-recoverable, you cannot use the migration utility to migrate keys. Contact Support.

### About the migration utility

You can run the migration utility in the following modes:

- **Plan mode:** Returns a list of steps for migration and the required card sets and pass phrases.
- **Perform mode:** Enables you to migrate keys interactively.

### Usage and options

---

```
migrate-world [OPTIONS] --source=<source-kmdata-path> [--plan | --perform]
```

---

| Option     | Enables you to...                                                  |
|------------|--------------------------------------------------------------------|
| --version  | View the version number of the utility.                            |
| -h, --help | Obtain information about the options you can use with the utility. |

| Option                                                    | Enables you to...                                                                                                                                                                                                    |
|-----------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| -m, --module                                              | Specify which module ID to use. If no module is specified, the default is module 1.                                                                                                                                  |
| -c <i>CARDSETS</i><br>--cardsets-at-once= <i>CARDSETS</i> | Migrate keys protected by this number of card sets or softcards per ACS loading. This is useful to prevent ACS time-outs if you have very many different card sets or softcards to migrate. [0=unlimited, default=0] |
| -k <i>KEYS</i><br>--keys-at-once= <i>KEYS</i>             | Migrate no more than this number of keys per ACS loading. This is useful to prevent ACS time-outs if you have very many keys to migrate. [0=unlimited, default=0]                                                    |
| --source= <i>SOURCE</i>                                   | Specify the path to the folder that contains the source Security World data.                                                                                                                                         |
| --plan                                                    | View the list of steps that will be carried out .                                                                                                                                                                    |
| --perform                                                 | Migrate keys interactively.                                                                                                                                                                                          |

## Migrating keys to an SP800-131 Security World

Note Throughout the following sections, the term:

- *Source world* refers to the source Security World from which you want to migrate keys.
- *Destination world* refers to the SP800-131 Security World to which you want to migrate keys.

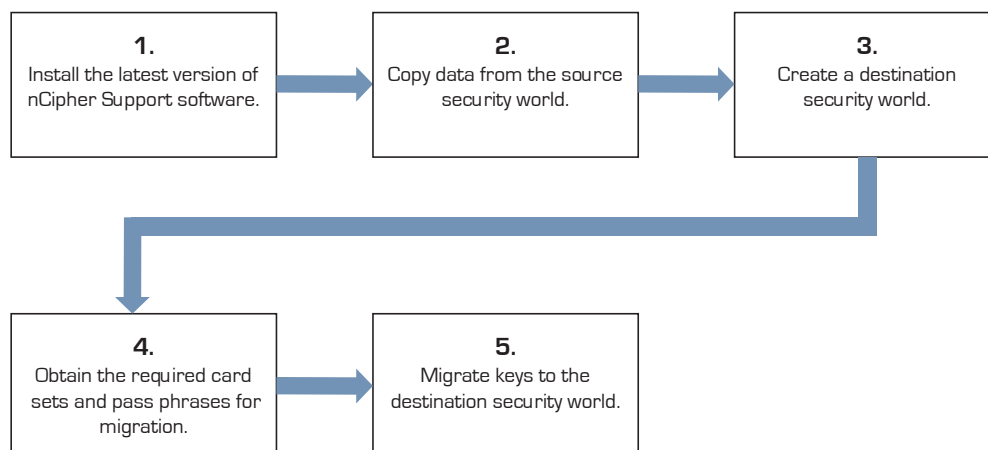
Migrating keys to an SP800-131 Security World involves:

- 1 Preparing for migration, which includes setting up a destination world (if it does not exist already).
- 2 Migrating keys by running the migration utility.

The migration process is shown in the following figure.



Figure 3 Migration process



## Preparing for migration

Before you begin:

- Install the latest version of Security World for nShield from the installation disc. For instructions, see [Support software installation](#) on page 35.
- Copy the source Security World data to a location that can be accessed by the migration tool.
- If the destination world does not exist already, create a new destination world and program the module to use this world.

For instructions, see:

- [Creating a Security World by using new-world](#) on page 79.
- [Creating a Security World with KeySafe](#) on page 85.
- Run the utility in the **plan** mode. The utility returns a list of steps for migration along with the details of the required card sets and pass phrases. Ensure that you have:
  - The required card sets and pass phrases of the source and destination worlds.
  - The appropriate number of blank smart cards to create the required card sets.

## Migrating keys

**Note** If your source world, destination world, or keys are non-recoverable, you cannot migrate keys with the migration utility. Contact Support.

If you do not have a quorum of the ACS of the source and destination worlds, you cannot migrate keys.

During migration, the keys are temporarily in a vulnerable state. To ensure the security of your keys, we recommend that:

- The migration process is overseen by ACS-holding personnel.
- The end-to-end migration process is completed continuously, without any breaks in the process. This will also reduce the possibility of your ACS experiencing a time-out.

To migrate keys to the destination world:

- 1 Run the migration utility in the **perform** mode with the required options. For information about the usage and options you can use, see [About the migration utility](#) on page 103.

Ensure that the data for the current (destination) world is in the standard location for Security World data, derived from one of the following:

- Either the environment variable **NFAST\_KMLOCAL** or **NFAST\_KMDATA**.
- The default directory: **/opt/nfast/kmdata/local**.

- 2 If the module is not configured to use the destination world, the utility prompts you to program the module and supply the ACS of the destination world.

The next steps in the migration procedure vary depending on whether you are transferring keys from:

- Either a FIPS or non-FIPS Security World to a non-FIPS Security World.
- A Non-FIPS Security World to a FIPS Security World.
- A FIPS Security World to another FIPS Security World.

Depending on the scenario, the utility guides you through specific steps, prompting you to supply the required card sets and pass phrases. The high-level steps and the requirements for each task are described in the following sections. For detailed steps, follow the onscreen instructions.

**Note** The utility prompts you to change the module state to either **initialization** or **operational** at various points during the procedure.

For information on changing the mode, see [Changing the module's mode](#) on page 303.

### Any Security World (FIPS or non-FIPS) to non-FIPS Security World

| Step                                                                                                                                               | Requirements                                                                                                                                                 |
|----------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Program the source module key into the destination world. This key is used for decrypting keys when they are transferred to the destination world. | <ul style="list-style-type: none"> <li>• ACS of the destination world.</li> <li>• ACS of the source world.</li> </ul>                                        |
| Create replacement softcards and/or Operator Card Sets, as appropriate. You can set pass phrases of your choice.                                   | <ul style="list-style-type: none"> <li>• ACS of the destination world.</li> <li>• Blank smart cards to create Operator Card Sets, if appropriate.</li> </ul> |
| Migrate keys to the destination world.                                                                                                             | The replacement softcards or Operator Card Sets and pass phrases.                                                                                            |
| <p><b>Note</b> The module is programmed to use the new world by default.</p>                                                                       |                                                                                                                                                              |

### Non-FIPS Security World to FIPS Security World

| Step                                                                                                             | Requirements                                                                                                                                      |
|------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------|
| Create replacement softcards and/or Operator Card Sets, as appropriate. You can set pass phrases of your choice. | <ul style="list-style-type: none"> <li>• ACS of the destination world.</li> <li>• Blank smart cards to create the OCS, if appropriate.</li> </ul> |
| Program the source world into the module.                                                                        | ACS of the source world.                                                                                                                          |

| Step                                                                                                                                               | Requirements                                                                                                                           |
|----------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------|
| Program the destination module key into the source world. This key is used for encrypting keys when they are transferred to the destination world. | <ul style="list-style-type: none"> <li>ACS of the source world.</li> <li>ACS of the destination world.</li> </ul>                      |
| Migrate keys to the destination world.                                                                                                             | <ul style="list-style-type: none"> <li>ACS of the source world.</li> <li>The replacement softcards or OCS and pass phrases.</li> </ul> |
| If you want to start using the destination world, program the module to use the new Security World.                                                | ACS of the destination world.                                                                                                          |

## FIPS Security World to FIPS Security World

**Note** If both Security Worlds are FIPS, the utility creates an intermediate non-FIPS Security World (intermediate world), which is removed after migration. The utility also prompts you to create a 1-of-1 ACS for the intermediate world; we recommend that you destroy this ACS after migration.

| Step                                                                                                                                                          | Requirements                                                                                                                                  |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------|
| Create replacement softcards and/or Operator Card Sets, as appropriate. You can set pass phrases of your choice.                                              | <ul style="list-style-type: none"> <li>ACS of the destination world.</li> <li>Blank smart cards to create the OCS, if appropriate.</li> </ul> |
| Create an intermediate world and card set.                                                                                                                    | Blank smart card to create the ACS for the intermediate world.                                                                                |
| Program the destination module key into the intermediate world. This key is used for encrypting keys when they are transferred to the destination world.      | <ul style="list-style-type: none"> <li>ACS of the intermediate world.</li> <li>ACS of the destination world.</li> </ul>                       |
| Program the source module key into the intermediate world. This key is used for decrypting keys when they are transferred to the intermediate Security World. | <ul style="list-style-type: none"> <li>ACS of the intermediate world.</li> <li>ACS of the destination world.</li> </ul>                       |
| Migrate keys to the intermediate world.                                                                                                                       | ACS of the source world.                                                                                                                      |
| Migrate keys to the destination world.                                                                                                                        | <ul style="list-style-type: none"> <li>ACS of the intermediate world.</li> <li>The replacement softcards or OCS and pass phrases.</li> </ul>  |
| If you want to start using the new world, program the module to use the destination world.                                                                    | ACS of the destination world.                                                                                                                 |

## Verifying the integrity of the migrated keys

To verify the integrity of the migrated keys, run the **nfmverify** utility with the following options, as appropriate:

- If the keys are module-protected, run the utility with one of the following options:
  - **-L** option, which checks the ACL of a loaded key instead of the generation certificate.
  - **-R** option, which checks the ACL of a key loaded from a recovery blob.
- If the keys are protected by cardsets or softcards, run the utility with the **-R** option.

**Note** Do not use the **nfmverify** utility with the default **-C** option. If you use this option, the utility returns errors because the certificate in the old Security World and the certificate in the new Security World are different.

## Troubleshooting

**Note** If you encounter any errors that are not listed in the following table, contact Support.

| Error                                                                                                                                                                                                                                                                                         | Explanation                                                                                                                                                                                                                                                                                                   | Action           |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------|
| No keys to migrate.                                                                                                                                                                                                                                                                           | Any migrateable keys found in the source world already exist in the destination world.<br><br>The migration utility returns this error if: <ul style="list-style-type: none"> <li>• The keys have already been migrated.</li> <li>• All keys are non-recoverable and therefore cannot be migrated.</li> </ul> | None.            |
| <ul style="list-style-type: none"> <li>• Source world has indistinguishable cardsets or softcards.</li> <li>• Destination world has indistinguishable keys.</li> </ul>                                                                                                                        | There are irregularities in one of the Security Worlds, but these irregularities <i>do not</i> affect the migration process.                                                                                                                                                                                  | None.            |
| <ul style="list-style-type: none"> <li>• Destination world has indistinguishable card sets or softcards.</li> <li>• Source world has indistinguishable card sets or softcards.</li> <li>• Source world has indistinguishable keys.</li> <li>• Cannot determine protection of keys.</li> </ul> | There are problems with one of the Security Worlds.                                                                                                                                                                                                                                                           | Contact Support. |

| Error                                                                                                                                        | Explanation                                                                                                                                                                                                                                                                                      | Action                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|----------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <ul style="list-style-type: none"> <li>Source world not recoverable.</li> <li>Destination world not recoverable.</li> </ul>                  | The source and destination worlds are not recoverable and the keys therefore cannot be migrated.                                                                                                                                                                                                 | If the source world is not recoverable, you cannot use the migration utility to migrate keys. Contact Support.                                                                                                                                                                                                                                                                                                                                 |
| Missing Security World at <i>PATH</i> .                                                                                                      | <ul style="list-style-type: none"> <li>The path for the source world is wrong.</li> <li>There is no Security World data at the location that was specified when running the migration utility.</li> </ul>                                                                                        | <p>Supply the correct path to the source world.</p> <p>If you have supplied the correct path to the directory that contains the source world data, the migration utility has not found a destination world. Ensure that the destination world is the current world.</p>                                                                                                                                                                        |
| Source world is the same as the destination world.                                                                                           | <ul style="list-style-type: none"> <li>An incorrect path was supplied for the source world data when running the utility.</li> <li>The destination world data does not exist in the default location defined by the environment variable <b>NFAST_KMLOCAL</b> or <b>NFAST_KMDATA</b>.</li> </ul> | <ul style="list-style-type: none"> <li>Run the utility with the correct path to the source world data.</li> <li>Move the source world data to a different location and then copy the destination world data to the default location.</li> <li>If the default location is defined by an environment variable, configure the variable to point to the location of the destination world, which then becomes the new default location.</li> </ul> |
| <p>Cannot find <i>NAME</i> utility, needed by this utility.</p> <p><i>NAME</i> utility is too old, need at least version <i>VERSION</i>.</p> | <p>The software installation is partially completed.</p> <ul style="list-style-type: none"> <li>The software installation is partially completed.</li> <li>The path (in the environment variable for the operating system) might be pointing to an old version of the software.</li> </ul>       | <ul style="list-style-type: none"> <li>Reinstall the software.</li> <li>Ensure that the path points to the latest version of the software.</li> </ul>                                                                                                                                                                                                                                                                                          |
| nFast error: TimeLimitExceeded; in response to SetKM...                                                                                      | The ACS time-out limit has expired.                                                                                                                                                                                                                                                              | Restart the key migration process; see <a href="#">Migrating keys</a> on page 106.                                                                                                                                                                                                                                                                                                                                                             |

## Erasing a module from a Security World

Erasing a module from a Security World deletes from the module all of the secret information that is used to protect your Security World. This returns the module to the factory state. Provided that you still have the ACS and the host data, you can restore the secrets by adding the module to the Security World.

Erasing a module removes any data stored in its nonvolatile memory (for example, data for an SEE program or NVRAM-stored keys). To preserve this data, you must back it up before erasing the module. We provide the **nvrn-backup** utility to enable data stored in nonvolatile memory to be backed up and restored.

In order to erase a module, you must:

- be logged in to your computer as root; see [Hardserver start-up settings](#) on page 59 and [server\\_startup](#) on page 284.
- have started the module in the pre-initialization state. For more information about module modes, see Appendix I: [Checking and changing module mode](#).



You do not need the ACS to erase a module. However, unless you have a valid ACS and the host data for this Security World, you cannot restore the Security World after you have erased it.

After you have erased a module, it is in the same state as when it left Thales (that is, it has a random module key and a known  $K_{NSO}$ ).

## Erasing a module with new-world

The **new-world** command-line utility can erase any modules that are in the pre-initialization state.

To erase modules with the **new-world** utility, run the command:

---

```
new-world [-e|--factory] [-m|--module=MODULE]
```

---

In this command:

| Option                     | Description                                                                                                                                                                                                 |
|----------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-e, --factory</b>       | These options tell <b>new-world</b> to restore a module to its factory state.                                                                                                                               |
| <b>-m, --module=MODULE</b> | These options specify the <b>ModuleID</b> to use. <b>new-world</b> erases only one module at a time. To erase multiple modules, you must run <b>new-world</b> once for every module that you want to erase. |

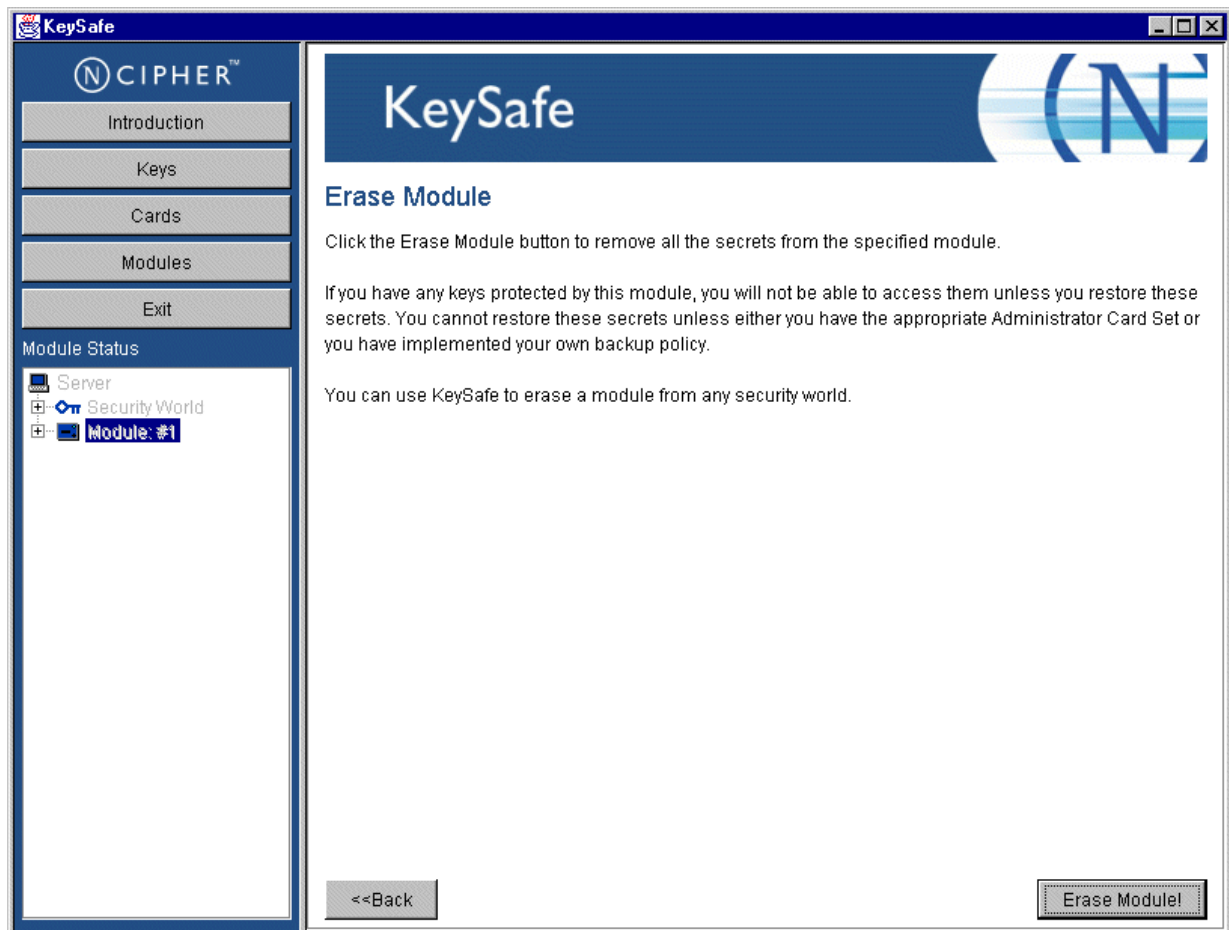
## Output

If **new-world** successfully erased a module, it does not display any messages. Otherwise, **new-world** returns an error message.

## Erasing a module with KeySafe

You can erase a module on a server with KeySafe by following these steps:

- 1 Start KeySafe. (For an introduction to KeySafe and information on starting the software, see Appendix A: [Using KeySafe](#).)
- 2 Click the **Modules** menu button. KeySafe takes you to the Module Operations panel.
- 3 Click the **Erase Module** navigation button. KeySafe takes you to the Erase Module panel:



- 4 Select the module that you want to erase by clicking its listing on the Module Status tree, then click the **Erase Module!** command button.



5 **KeySafe** erases all secrets from the module, returning it to its factory state.

**Note** If you have any keys that were protected by an erased module, you cannot access them unless you restore these secrets. You cannot restore these secrets unless you have the appropriate ACS.

## Erasing a module with **initunit**

The **initunit** command-line utility erases any modules that are in the pre-initialization state.

To erase modules with the **initunit** utility, run the command:

---

```
initunit [-m|--module=MODULE] [-s|--strong-kml]
```

---

In this command, **--module=MODULE** specifies the ID of the module you want to erase. If you do not specify this option, all modules in the pre-initialization state are erased. **--strong-kml** specifies that the module generates an AES (SP800-131) module signing key, rather than the default key.

## Output

If **initunit** is successful, for each module that is in the pre-initialization state, it returns a message similar to this:

---

```
Initialising Unit
Setting dummy HKNSO Module Key Info:
HKNSO
HKM
#####
```

---

Otherwise, **initunit** returns an error message.

## Deleting a Security World

You can remove an existing Security World and replace it with a new one if, for example, you believe that your existing Security World has been compromised. However:

- you are not able to access any keys that you previously used in a deleted Security World

- you must reformat any smart cards that were used as Operator Cards within this Security World *before* you delete it. For more information about reformatting (or erasing) Operator Cards, see [Erasing cards and softcards](#) on page 128.

**Note** If you do not reformat the smart cards used as Operator Cards before you reinitialize your module, you must throw them away because they cannot be used, erased, or reformatted without the old Security World key.



You can, and should, reuse the smart cards from a deleted Security World's ACS. If you do not reuse or destroy these cards, then an attacker with these smart cards, a copy of your data (for example, a weekly backup) and access to any Thales key management module can access your old keys.

To delete an existing Security World:

- 1 Remove all the modules from the Security World.
- 2 Delete the files in the Key Management Data directory.



There may be copies of the Security World data archive saved on your backup media. If you have not reused or destroyed the old ACS, an attacker in possession of these cards could access your old keys using this backup media.



## Chapter 6: Managing card sets and softcards

This chapter describes how to create and manage card sets and softcards, using a Security World.

When you create a Security World, an Administrator Card Set (ACS) is created at the same time. You use the ACS to:

- control access to Security World configuration
- authorize recovery and replacement operations.

The Security World is used to create and manage keys, and the Operator Card Sets (OCSs) and softcards you create with the Security World are used to protect those keys.

A security world offers three levels of key protection:

| Level of protection | Description                                                                                                                      |
|---------------------|----------------------------------------------------------------------------------------------------------------------------------|
| Direct protection   | Keys that are directly protected by the security world are usable at any time without further authorization.                     |
| Softcard            | Keys that are protected by a softcard can only be used by the operator who possesses the relevant pass phrases.                  |
| OCS                 | Keys that are protected by an OCS can only be used by the operator who possesses the OCS and any relevant pass phrases [if set]. |

For more information about creating a Security World, see Chapter 5: [Creating and managing a Security World](#).

For more information about key management, see Chapter 9: [Working with keys](#).

After a Security World has been created, you can use it to create and manage OCSs and softcards (as described in this chapter), as well as to create and manage the keys it protects (see Chapter 9: [Working with keys](#)).

If you want to use the Remote Operator feature to configure smart cards for use with a remote module, see Chapter 8: [Remote Operator Card Sets](#).

## Creating Operator Card Sets (OCSs)

You can use an Operator Card Set (OCS) to control access to application keys. OCSs are optional, but if you require one, create it before you start to use the hardware security device with applications. You must create an OCS before you create the keys that it is to protect.

You can create OCSs that have:

- names for individual cards, as well as a name for the whole card set
- specific *K/N* policies
- optional pass phrases for any card within a given set
- formal FIPS 140-2 level 3 compliance.

**Note** Some third-party applications impose restrictions on the OCS smart card quorums [*K/M*] or the use of smart card pass phrases. For more information, see the appropriate integration guide for the application. Integration guides for third-party applications are available from the Thales web site:  
<http://iss.thalesgroup.com/Resources.aspx>.

OCSs belong to the security world in which they are created. When you create an OCS, the smart cards in that set can only be read by hardware security devices belonging to the same security world. The cards cannot be read, erased, or reformatted by a hardware security device from a different security world.

You can use the following tools to create an OCS:

- the **createocs** command-line utility, as described in [Creating an Operator Card Set from the command line](#) on page 118
- KeySafe, as described in [Creating an Operator Card Set with KeySafe](#) on page 120

## Persistent Operator Card Sets

If you create a standard (non-persistent) OCS, the keys it protects can only be used while the last required card of the quorum remains loaded in the smart card reader of the Thales hardware security device. The keys protected by this card are removed from the memory of the device as soon as the card is removed from the smart card reader. If you want to be able to use the keys after you have removed the last card, you must make that OCS persistent.

Keys protected by a persistent card set can be used for as long as the application that loaded the OCS remains connected to the hardware security device (unless that application removes the keys).

For more information about persistent OCSs, see [Using persistent Operator Card Sets](#) on page 26.

## Time-outs

OCSs can be created with a time-out, so that they can only be used for limited time after the OCS is loaded. An OCS is loaded by most applications at start up or when the user supplies the final required pass phrase. After an OCS has timed out, it is not loadable by another application unless it is removed and reinserted. Time-outs operate independently of OCS persistence.

## FIPS 140-2 level 3-compliant security worlds

When you attempt to create an OCS for a security world that complies with FIPS 140-2 level 3, you are prompted to insert an Administrator Card or Operator Card from an existing set. You may need to specify to the application the slot you are going to use to insert the card. You need to insert the card only once in a session.

## Creating an Operator Card Set from the command line

To create an OCS from the command line:

### 1 Run the command:

---

```
createocs -m MODULE | --module=MODULE -Q | --ocs-quorum=K/N
-N | --name=NAME [-M | --name-cards]
[[-p | --persist] | [-P | --no-persist]] [[-R | --no-pp-recovery] | --pp-recovery]
[-q | --remotely-readable] [-T | --timeout=TIME] [-e | --erase]
```

---

This command uses the following options:

| Option                                          | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|-------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-m <i>MODULE</i>, --module=<i>MODULE</i></b> | This option specifies the number of the hardware security device to be used to create the token. If you only have one hardware security device, <b><i>MODULE</i></b> is 1.                                                                                                                                                                                                                                                                                                                                              |
| <b>-Q   --ocs-quorum=<i>K</i>/<i>N</i></b>      | In this option, <b><i>K</i></b> is the minimum required number of cards. If you do not specify the value <b><i>K</i></b> , the default is 1.<br><br><div style="text-align: center;"> <p><b>Note</b> Some applications do not have mechanisms for requesting that cards be inserted. Therefore any OCSs that you create for use with these applications must have <b><i>K</i>=1</b>.</p> </div> <p><b><i>N</i></b> is the total number of cards. If you do not specify the value <b><i>N</i></b>, the default is 1.</p> |
| <b>-N   --name=<i>NAME</i></b>                  | This option specifies a name for the card set. The card set must be named with this option before individual cards can be named using the <b>-M   --name-cards=<i>NAME</i></b> options.                                                                                                                                                                                                                                                                                                                                 |
| <b>-M   --name-cards</b>                        | Specifying this option allows you to name individual cards within the card set. You can only use this option after the card set has been named by using the <b>--name=<i>NAME</i></b> option. <b>createocs</b> prompts for the names of the cards as they are created. Not all applications can display individual card names.                                                                                                                                                                                          |
| <b>-p   --persist</b>                           | This option creates a persistent card set.                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| <b>-P   --no-persist</b>                        | This option creates a non-persistent card set.                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| <b>-R   --no-pp-recovery</b>                    | This option specifies that pass phrase replacement for this OCS is disabled. Setting this option overrides the default setting, which is that the card pass phrases are replaceable. You can specify the enablement of pass phrase replacement explicitly by setting the <b>--pp-recovery</b> option.                                                                                                                                                                                                                   |
| <b>-q   --remotely-readable</b>                 | This option allows this card set to be read remotely. For information on configuring Remote OCSs, see Chapter 8: <a href="#">Remote Operator Card Sets</a> .                                                                                                                                                                                                                                                                                                                                                            |

| Option                          | Description                                                                                                                                                                                                                                                                                                                                                                           |
|---------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-T -timeout= <i>TIME</i></b> | This option sets the time-out for the card set. Use the suffix <b>s</b> to specify seconds, <b>m</b> for minutes, <b>h</b> for hours, and <b>d</b> for days. If the time-out is set to 0, the OCS never times out. Otherwise, the hardware security device automatically unloads the OCS when the amount of time specified by <b><i>TIME</i></b> has passed since the OCS was loaded. |
| <b>-e -erase</b>                | Specifying this option erases a card (instead of creating a card set). You can specify this option twice in the form <b>-ee</b> to repeatedly erase cards.                                                                                                                                                                                                                            |

If you have created a FIPS 140-2 level 3 compliant Security World, you must provide authorization to create new Operator Cards; **createocs** prompts you to insert a card that contains this authorization. Insert any card from the Administrator Card Set or any Operator Card from the current Security World.

When **createocs** has obtained the authorization from a valid card, or if no authorization is required, it prompts you to insert a card into the hardware security device you specified.

## 2 Insert the smart card to use.

If you insert an Administrator Card from another Security World or an Operator Card that you have just created, **createocs** displays the following message:

---

```
Module x slot n: unknown card
Module x slot n: Overwrite card ? (press Return)
```

---

where **x** is the hardware security device number and **n** is the slot number. If you insert an Operator Card from another Security World, **createocs** displays the following message:

---

```
Module x slot n: inappropriate Operator Card (TokenAuthFailed).
```

---

When you insert a valid card, **createocs** prompts you to type a pass phrase.

**Note** The nCipher PKCS #11 library requires Operator Cards with pass phrases.

**Note** Some applications do not have mechanisms for entering pass phrases. Do not give pass phrases to Operator Cards that are to be used with these applications.

## 3 Type a pass phrase and press **Enter**. Alternatively, press **Enter** if you do not want this card to have a pass phrase.

A pass phrase can be of any length and can contain any character that you can type.

If you entered a pass phrase, **createocs** prompts you to confirm it.

- 4 Type the pass phrase again and press **Enter**.

If the pass phrases do not match, **createocs** prompts you to input and confirm the pass phrase again.

- 5 When the new card has been created, if you are creating a card set with more than one card in it, **createocs** prompts you to insert another card.
- 6 For each additional card in the OCS, follow the instructions from step 2 through step 4.

## Creating an Operator Card Set with KeySafe

KeySafe enables you to create OCSs with:

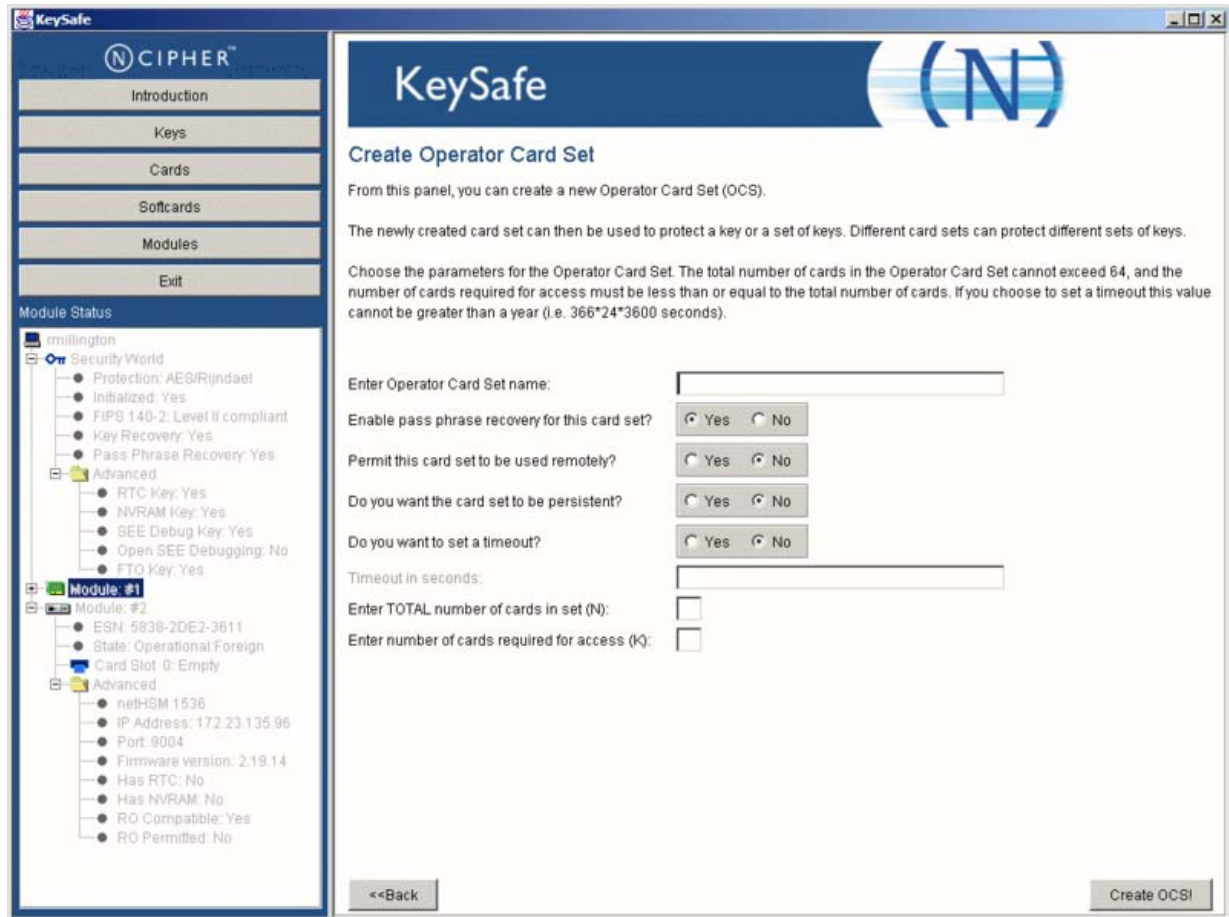
- their own names
- **K/N** policies
- optional pass phrases for any card within the OCS
- formal FIPS 140-2 level 3 compliance.

To create an OCS with KeySafe:

- 1 Start KeySafe. (For an introduction to KeySafe and information on starting the software, see Appendix A: [Using KeySafe](#).)
- 2 Click the **Cards** menu button. KeySafe takes you to the **Card Operations** panel.

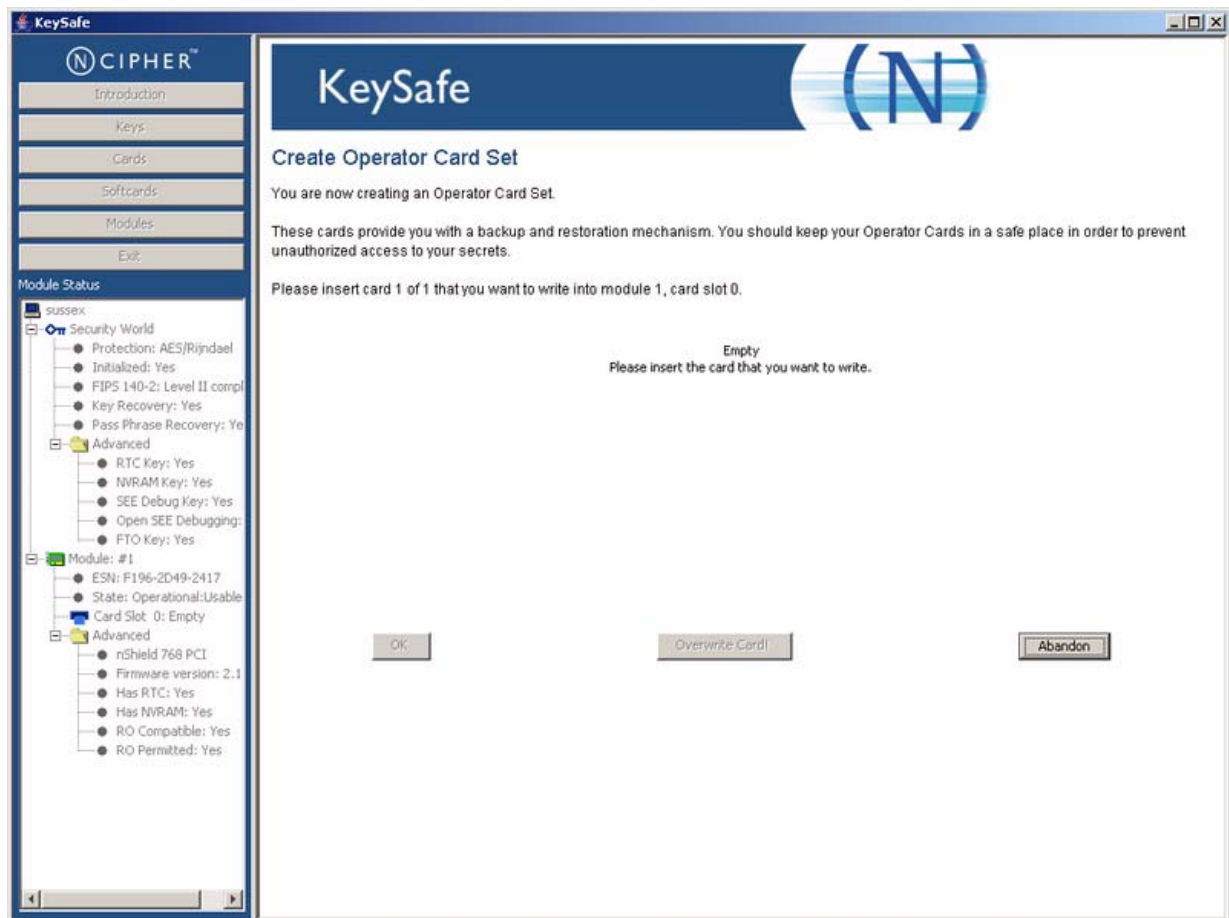


- Click the **Create New OCS** navigation button. KeySafe takes you to the **Create Operator Card Set** panel:



- Enter a name for the card set.
- If you are creating the card set in a FIPS 140-2 level 3 security world, insert an Administrator Card or an existing Operator Card when prompted.
- Choose whether you want pass phrase replacement to be enabled for the OCS.
- Choose whether you want the card set to be used remotely. For more information, see Chapter 8: [Remote Operator Card Sets](#).
- Choose whether you want this OCS to be persistent.
- Enter the total number of Operator Cards (*N*) that you want this OCS to have. This must be a value in the range 1 – 64.
- Choose whether you want this OCS to have a time-out, after which the card set must be inserted again, and enter the value for the time-out length in seconds.

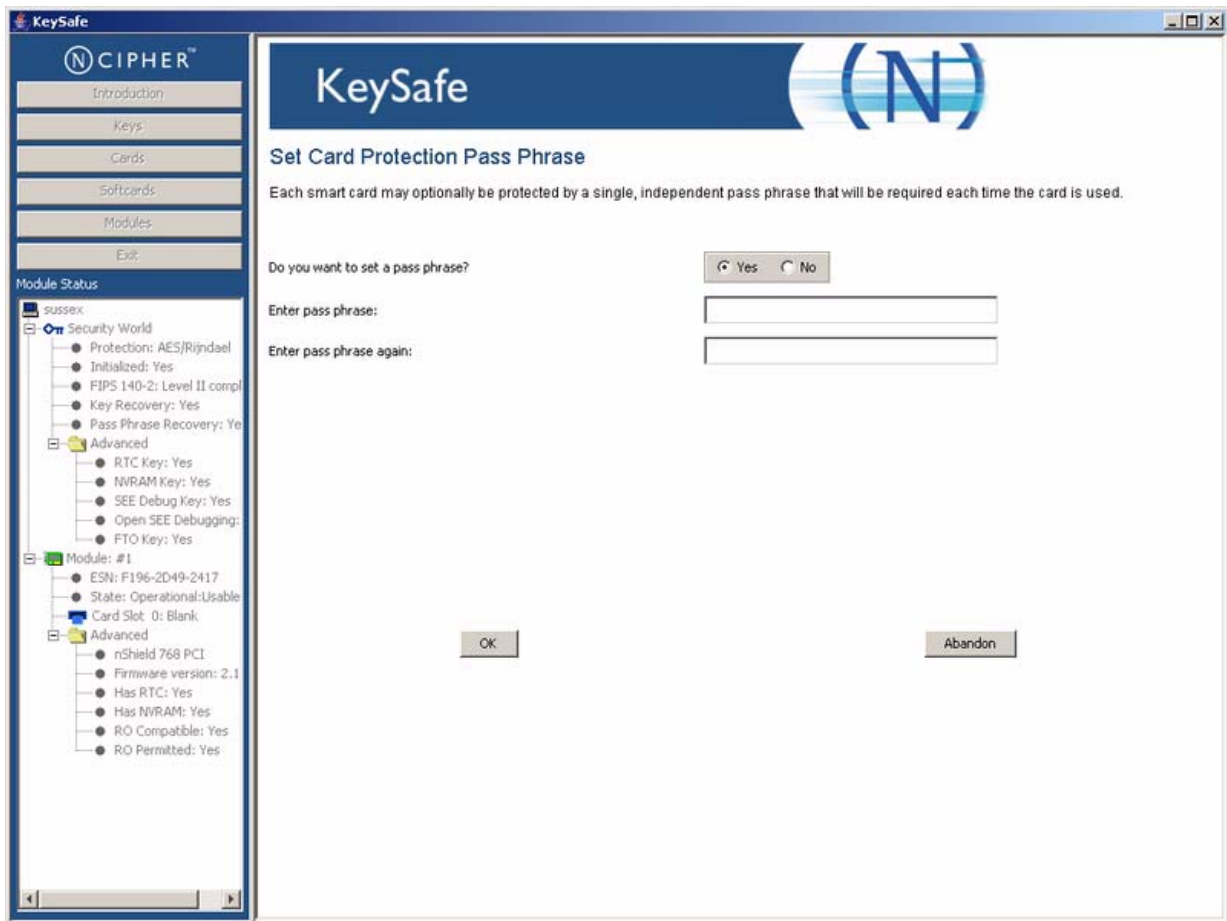
- 11 Enter the number of Operator Cards needed to re-create a key (**K**). **K** must be less than or equal to **N**.
- 12 When you have entered all the details, click the **Create OCS!** command button. KeySafe takes you to a new Create Operator Card Set panel, which prompts you to insert a card to use as the first card in the set:



- 13 Insert a blank card into the appropriate smart card slot. If you insert a card from another OCS or another security world, KeySafe asks whether you want to erase it. If you insert an Administrator Card from the current security world, KeySafe prevents you from accidentally erasing it.

**Note** When creating a card set, KeySafe recognizes cards that already belong to the set before the card set is complete. If you accidentally insert a card to be written again after it has already been written, you receive a warning.

- 14 After you insert a blank card (or have erased an existing card), click the **OK** button to continue the process of creating an OCS. KeySafe takes you to the **Set Card Protection Pass Phrase** panel:



Select whether or not you want to set a pass phrase for the currently inserted card. Each card in a set can have an individual pass phrase, and you can also create a set in which some cards have pass phrases and others do not.

To set a pass phrase for the currently inserted card, enter the same pass phrase in both text fields. A pass phrase can contain any characters you can type except for tabs or carriage returns (because these keys are used to move between data fields).

You can change a pass phrase at any time. If you do not set a pass phrase now, you can use the KeySafe **Change Pass Phrase** option (on the **Examine/Change Card** panel) to add one later. Likewise, if you later decide that you do not need a pass phrase on a card, you can use this option to remove it.

After entering your desired pass phrase (if any) in both text fields, click the **OK** button. Unless you have entered details for the last card in the set, KeySafe returns you to the Create Operator Card Set panel and prompts you to enter the next card in the set to be written.

- 15 After KeySafe has written the details of the last smart card in the set, it displays a dialog indicating that the OCS has been successfully created. Click the **OK** button, and KeySafe returns you to the Create Operator Card Set panel, where you can create another OCS or choose a different operation by clicking one of the menu buttons.

## Creating softcards

You must create a softcard before you create the keys that it is to protect.

A softcard is a file containing a logical token that cannot be loaded without a pass phrase; its logical token must be loaded in order to authorize the loading of any key that is protected by the softcard. Softcard files are stored in the Key Management Data directory and have names of the form **softcard\_hash** (where *hash* is the hash of the logical token share). Softcards belong to the security world in which they are created.

A softcard's pass phrase is set when you generate it, and you can use a single softcard to protect multiple keys. Softcards are persistent; after a softcard is loaded, it remains valid for loading the keys it protects until its **KeyID** is destroyed.

**Note** It is possible to generate multiple softcards with the same name or pass phrase. For this reason, the hash of each softcard is made unique (unrelated to the hash of its pass phrase).

**Note** Softcards are not supported for use with the nCipherKM JCA/JCE CSP in Security Worlds that are compliant with FIPS 140-2 level 3.

**Note** To use softcards with PKCS #11, you must have **CKNFAST\_LOADSHARING** set to a nonzero value. When using pre-loaded softcards or other objects, the PKCS #11 library automatically sets **CKNFAST\_LOADSHARING=1** (load-sharing mode on) unless it has been explicitly set to **0** (load-sharing mode off).



As with OCSs, if debugging is enabled, a softcard's pass phrase hash is available in the debug output (as a parameter to a **ReadShare** command).

You can create softcards from either:

- the command-line (see [Creating a softcard with ppmk](#) on page 125)
- KeySafe (see [Creating softcards with KeySafe](#) on page 125).

## Creating a softcard with ppmk

To create a new softcard using the **ppmk** command-line utility:

- 1 Decide whether you want the new softcard's pass phrase to be replaceable or non-replaceable. To create a softcard with a replaceable pass phrase, run the command:

---

```
ppmk --new --recoverable NAME
```

---

To create a softcard with a non-replaceable pass phrase, run the command:

---

```
ppmk --new --non-recoverable NAME
```

---

In these commands, *NAME* specifies the name of the new softcard to be created.

- 2 If you are working within a FIPS 140-2 level 3 compliant Security World, you must provide authorization to create new softcards. The **ppmk** utility prompts you to insert a card that contains this authorization. Insert any card from the ACS or any . If you insert an Administrator Card from another Security World, **ppmk** displays an error message and prompts you to insert a card with valid authorization.

When **ppmk** has obtained the authorization from a valid card, or if no authorization is required, it prompts you to type a pass phrase.

- 3 When prompted, type a pass phrase for the new softcard, and press **Enter**.

A pass phrase can be of any length and contain any characters that you can type except for tabs or carriage returns (because these keys are used to move between data fields).

- 4 When prompted, type the pass phrase again to confirm it, and press **Enter**.

If the pass phrases do not match, **ppmk** prompts you to input and confirm the pass phrase again.

After you have confirmed the pass phrase, **ppmk** completes creation of the new softcard.

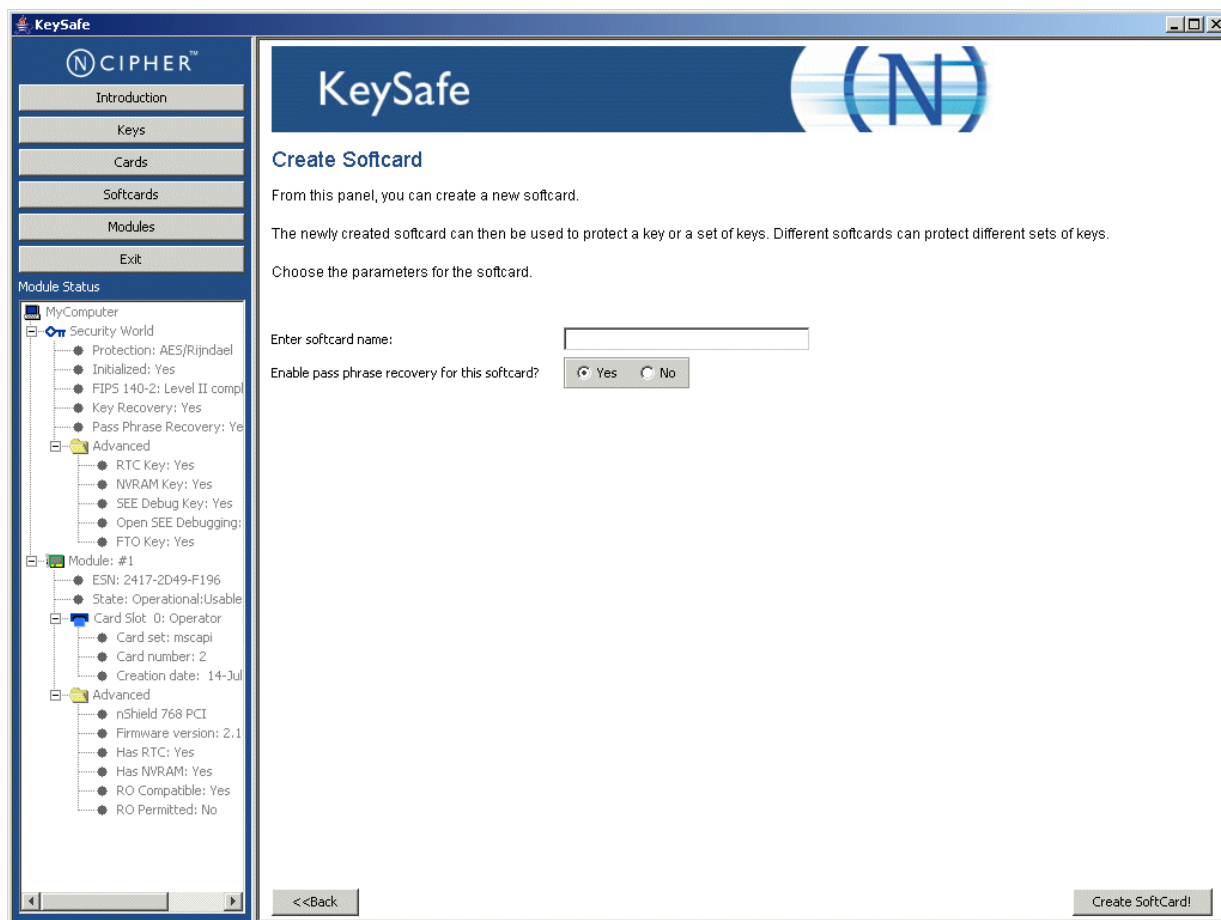
## Creating softcards with KeySafe

To create a softcard with KeySafe:

- 1 Start KeySafe. (For an introduction to KeySafe and information on starting the software, see Appendix A: [Using KeySafe](#).)
- 2 Click the **Softcards** menu button. KeySafe takes you to the **Softcard Operations** panel.

### 3 Click the **Create New Softcard** navigation button.

KeySafe takes you to the **Create Softcard** panel:



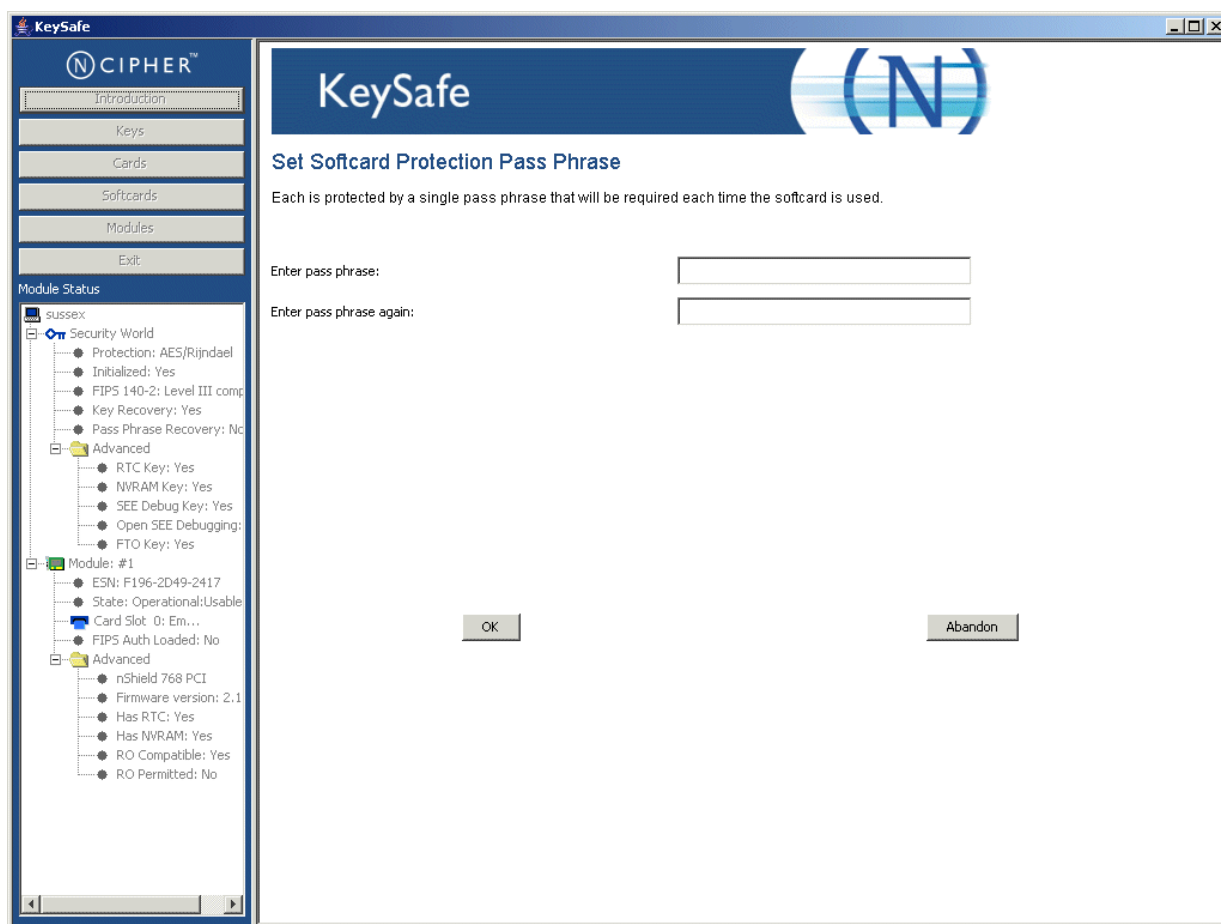
### 4 Choose parameters for the softcard:

- a Enter a name for the softcard.
- b Choose whether you want pass phrase replacement to be enabled for the softcard.

- 5 Click the **Create Softcard!** command button.

If you are creating the softcard in a FIPS 140-2 level 3 security world, insert an Administrator Card or an existing Operator Card when prompted.

KeySafe then takes you to the **Set Softcard Protection Pass Phrase** pane:



- 6 Set a pass phrase for the softcard by entering the same pass phrase in both text fields.

A pass phrase can contain any characters you can type except for tabs or carriage returns (because these keys are used to move between data fields) and can be up to 1024 characters long. You can change a pass phrase at any time.

- 7 After entering your desired pass phrase in both text fields, click the **OK** button.

KeySafe displays a dialog indicating that the softcard has been successfully created.

- 8 Click the **OK** button.

KeySafe returns you to the **Create Softcard** panel, where you can create another softcard or choose a different operation by clicking one of the menu buttons.



## Erasing cards and softcards

Erasing a card or softcard removes all the secret information from the card or softcard and deletes information about the card or softcard from the host.



If you do not erase the smart cards that you have used as Operator Cards before you reinitialize your hardware security device, you must discard them because they cannot be used, erased, or reformatted without the old Security World key.

You can erase Operator Cards using KeySafe or the **createocs** utility. You can also use these methods to erase Administrator Cards other than those in the current Security World's ACS (for example, you could use these methods to erase the remaining Administrator Cards from an incomplete set that has been replaced or Administrator Cards from another Security World).

**Note** None of these tools erases cards from the current Security World's ACS.

If you erase an Operator Card that is the only card in an OCS, information about the card set is deleted. However, if you erase one card from an OCS of multiple cards, you must remove the card information after you have erased the last card.

**Note** You can erase an entire card set at one time with the KeySafe **Remove OCS!** feature. For more information, see [List an Operator Card Set](#) on page 132.

## FIPS 140-2 level 3-compliant Security Worlds

When you attempt to erase cards for a Security World that complies with FIPS 140-2 level 3, you are prompted to insert an Administrator Card or Operator Card from an existing set. You may need to specify to the application the slot you are going to use to insert the card. You need to insert the card only once in a session. You can therefore use one of the cards that you are about to erase.

## Erasing cards with KeySafe

To erase a card using KeySafe use the following procedure:

- 1 Start KeySafe. (For an introduction to KeySafe and information on starting the software, see Appendix A: [Using KeySafe](#).)
- 2 Click the **Cards** menu button. KeySafe takes you to the Card Operations panel.
- 3 Click the **Examine/Change Card** navigation button. KeySafe takes you to the Examine/Change Card panel.
- 4 Insert the card that you want to erase into the reader.



- 5 Click the **Erase Card!** button. You do not need to supply the pass phrase (if there is one) to erase an Operator Card.
- 6 KeySafe asks you to confirm that you want to erase this card. If you are sure that you want to erase it, click the **Yes** button.

**Note** Erasing a card does not erase the keys protected by that card. The keys are still listed on the keys panel but are unusable.

If you erase an Operator Card that is the only card in an OCS, KeySafe deletes information about that card set. However, if you erase one card from an OCS of multiple cards, you must remove the card information after you have erased the last card.

- 7 After erasing a card, KeySafe displays a dialog to confirm that the card has been erased. Click **OK** to continue using KeySafe.

**Note** You can erase an entire card set at one time with the KeySafe **Remove OCS!** feature; see [List an Operator Card Set](#) on page 132.

## Erasing cards from the command line

To erase a card from the command line, run the command:

---

```
createocs -m|--module=MODULE -e|--erase
```

---

This command uses the following options:

| Option                    | Description                                                                                             |
|---------------------------|---------------------------------------------------------------------------------------------------------|
| <b>-m --module=MODULE</b> | These options specify the module number of the module. If you only have one module, <b>MODULE</b> is 1. |
| <b>-e --erase</b>         | These options specify that you want to erase a card (rather than create an OCS).                        |

**Note** If you have more than one card reader and there is more than one card available, **createocs** prompts you to confirm which card you wish to erase. Use **Ctrl-X** to switch between cards.

If you have created a FIPS 140-2 level 3 compliant Security World, you must provide authorization in order to erase or create Operator Cards. You can obtain this authorization from any card in the ACS or from any Operator Card in the current Security World, including cards that are to be erased. After you insert a card containing this authorization, **createocs** prompts you to insert the card to be erased.

## Erasing softcards

Erasing a softcard deletes all information about the softcard from the host. You can erase softcards using KeySafe or with the **ppmk** command-line utility.

### Erasing softcards with KeySafe

To erase softcards with KeySafe:

- 1 Start KeySafe.
- 2 Click the **Softcards** menu button. KeySafe takes you to the Softcard Operations panel.
- 3 Click the **List Softcards** navigation button. KeySafe takes you to the List Softcards panel.
- 4 Select the softcard you want to erase from the list.
- 5 Click the **Remove Softcard!** button.
- 6 KeySafe asks you to confirm that you want to erase this card. Click **Yes** to confirm.
- 7 After erasing a softcard, KeySafe displays a dialog box to confirm that the card has been erased. Click **OK** to continue using KeySafe.

### Erasing softcards with ppmk

To erase a softcard with **ppmk**, open a command window, and give the command:

---

```
ppmk --delete NAME|IDENT
```

---

In this command, you can identify the softcard to be erased either by its name (*NAME*) or by its logical token hash as listed by **nfkminfo** (*IDENT*).

If you are working within a FIPS 140-2 level 3 compliant Security World, you must provide authorization to erase softcards; **ppmk** prompts you to insert a card that contains this authorization. Insert any card from the ACS or any Operator Card from the current Security World.

If you insert an Administrator Card from another Security World or an Operator Card that you have just created, **ppmk** displays an error message and prompts you to insert a card with valid authorization. When **ppmk** has obtained the authorization from a valid card or if no authorization is required, it completes the process of erasing the softcard.

## Viewing cards and softcards

It is often necessary to obtain information from card sets, usually because for security reasons they are left without any identifying markings.

To view details of all the Operator Cards in a Security World or details of an individual Operator Card, you can use KeySafe or the **nfkminfo** command-line utility. To check which pass phrase is associated with a card, you can use the **cardpp** command-line utility.

To list all softcards in a Security World or to show details of an individual softcard, you can use the **ppmk** or **nfkminfo** command-line utilities. To check which pass phrase is associated with a softcard, you can use the **ppmk** command-line utility.

### Viewing card sets with KeySafe

You can use KeySafe to view details of all the Operator Cards in a Security World, details of individual OCSs or details of an individual Operator Card.

### Examining a Card

In order to view information about individual cards with KeySafe, follow these steps:

- 1 Start KeySafe. (For an introduction to KeySafe and information on starting the software, see Appendix A: [Using KeySafe](#).)
- 2 Click the **Cards** menu button. KeySafe takes you to the **Card Operations** panel.
- 3 Click the **Examine/Change Card** navigation button. KeySafe takes you to the **Examine/Change Card** panel.
- 4 Insert a card into the appropriate smart card slot. KeySafe displays information about the smart card currently in the slot. If there is no smart card in the slot, KeySafe displays a message **Card slot empty - please insert the card that you want to examine**.

From the **Examine/Change Card** panel, you can also:

- change a card's pass phrase (if it has one)
- give a pass phrase to a card that does not already have one
- remove a pass phrase from a card that currently has one.

## List an Operator Card Set

In order to view information about whole OCSs with KeySafe, follow these steps:

- 1 Start KeySafe. (For an introduction to KeySafe and information on starting the software, see Appendix A: [Using KeySafe](#).)
- 2 Click the **Cards** menu button. KeySafe takes you to the **Card Operations** panel.
- 3 Click the **List OCS** navigation button. KeySafe takes you to the **List Operator Card Sets** panel, which displays information about all OCSs in the current Security World.

From the **List Operator Card Sets** panel, you can also choose to remove an OCS from the Security World by clicking the **Remove OCS!** button.

## Viewing card sets from the command line

You can use the **nfkminfo** command-line utility to view details of either all the Operator Cards in a Security World or of an individual Operator Card.

To list the OCSs in the current Security World from the command line, open a command window, and give the command:

---

```
nfkminfo --cardset-list
```

---

In this command, **--cardset-list** specifies that you want to list the operator card sets in the current Security World.

**nfkminfo** displays output information similar to the following:

---

```
Cardset summary - 1 cardsets: (in timeout, P=persistent, N=not)
Operator logical token hash k/n timeout name
hash 1/1 none-N name
```

---

To list information for a specific card, use the command:

---

```
nfkminfo TOKENHASH
```

---

In this command, **TOKENHASH** is the **Operator logical token hash** of the card (as listed when the command **nfkminfo --cardset-list** is run).

This command displays output information similar to the following:

---

|            |                                          |
|------------|------------------------------------------|
| name       | "name"                                   |
| k-out-of-n | 1/1                                      |
| flags      | NotPersistent                            |
| timeout    | none                                     |
| card names | " "                                      |
| hkltu      | 794ada39038fa8c4e9ea46a24136bbb2b8b337f2 |

---

Note In the current release, not all software can give names to individual cards.

## Viewing softcards

To view softcards, use KeySafe or the command line. The command line provides several options for viewing softcard information.

### Viewing softcards with KeySafe

To view a softcard with KeySafe, follow these steps:

- 1 Start KeySafe.
- 2 Click the **Softcards** menu button. KeySafe takes you to the Softcard Operations panel.
- 3 Click the **List Softcards** navigation button. KeySafe takes you to the List Softcards panel, which displays information about all softcards in the current Security World.

From the List Softcards panel, you can also choose to remove a softcard from the Security World. For more information about this procedure, see [Erasing cards and softcards](#) on page 128.

### Viewing softcards with nfkminfo

To list the softcards in the current Security World using the **nfkminfo** command-line utility, give the command:

---

```
nfkminfo --softcard-list
```

---

In this command **--softcard-list** specifies that you want to list the softcards in the current Security World.

To show information for a specific softcard using the **nfkminfo** command-line utility, give the command:

---

```
nfkminfo --softcard-list IDENT
```

---

In this command **IDENT** is the softcard's logical token hash (as given by running the command **nfkminfo --softcard-list**). This command displays output information similar to the following:

---

```
SoftCard
 name "mysoftcard"
 hkltu 7fb95888ea2850d4e3ffcc8f0c22100937344308
Keys protected by softcard 7fb95888ea2850d4e3ffcc8f0c22100937344308:
 AppName simple Ident mykey
 AppName simple Ident myotherkey
```

---

## Viewing softcards with ppmk

To list the softcards in the current Security World using the **ppmk** command-line utility, use the command:

---

```
ppmk --list
```

---

In this command **--list** specifies that you want to list the softcards in the current Security World.

In order to view the details of a particular softcard using the **ppmk** command-line utility, give the command:

---

```
ppmk --info NAME|IDENT
```

---

In this command, you can identify the softcard whose details you want to view either by its name (**NAME**) or by its logical token hash (as given by running the command **nfkminfo --softcard-list**).

## Verifying the pass phrase of a card or softcard

### Verifying the pass phrase of a card with cardpp

To verify the pass phrase associated with a card using the **cardpp** command-line utility, use the command:

---

```
cardpp --check [-m|--module=MODULE]
```

---

This command uses the following options:

| Option                 | Description                                                                                                                                                                               |
|------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>--check</b>         | This option tells <b>cardpp</b> to check the pass phrase.                                                                                                                                 |
| <b>--module=MODULE</b> | This option specifies the number of the module to use. If you only have one module, <b>MODULE</b> is 1. If you do not specify a module number, <b>cardpp</b> uses all modules by default. |

The **cardpp** utility polls all available slots; if there is no card inserted, it prompts you to insert one. If the card belongs to this Security World, **cardpp** either tells you if no pass phrase is set or prompts you to enter the pass phrase and checks to see if it is correct.

## Verifying the pass phrase of a softcard with ppmk

In order to verify the pass phrase of a particular softcard, open a command window, and give the command:

---

```
ppmk --check NAME|IDENT
```

---

In this command, you can identify the softcard whose pass phrase you want to verify either by its name (**NAME**) or by its logical token hash (as given by running the command **nfkminfo --softcard-list**).

**ppmk** prompts you to enter the pass phrase and then tells you whether the pass phrase you entered is correct for the specified softcard.

## Changing card and softcard pass phrases

Each softcard or card of a card set can have its own individual pass phrase: you can even have a card set in which some cards have pass phrases and others do not, and you can have distinct softcards that nevertheless use the same pass phrase. A pass phrase can be of any length and can contain any characters that you can type.

Normally, in order to change the pass phrase of a card or softcard, you need the card or softcard and the existing pass phrase. Known card pass phrases can be changed using KeySafe or the **cardpp** command-line utility; softcard pass phrases can be changed using KeySafe or the **ppmk** command-line utility. You can also add a pass phrase to a card or softcard that currently does not have one or remove a pass phrase from a card that does currently have one.

If you generated your Security World with the pass phrase replacement option, you can also replace the pass phrase of a card or softcard even if you do not know the existing pass phrase. Such a pass phrase replacement operation requires authorization from the ACS.

## Changing known pass phrases

### Changing known pass phrases with KeySafe

To change a card pass phrase, you need the card and the old pass phrase.

Each card in a set can have its own individual pass phrase: you can even have a set in which some cards have pass phrases and others do not. A pass phrase can be of any length and can contain any characters that you can type.

**Note** There is no absolute limit on the length of pass phrases. However, some applications may not accept pass phrases longer than 255 characters. Likewise, the Security World does not impose restrictions on which characters you can use, although some applications may not accept certain characters.

### Changing known card pass phrases with KeySafe

To change a known pass phrase for an Operator Card using KeySafe:

- 1 Start KeySafe. (For an introduction to KeySafe and information on starting the software, see Appendix A: [Using KeySafe.](#))
- 2 Click **Cards**. The Examine/Change panel is displayed.
- 3 Click **Change pass phrase**. The Set Card Protection pass phrase panel is displayed.
- 4 Enter the old pass phrase, and click the **OK** button.
- 5 A screen is displayed asking **Do you want to set a pass phrase?**. Select **Yes**.
- 6 Enter your new pass phrase, and enter it again in the second box as confirmation of the change.
- 7 Click **OK**.

### Changing a known softcard pass phrase with KeySafe

To change a known pass phrase for a softcard using KeySafe:

- 1 Start KeySafe. (For an introduction to KeySafe and information on starting the software, see Appendix A: [Using KeySafe.](#))
- 2 Click the **Softcards** menu button. KeySafe takes you to the Softcard Operations panel.
- 3 Click the **Change Pass phrase** navigation button. KeySafe takes you to the **Change/Recover Softcard Pass phrase** panel.



- 4 Select the softcard whose pass phrase you want to change, and click the **Change Pass phrase** button. KeySafe takes you to the **Get Softcard Protection** pass phrase panel.
- 5 Enter the old pass phrase, and click the **OK** button.

KeySafe either displays an error dialog (if the pass phrase is not correct) or takes you to the **Set Softcard Protection** pass phrase panel.

- 6 Enter your new pass phrase, and enter it again in the second field to confirm the pass phrase is correct.
- 7 Click the **OK** button.

After changing a pass phrase, KeySafe displays a dialog to confirm that the pass phrase has been successfully changes.

- 8 Click the **OK** button to continue using KeySafe.

## Changing known pass phrases with cardpp

Each card in a card set can have its own individual pass phrase. You can even have a set in which some cards have pass phrases and others do not. A pass phrase can be of any length and can contain any characters that you can type.

To change a known card's pass phrase with the **cardpp** command-line utility, take the following steps:

- 1 Run the **cardpp** utility using the command:

---

```
cardpp --change [-m|--module=MODULE]
```

---

If you only have one module, **MODULE** is 1. If you do not specify a module number, **cardpp** uses all modules by default.

- 2 If prompted, insert the card whose pass phrase you want to change. (If there is a card already in the slot, you are not prompted.)
- 3 If prompted, enter the existing pass phrase for the card (If the card has no current pass phrase you are not prompted.) If you enter the pass phrase correctly, **cardpp** prompts you to enter the new pass phrase.
- 4 Enter a new pass phrase, and then enter it again to confirm it.

After you have confirmed the new pass phrase, **cardpp** changes the card's pass phrase.

## Changing known softcard pass phrases with ppmk

To change a known softcard's pass phrase when you know the pass phrase, follow these steps:

- 1 Give the following command:

---

```
ppmk --change NAME|IDENT
```

---

In this command, you can identify the softcard whose pass phrase you want to change either by its name (*NAME*) or by its logical token hash as listed by **nfkminfo** (*IDENT*).

**ppmk** prompts you to enter the old pass phrase.

- 2 Type the old pass phrase, and press **Enter**. If you enter the old pass phrase correctly, **ppmk** prompts you to enter the new pass phrase.
- 3 Type the old pass phrase, and press **Enter**. Type the new pass phrase again, and press **Enter** to confirm it.

After you have confirmed the new pass phrase, **ppmk** then changes the softcard's pass phrase.

## Changing unknown or lost pass phrases

### Changing unknown card pass phrases with cardpp

If you generated your Security World with the pass phrase replacement option, you can change the pass phrase of a card even if you do not know its existing pass phrase. Such a pass phrase replacement operation requires authorization from the ACS.

To change an unknown card pass phrase with the **cardpp** command-line utility:

- Run a command of the form:

---

```
cardpp --recover [--module=MODULE]
```

---

In this command, *MODULE* specifies the number of the hardware security device to use. If you only have one hardware security device, *MODULE* is 1. If you do not specify a number, **cardpp** uses all hardware security devices by default.

- As prompted, insert the appropriate number of cards from the ACS required to authorize pass phrase replacement.

- When prompted, insert the Operator Card whose pass phrase you want to replace. To replace its pass phrase:
  - a When prompted, type the new pass phrase, and then press `[Enter]`.
  - b When prompted, type the new pass phrase again to confirm it, and then press `[Enter]`.

**cardpp** sets the new pass phrase, and then prompts you for another Operator Card.
- Repeat the process in the previous step to change the pass phrase on further cards, or press `[Q]` to quit.



Only insert Administrator Cards into a hardware security device that is connected to a trusted server.

## Replacing unknown pass phrases with ppmk

If you generated your Security World with the pass phrase replacement option, you can change the pass phrase of a softcard even if you do not know its existing pass phrase. Such a pass phrase replacement operation requires authorization from the ACS.

To change an unknown softcard pass phrase with the **ppmk** command-line utility:

- 1 Run a command of the form:

---

```
preload --admin=p ppmk --recover NAME|IDENT
```

---

In this command, you can identify the softcard by its *NAME*) or by its *IDENT* (its logical token hash as shown in output from the **nfkminfo** command-line utility).

- 2 As prompted, insert the appropriate number of cards from the ACS required to authorize pass phrase replacement.
- 3 When prompted, type the new pass phrase, and then press `[Enter]`.
- 4 When prompted, type the new pass phrase again to confirm it, and then press `[Enter]`.

If the pass phrases do not match, **ppmk** prompts you to input and confirm the pass phrase again.

After you successfully confirm the new pass phrase, **ppmk** finishes configuring the softcard to use the new pass phrase.



Only insert Administrator Cards into a hardware security device that is connected to a trusted server.

## Replacing Operator Card Sets

**Note** Replacing an OCS requires authorization from the ACS of the security-world to which it belongs. You cannot replace an OCS unless you have the required number of cards from the appropriate ACS.

If you have lost a card from a card set, you can replace the card set using the **rocs** utility or the KeySafe Replace Operator Card Set option (reached from the **Card Operations** panel).

We recommend that after you have replaced an OCS, you then erase the remaining cards in the old card set and remove the old card set from the Security World. For more information, see [Erasing cards and softcards](#) on page 128.

Deleting the information about an OCS from the host does not remove the data for keys protected by that card set. On the KeySafe **List Keys** panel (reachable from the **Key Operations** panel), such keys are listed as being protected by **Deleted Card Set**.

To prevent you from losing access to your keys if the smart card you are using as the Operator Card is lost or damaged, Thales supplies several utilities that can recover the keys protected by the lost Operator Card to another token:

- KeySafe includes an option to replace OCSs on the Card Operations panel (click the **Replace OCS** navigation button).
- The **rocs** command-line utility provides an interactive method or a command-line only method to replace OCSs.

Replacing one OCS with another OCS also transfers the keys protected by the first OCS to the protection of the new OCS.

When you replace an OCS or softcard and recover its keys to a different OCS or softcard, the key material is not changed by the process. The process deletes the original host data (that is, the encrypted version of the key or keys and the smart card or softcard data file) and replaces this data with host data protected by the new OCS or softcard.

To replace an OCS or softcard, you must:

- have enabled OCS and softcard replacement when you created the Security World

**Note** If you did not enable OCS and softcard replacement, or if you created the Security World with an early version of the **pkcs-init** command-line utility that did not support OCS and softcard replacement, you cannot recover keys from lost or damaged smart cards or softcards.

- have created the original OCS using **createocs**, **createocs-simple**, KeySafe, or the nCipher PKCS #11 library version 1.6 or later



If you initialized the token using **ckinittoken** from the nCipher PKCS #11 library version 1.5 or earlier, you must contact Support to arrange for them to convert the token to the new format while you still possess a valid card.

- have a sufficient number of cards from the ACS to authorize recovery and replacement

**Note** All recovery and replacement operations require authorization from the ACS. If any of the smart cards in the ACS are lost or damaged, immediately replace the entire ACS.

- have initialized a second OCS using **createocs**, **createocs-simple**, KeySafe, or the nCipher PKCS #11 library version 1.6 or later.

**Note** The new OCS need not have the same **K/N** policy as the old set.

If you are sharing the Security World across several host computers, you must ensure that the changes to the host data are propagated to all your computers. One way to achieve this is to use client cooperation. For more information, see [Setting up client cooperation](#) on page 51.

## Replacing OCSs with KeySafe

In order to replace an OCS, you must have another OCS onto which to copy the first set's data. If you do not already have an existing second OCS, you must create a new one. For more information, see [Creating Operator Card Sets \(OCSs\)](#) on page 116.

When you have a second OCS ready, follow these steps in order to replace the first OCS:

- 1 Start KeySafe. (For an introduction to KeySafe and information on starting the software, see Appendix A: [Using KeySafe](#).)
- 2 Click the **Cards** menu button. KeySafe takes you to the **Card Operations** panel.

- Click the **Replace OCS** navigation button, and KeySafe takes you to the **Replace Operator Card Set** panel:



This panel lists existing OCSs in tabular form. For each card set it displays:

| Attribute                       | Description                                                                     |
|---------------------------------|---------------------------------------------------------------------------------|
| <b>Name</b>                     | The name of the card set.                                                       |
| <b>Required (K)</b>             | The number of cards needed to re-create a key.                                  |
| <b>Total (N)</b>                | The total number of cards in the set.                                           |
| <b>Persistent</b>               | Indicates whether or not the card set is persistent.                            |
| <b>Recoverable Key Count</b>    | The number of private keys protected by this card set that are recoverable.     |
| <b>Nonrecoverable Key Count</b> | The number of private keys protected by this card set that are not recoverable. |

You can click and drag with your mouse in order to resize the column widths and to rearrange the column order of this table. Clicking a column heading sorts the rows in ascending order based on that column heading.

- Select an OCS that you want to replace by clicking its list entry.

- 5 Click the **Replace OCS!** command button.

Note If an OCS does not have any recoverable keys, it cannot be replaced.

- 6 KeySafe takes you to the **Load Administrator Card Set** panel, where it prompts you to insert cards from the ACS in order to authorize the action. Each time you insert an Administrator Card into the smart card of the hardware security module slot, you must click the **OK** button to load the card.

Note Only insert your ACS into a module that is connected to a trusted server.

- 7 When you have loaded enough cards from the ACS to authorize the procedure, KeySafe takes you to the **Load Operator Card Set** panel, where it prompts you to insert the OCS that is to protect the recoverable keys (this is the OCS onto which you are copying data from the OCS you are replacing). Each time you insert a card from the new OCS into the smart card slot of the hardware security device, you must click the **OK** button.

When you have loaded enough cards from the new OCS, KeySafe creates new working versions of the recoverable keys that are protected by this card set.

KeySafe deletes the original host data for all recovered keys and replaces this data with host data that is protected by the new OCS. If there are no nonrecoverable keys protected by the card set, KeySafe also removes the old card set from the Security World. However, if the OCS has nonrecoverable keys, the host data for the original card set and for the nonrecoverable keys is not deleted. These keys can only be accessed with the original OCS. If you want to delete these files, use the **Remove OCS** option.

- 8 When the process is complete, KeySafe displays a dialog indicating that the OCS has been successfully replaced. Click the **OK** button. KeySafe returns you the Replace Operator Card Set panel, where you may replace another OCS or choose a different operation.

## Replacing OCSs or softcards with rocs

You can use the **rocs** command-line utility interactively, or you can supply all the parameters on the command line.

### Using rocs interactively

To use the **rocs** command-line utility interactively, run it without any parameters:

---

```
rocs
```

---

**rocs** displays the following prompt:

---

```
'rocs' key recovery tool
Useful commands: 'help', 'help intro', 'quit'.
rocs >
```

---

In order to use **rocs** to replace an OCS or recover keys to a softcard, take the following steps:

- 1 You must select a hardware security device to use by using the **module** command, which is described in the section [module number](#) on page 148.
- 2 List the OCSs and softcards in the current Security World by using the **list cardsets** command, which is described in the section [list cardsets](#) on page 146.
- 3 Select the OCS or softcard to which you want to transfer the keys by using the **target** command, which is described in the section [target cardset-spec](#) on page 149.

**Note** Keys protected by an OCS can only be recovered to another OCS, and not to a softcard. Likewise, softcard-protected keys can only be recovered to another softcard, and not to an OCS.

- 4 List the keys in the current Security World using the **list keys** command, which is described in the section [list keys](#) on page 147.
- 5 Select the keys that are to be recovered (from a different OCS or softcard than the one you selected for key transfer) by using the **mark** command, which is described in the section [mark key-spec](#) on page 147.
- 6 If you have selected any keys by mistake, deselect them by using the **unmark** command, which is described in the section [unmark key-spec](#) on page 149.
- 7 After you have selected the keys that are to be recovered, transfer these keys by using the **recover** command, which is described in the section [recover](#) on page 148.

**rocs** prompts you to insert a card from the ACS.

- 8 Insert a card from the ACS.

**rocs** prompts you for the pass phrase for this card. This action is repeated until you have loaded the required number of cards from the ACS.

If you do not have the required number of cards from the ACS, press **Q** and then **Enter**. The **rocs** utility returns you to the **rocs >** prompt without processing any keys.



Only insert Administrator Cards into a hardware security device that is connected to a trusted server.



9 If you are recovering keys to an OCS:

- a **rocs** prompts you to insert a card from the first OCS that you have selected as the target. OCSs are processed in ascending numerical order as listed by the **list cardsets** command.
- b Insert a card from this OCS.
- c **rocs** prompts you for the pass phrase for this card. This action is repeated until you have loaded the required number of cards from the OCS.

If you are recovering keys to a softcard, **rocs** prompts you for the pass phrase for the softcard that you have selected as the target.

If you decide that you do not want to transfer the keys to the selected card set or softcard, press **Q** and then **Enter** (to quit. **rocs** returns you to the **rocs >** prompt and does not process any further OCSs or softcards.

When you have loaded the target softcard or the required number of cards from the target OCS, **rocs** transfers the selected keys to the target OCS or softcard.

If you have selected other target OCSs or softcards, **rocs** prompts for a card from the next OCS.

10 Repeat step 9 for each selected target.

- 11 If you have transferred the correct keys, write the key blobs to disk by using the **save** function (described in the section [save key-spec](#) on page 148). If you have transferred a key by mistake, you can restore it to its original protection by using the **revert** command (described in the section [revert key-spec](#) on page 148).

At the **rocs** prompt, you can use the following commands:

- **help** *topic*
- **help** **intro**
- **list** **cardsets**
- **list** **keys**
- **mark** *key-spec*
- **module** *number*
- **quit**
- **recover**
- **rescan**

- **revert** *key-spec*
- **save** *key-spec*
- **status**
- **target** *cardset-spec*
- **unmark** *key-spec*

**Note** You can specify a command by typing enough characters to identify the command uniquely. For example, for the **status** command, you can type **st** and then press **Enter**.

## help

With no arguments specified, **help** shows a list of available commands with brief usage messages and a list of other help topics. With an argument, **help** shows detailed help information about a given topic.

**help intro** displays a brief step-by-step guide to using **rocs**.

## list cardsets

This command lists the OCSs and softcards in the current Security World. For example:

| No. | Name  | Keys (recov) | Sharing                   |
|-----|-------|--------------|---------------------------|
| 1   | test  | 6 (6)        | 3 of 5; 20 minute timeout |
| 2   | test2 | 3 (2)        | 2 of 3                    |
| 3   | test3 | 1 (1)        | 1 of 1; persistent        |

In this output:

| Output                    | Description                                                                                           |
|---------------------------|-------------------------------------------------------------------------------------------------------|
| <b>No.</b>                | The card set or softcard number, which you can use to identify this card set in <b>rocs</b> commands. |
| <b>Name</b>               | The OCS or softcard name.                                                                             |
| <b>Keys</b>               | The number of keys protected by this OCS or softcard.                                                 |
| <b>(recov)</b>            | The number of keys protected by this OCS or softcard.                                                 |
| <b>Sharing</b>            | The <b>K</b> of <b>N</b> parameters for this OCS.                                                     |
| <b>persistent</b>         | The OCS is persistent and does not have a time-out set.                                               |
| <b>### minute timeout</b> | The OCS is persistent and has a time-out set.                                                         |

## list keys

This command lists the keys in the current Security World, as in the following example:

| No. | Name                     | App    | Protected by             |
|-----|--------------------------|--------|--------------------------|
| 1   | rsa-test                 | hwcrhk | module                   |
| 2   | Id: uc63e0ca3cb032d71c1c | pkcs11 | test2                    |
| R 3 | Server-Cert              | pkcs11 | test --> test2           |
| 4   | Id: uc63e0ca3cb032d71c1c | pkcs11 | test --> test3           |
| 5   | Server-Cert              | pkcs11 | module (test ---> fred2) |

In this output:

| Output              | Description                                                                  |
|---------------------|------------------------------------------------------------------------------|
| <b>No.</b>          | The key number, which you can use in <b>mark</b> and <b>unmark</b> commands. |
| <b>Name</b>         | The key name.                                                                |
| <b>App</b>          | The application with which the key is associated.                            |
| <b>Protected by</b> | This indicates the protection method (see table below).                      |

In this output, the protection methods include:

| Method                            | Description                                                                                                       |
|-----------------------------------|-------------------------------------------------------------------------------------------------------------------|
| module                            | Key protected by the Security World.                                                                              |
| <i>name</i>                       | Key protected by the named OCS or softcard.                                                                       |
| <i>name-&gt;name2</i>             | Key protected by the OCS or softcard <i>name1</i> marked for recovery to OCS or softcard <i>name2</i> .           |
| module [name]                     | PKCS #11 public object. These are protected by the Security World but associated with a specific OCS or softcard. |
| module [ <i>name--&gt;name2</i> ] | PKCS #11 public object marked for recovery.                                                                       |

## mark *key-spec*

This command marks the listed keys that are to be recovered to the target OCS or softcard. You can mark one or more keys by number, **ident**, OCS or softcard, or hash. For more information, see [Specifying keys](#) on page 151.

To mark more than one key at a time, ensure that each **key-spec** is separated from the other by spaces, as in the following example:

```
mark key-spec1 key-spec2 key-spec3
```

If you have not selected a target OCS or softcard, or if **rocs** cannot parse the **key-spec**, then **rocs** displays an error message.

You can mark and remark the keys to be recovered to various target OCSs or softcards. Remarking a key displaces the first target in favor of the second target.

**Note** Keys protected by an OCS can only be recovered to another OCS, and not to a softcard. Likewise, softcard-protected keys can only be recovered to another softcard, and not to an OCS.

*module number*

This command selects the hardware security device to be used. The module number must correspond to a hardware security device in the current Security World. If the hardware security device does not exist, is not in the Security World, or is otherwise unusable, then **rocs** displays an error message and does not change to the selected module.

*quit*

This command allows you to leave **rocs**. If you attempt to **quit** when you have recovered keys but have not saved them, **rocs** displays a warning.

*recover*

This command transfers the marked keys to their target OCSs or softcards. This operation is not permanent until you save these keys by using the **save** command.

*rescan*

This command updates the card set and key information.

*revert key-spec*

This command returns keys that have been recovered, but not saved, to being protected by the original protection method. If the selected keys have not been recovered, **rocs** displays an error message.

*save key-spec*

This command writes the new key blobs to disk. If you specify a *key-spec*, only those keys are saved. Otherwise, all recovered keys are saved.

*status*

This command lists the currently selected hardware security device and target OCS or softcard.

target *cardset-spec*

This command selects a given OCS or softcard as the target. You can specify the card set or softcard name, the number returned by **list cardsets**, or the hash.

unmark *key-spec*

This command unmarks the listed keys. Unmarked keys are not recovered.

## Using rocs from the command line

You can select all the options for **rocs** on the command line by running a command of the form:

---

```
rocs -m|--module=MODULE [-t|--target=CARDSET-SPEC] [-k|--keys=KEYS-SPEC] [-c|--cardset=CARDSET-SPEC] [-i|--interactive]
```

---

In this command:

| Option                            | Description                                                                                                                                                |
|-----------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-m, --module=MODULE</b>        | These options specify the number of the hardware security device to use.                                                                                   |
| <b>-t, --target=CARDSET-SPEC</b>  | These options specify the OCS or softcard to be used to protect the keys. For more information, see <a href="#">Specifying card sets</a> on page 150.      |
| <b>-k, --keys=KEYS-SPEC</b>       | These options select the keys to be recovered. For more information, see <a href="#">Specifying keys</a> on page 151.                                      |
| <b>-c, --cardset=CARDSET-SPEC</b> | These options select all keys that are protected by the given OCS or softcard. For more information, see <a href="#">Specifying card sets</a> on page 150. |
| <b>-i, --interactive</b>          | These options force <b>rocs</b> to start interactively even if you have already selected keys.                                                             |

You must specify the target before you specify keys.

You can use multiple **--keys=KEYS-SPEC** and **--cardset=CARDSET-SPEC** options, if necessary.

You can specify multiple targets on one command line by including separate **--keys=KEYS-SPEC** or **--cardset=CARDSET-SPEC** options for each target. If a key is defined by **--keys=KEYS-SPEC** or **--cardset=CARDSET-SPEC** options for more than one target, it is transferred to the last target for which it is defined.

If you have selected a hardware security device, a target OCS or softcard, and keys to recover but have not specified the **--interactive** option, **rocs** automatically recovers the keys. **rocs** prompts you for the ACS and OCS or softcard. For more information, see [Using rocs interactively](#) on page 143.

**Note** If you use **rocs** from the command line, all keys are recovered and saved automatically. You cannot revert the keys unless you still have cards from the original OCS.

If you do not specify the target and keys to recover, or if you specify the **--interactive** option, **rocs** starts in interactive mode with the selections you have made. You can then use further **rocs** commands to modify your selection before using the **recover** and **save** commands to transfer the keys.

## Specifying card sets

The value of **CARDSET-SPEC** identifies one or more OCSs or softcards. It may have any of the following forms:

| Value                                 | Description                                                                                                                                                                     |
|---------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>[number] <i>cardset-number</i></b> | A value of this form selects the OCS or softcard with the given number from the list produced by the <b>list cardsets</b> command.                                              |
| <b>[name] <i>cardset-name</i></b>     | A value of this form selects card sets or softcards by their names (the card set or softcard name may be a wildcard pattern in order to select all matching OCSs or softcards). |
| <b>hash <i>cardset-hash</i></b>       | A value of this form selects the OCS or softcard with the given hash.                                                                                                           |

In order to specify multiple OCSs or softcards, include several **CARDSET-SPECs** on the command line.

**Note** Keys protected by an OCS can only be recovered to another OCS, and not to a softcard. Likewise, softcard-protected keys can only be recovered to another softcard, and not to an OCS.

## Specifying keys

The **--keys=KEYS-SPEC** option identifies one or more keys. It may have any of the following forms:

| Value                                | Description                                                                                                                                                                                                                                                                                                                                 |
|--------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b><i>mark key-number</i></b>        | <p>A value of this form selects the key with the given number from the list produced by the list keys command. Examples of usage are:</p> <pre>rocs -t target_OCS -k key_number</pre> <p>and</p> <pre>rocs -t target_OCS -k "mark 56"</pre>                                                                                                 |
| <b><i>appname:keyident</i></b>       | <p>A value of this form selects keys by their internal application name and <b>ident</b>. You must supply at least one of <b>appname</b> or <b>keyident</b>, but you can use wildcard patterns for either or both in order to select all matching keys. An example of usage is:</p> <pre>rocs -t target_OCS --keys="simple:simplekey"</pre> |
| <b><i>hash keyhash</i></b>           | <p>A value of this form selects the key with the given key hash. An example of usage is:</p> <pre>rocs -t target_OCS --keys="hash e364[...]"</pre>                                                                                                                                                                                          |
| <b><i>--cardset cardset-spec</i></b> | A value of this form selects all keys protected by a given card set.                                                                                                                                                                                                                                                                        |

## Replacing the Administrator Card Set

Replacing the ACS requires a quorum of cards from the current ACS ( $K/N$ ) to perform the following sequence of tasks:

- 1 loading the secret information that is to be used to protect the archived copy of the Security World key.
- 2 creating a new secret that is to be shared between a new set of cards
- 3 creating a new archive that is to be protected by this secret.

If you discover that one of the cards in the current ACS has been damaged or lost, use the command-line utility **racs** or the KeySafe **Replace Administrator Card Set** option to create a new set immediately. If further cards are damaged, you may not be able to re-create your Security World.



We recommend that you erase your old Administrator Cards as soon as you have created the new ACS. An attacker with the old ACS and a copy of the old host data could still re-create all your keys. With a copy of a current backup, they could even access keys that were created after you replaced the ACS.

**Note** Before you start to replace an ACS, you must ensure that you have enough blank cards to create a complete new ACS. If you start the procedure without enough cards, you will have to cancel the procedure part way through.

## Replacing an ACS with KeySafe

When you have enough cards to create a complete new ACS ready and a quorum of the ACS you want to replace, follow these steps:

- 1 Start KeySafe. (For an introduction to KeySafe and information on starting the software, see Appendix A: [Using KeySafe](#).)
- 2 Click the **Cards** menu button. KeySafe takes you to the Card Operations panel.
- 3 Click the **Replace ACS** navigation button, and KeySafe takes you to the Replace Administrator Card Set panel.
- 4 If you are sure that you want to replace the ACS, click the **Replace ACS!** command button
- 5 KeySafe takes you to the **Load Administrator Card Set** panel, where it prompts you to insert cards from the ACS in order to authorize the action. Each time you insert an Administrator Card into the module's smart card slot, you must click the **OK** button to load the card.

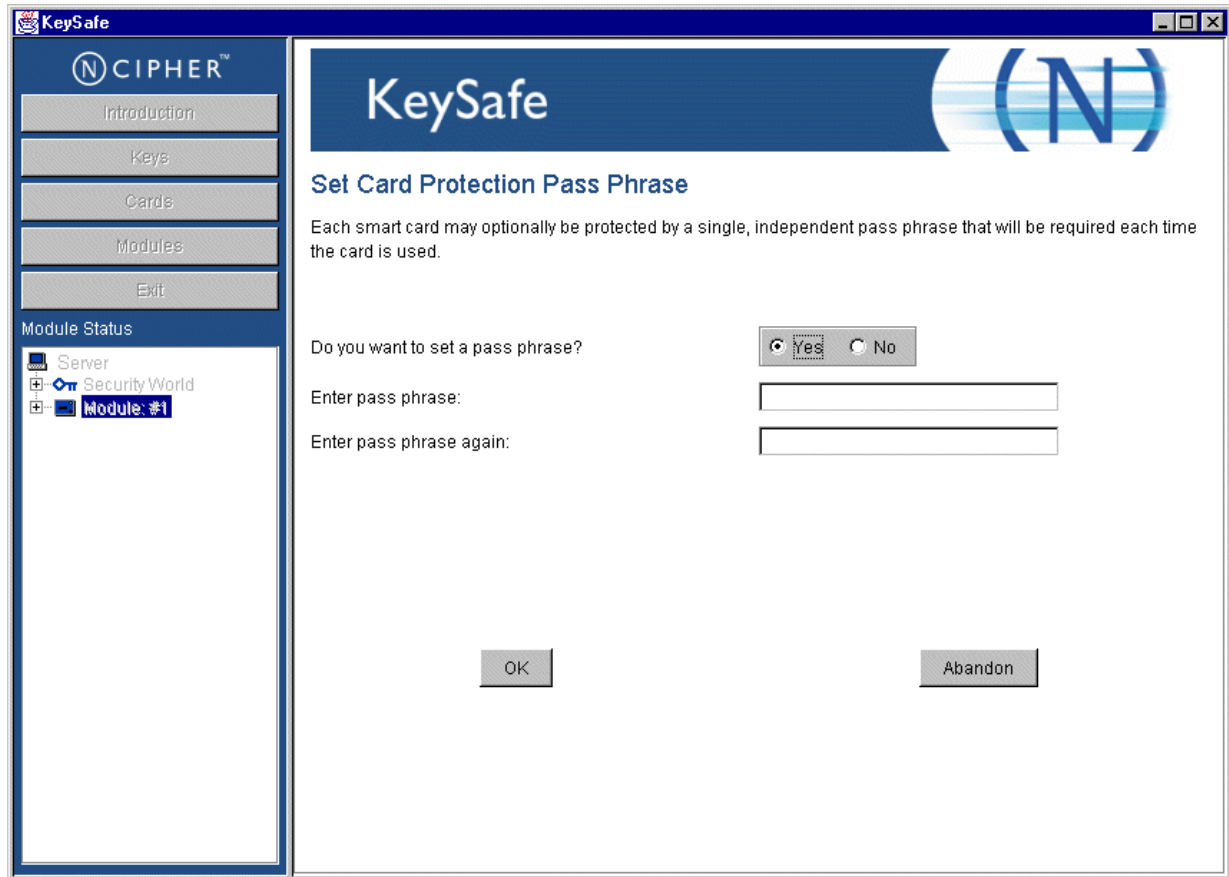
**Note** Only insert cards from your ACS into a module that is connected to a trusted server.

- 6 When you have loaded enough Administrator Cards to authorize the action, KeySafe takes you to the Create Administrator Card Set panel, where it prompts you to insert the cards that are to form the ACS. These must be blank cards or cards that KeySafe can erase. KeySafe will not let you use cards from the existing ACS. If you do not have enough cards to form a complete new ACS, cancel the operation now.

**Note** When creating a card set, KeySafe recognizes cards that belongs to the set even before the card set is complete. If you accidentally insert a card to be written again after it has already been written, KeySafe displays a warning.



- 7 When you insert a blank card, KeySafe takes you to the Set Card Protection Pass Phrase panel:



- 8 If you want to set a pass phrase for this Administrator Card:
- Select the **Yes** option.
  - Enter the same pass phrase in both text fields.
  - Click the **OK** button.

KeySafe then prompts you for the next card (if any). A given pass phrase is associated with a specific card, so each card can have a different pass phrase. You can change these pass phrases at any time by using the KeySafe **Examine/Change Card** option (available from the **Card Operations** panel) or the **cardpp** command-line utility.

- 9 If you do not want to set a pass phrase for this Administrator Card:
- Select the **No** option.
  - Click the **OK** button.

## 10 After you have created all the Administrator Cards, KeySafe displays a message:

---

```
ACS successfully replaced. Now consider erasing your old
Administrator Cards
```

---

Click the **OK** button, and KeySafe returns you to its introduction panel.

- 11 When you have finished replacing the ACS, erase the old Administrator Cards; for more information, see [Erasing cards and softcards](#) on page 128.

## Replacing an Administrator Card Set with **racs**

**Note** Before you start to replace an ACS, you must ensure you have enough cards to create a complete new ACS. If you start the procedure without enough cards, you are forced to cancel the procedure part way through.

The **racs** utility creates a new ACS to replace a set that was created with the **new-world** utility.

- 1 Ensure the hardware security device is in operational mode.
- 2 Run a command of the form:

---

```
racs [-m|--module=MODULE]
```

---

In this command, the **-m|--module=MODULE** option specifies the **ModuleID** (**MODULE**) of the module to use.

- 3 When prompted, insert the appropriate quorum of Administrator Cards to authorize the replacement.
- 4 When prompted that **racs** is writing the new ACS, insert blank cards as necessary on which to write the replacement Administrator Cards.

When you have finished replacing the ACS, erase the old Administrator Cards. For more information, see [Erasing cards and softcards](#) on page 128.



## Chapter 7: Application interfaces

This chapter explains how to use a module with various types of application:

- Cryptographic Hardware Interface Library (CHIL) applications
- nCipherKM JCA/JCE CSP
- PKCS #11 applications
- nShield native and Custom applications
- CodeSafe applications

You can use KeySafe or the **generatekey** command-line utility to generate or import keys for use with your applications (see Chapter 9: [Working with keys](#)). By default, KeySafe uses the same mechanisms and supports the same applications as the **generatekey** command-line utility.

**Note** You must add the user of any application that uses an nShield module to the group **nfast** before the application runs.

### Cryptographic Hardware Interface Library (CHIL)

The Cryptographic Hardware Interface Library (CHIL) is a simple programming interface for accelerating modulo exponentiation and accessing the RSA/DSA keys that are used by some application software. The Cryptographic Hardware Interface Library supports only RSA/DSA and Diffie-Hellman keys.

An interface for 64-bit CHIL is available on Solaris. The toolkits directory contains two CHIL plug-ins, **libhwcrhk.so** and **libhwcrhk64.so**. The **libhwcrhk64.so** plug-in must be used only with applications that have been compiled using the 64 bit **swspro** compiler.

If your application offers RSA/DSA keys in hardware support through the Cryptographic Hardware Interface Library, use the **hwcrhk** key type.

Some Cryptographic Hardware Interface Library applications that do not support the **hwcrhk** key can still be configured for key storage by using the **embed** key type (provided that they can read **PEM** (Privacy Enhanced Mail) format key files and use the Cryptographic Hardware Interface Library for all cryptographic operations).

## Using keys

Configure the application to load the Cryptographic Hardware Interface Library plug-in:  
`/opt/nfast/toolkits/hwcrhk/libnfhwcrhk.so`

Refer to the documentation for your application.

## Generating keys

Generate the key with KeySafe or **generatekey**, choosing a new identifier for the key; see [Generating keys](#) on page 200. Use the **hwcrhk** or **embed** key type, as appropriate.

The key identifier can only contain digits and lowercase letters; it *cannot* contain spaces, underscores (\_), or hyphens (-).

## nCipher JCA/JCE CSP

### Overview of the nCipherKM JCA/JCE CSP

The nCipherKM JCA/JCE CSP (Cryptographic Service Provider) allows Java applications and services to access the secure cryptographic operations and key management provided by nShield modules. This provider is used with the standard JCE (Java Cryptographic Extension) programming interface.

To use the nCipherKM JCA/JCE CSP, you must install:

- the **javasp Java Support (including KeySafe)** bundle
- the **jceesp nCipherKM JCA/JCE provider classes** component.

For more information about the bundles and components supplied on your Security World Software DVD-ROM, see Appendix C: [Components on Security World Software DVD-ROMs](#).

**Note** Java software is available from <http://java.sun.com/>. If your security policy does not allow the use of downloaded software, these components are available on DVD-ROM from Sun or your operating system vendor.

To use Security World Software Java components, you may need to install patches supplied by your operating system manufacturer. For more information, see the Sun documentation supplied with your Java installation.

Detailed documentation for the JCE interface can be found on the Sun Microsystems Web page <http://java.sun.com/j2se/1.5.0/docs/guide/security/jce/JCERefGuide.html>.

**Note** Softcards are not supported for use with the nCipherKM JCA/JCE CSP in Security Worlds that are compliant with FIPS 140-2 level 3.

## Installing the nCipherKM JCA/JCE CSP

To install the nCipherKM JCA/JCE CSP:

- 1 In the hardserver configuration file, ensure that:
  - **priv\_port** (the port on which the hardserver listens for local privileged TCP connections) is set to 9001
  - **nonpriv\_port** (the port on which the hardserver listens for local nonprivileged TCP connections) is set to 9000.

If you need to change either or both of these port settings, you restart the hardserver before continuing the nCipherKM JCA/JCE CSP installation process. For more information, see [Stopping and restarting the hardserver](#) on page 70.

- 2 Copy the **nCipherKM.jar** file from the **/opt/nfast/java/classes** directory to the extensions folder of your local Java Virtual Machine installation.

The location of the extensions folder depends on the type of your local Java Virtual Machine (JVM) installation:

| JVM type                       | Extensions folder                     |
|--------------------------------|---------------------------------------|
| Java Developer Kit (JDK)       | <i><b>\$JAVA_HOME/jre/lib/ext</b></i> |
| Java Runtime Environment (JRE) | <i><b>\$JAVA_HOME/lib/ext</b></i>     |

In these paths, ***\$JAVA\_HOME*** is the home directory of the Java installation (commonly specified in the **JAVA\_HOME** environment variable).

- 3 Add ***\$JAVA\_HOME/bin*** to your **PATH** system variable.

#### 4 Install the unlimited strength JCE jurisdiction policy files.

The Java Virtual Machine imposes limits on the cryptographic strength that may be used by default with JCE providers. Replace the default policy configuration files with the unlimited strength policy files.

To install the unlimited strength JCE jurisdiction policy files:

- a If necessary, download the archive containing the Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy Files from the Web site of your Java Virtual Machine vendor.

**Note** The Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy Files are covered and controlled by U.S. Export Control laws and may be subject to the export or import laws in other countries. We recommend that you take legal advice before downloading these files from your Java Virtual Machine vendor.

- b Extract the files **local\_policy.jar** and **US\_export\_policy.jar** from Java Virtual Machine vendor's Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy File archive.
- c Copy the extracted files **local\_policy.jar** and **US\_export\_policy.jar** into the security directory for your local Java Virtual Machine (JVM) installation:

| JVM type                       | Extensions folder                          |
|--------------------------------|--------------------------------------------|
| Java Developer Kit (JDK)       | <b><i>\$JAVA_HOME/jre/lib/security</i></b> |
| Java Runtime Environment (JRE) | <b><i>\$JAVA_HOME/lib/security</i></b>     |

In these paths, ***\$JAVA\_HOME*** is the home directory of the Java installation (commonly specified in the **JAVA\_HOME** environment variable).

**Note** Copying the files **local\_policy.jar** and **US\_export\_policy.jar** into the appropriate folder must overwrite any existing files with the same names.

- 5 Add the nCipherKM provider to the Java security configuration file **java.security** (located in the security directory for your local Java Virtual Machine (JVM) installation).

The **java.security** file contains list of providers in preference order that is used by the Java Virtual Machine to decide from which provider to request a mechanism instance. Ensure that the nCipher provider is registered in the first position in this list, as shown in the following example:

---

```
#
List of providers and their preference orders (see above):
#
security.provider.1=com.ncipher.provider.km.nCipherKM
security.provider.2=sun.security.provider.Sun
security.provider.3=sun.security.rsa.SunRsaSign
security.provider.4=com.sun.net.ssl.internal.ssl.Provider
security.provider.5=com.sun.crypto.provider.SunJCE
security.provider.6=sun.security.jgss.SunProvider
security.provider.7=com.sun.security.sasl.Provider
```

---

Placing the nCipher provider first in the list permits the nCipher provider's algorithms to override the algorithms that would be implemented by any other providers (except in cases where you explicitly request another provider name).

**Note** The nCipher provider cannot serve requests required for the SSL classes unless it is in the first position in the list of providers.

Do not change the relative order of the other providers in the list.

**Note** If you add the nCipher provider as **security.provider.1**, ensure that the subsequent providers are re-numbered correctly. Ensure you do not list multiple providers with the same number (for example, ensure your list of providers does not include two instances of **security.provider.1**, both **com.ncipher.provider.km.nCipherKM** and another provider).

- 6 Save your updates to the file **java.security**.

When you have installed the nCipherKM JCA/JCE CSP, you must have created a Security World before you can test or use it. For more information about creating a Security World, see [Creating a Security World](#) on page 71.

**Note** If you have a Java Enterprise Edition Application Server running, you must restart it before the installed nCipher provider is loaded into the Application Server virtual machine and ready for use.



## Testing the nCipherKM JCA/JCE CSP installation

After installation, you can test that the nCipherKM JCA/JCE CSP is functioning correctly by running the command:

---

```
java com.ncipher.provider.InstallationTest
```

---

**Note** For this command to work, you must have added ***\$JAVA\_HOME*** to your **PATH** system variable.

If the nCipherKM JCA/JCE CSP is functioning correctly, output from this command has the following form:

---

```
Installed providers:
1: nCipherKM
2: SUN
3: SunRsaSign
4: SunJSSE
5: SunJCE
6: SunJGSS
7: SunSASL
Unlimited strength jurisdiction files are installed.
The nCipher provider is correctly installed.
nCipher JCE services:
Alg.Alias.Cipher.1.2.840.113549.1.1.1
Alg.Alias.Cipher.1.2.840.113549.3.4
Alg.Alias.Cipher.AES
Alg.Alias.Cipher.DES3
....
```

---

If the **nCipherKM** provider is installed but is not registered at the top of the providers list in the **java.security** file, the **InstallationTest** command produces output that includes the message:

---

```
The nCipher provider is installed, but is not registered at
the top of the providers list in the java.security file. See
the user guide for more information about the recommended
system configuration.
```

---

In such a case, edit the **java.security** file (located in the security directory for your local JVM installation) so that the nCipher provider is registered in the first position in that file's list of providers. For more information about the **java.security** file, see [Installing the nCipherKM JCA/JCE CSP](#) on page 157.

If the nCipher provider is not installed at all, or you have not created a Security World, or if you have not configured ports correctly in the hardserver configuration file, the **InstallationTest** command produces output that includes the message:

---

```
The nCipher provider is not correctly installed.
```

---

In such case:

- Check that you have configured ports correctly, as described in [Installing the nCipherKM JCA/JCE CSP](#) on page 157. For more information about hardserver configuration file settings, see [server\\_startup](#) on page 284.
- Check that you have created a Security World. If you have not created a Security World, create a Security World. For more information, see [Creating a Security World](#) on page 71.
- If you have already created a Security World, repeat the nCipher JCA/JCE CSP installation process as described in [Installing the nCipherKM JCA/JCE CSP](#) on page 157.

After making any changes to the nCipherKM JCA/JCE CSP installation, run the **InstallationTest** command again and check the output.

Whether or not the nCipher provider is correctly installed, if the unlimited strength jurisdiction files are not installed or (not correctly installed), the **InstallationTest** command produces output that includes the message:

---

```
Unlimited strength jurisdiction files are NOT installed.
```

---

**Note** The **InstallationTest** command can only detect this situation if you are using JRE/JDK version 1.5 or later.

This message means that, because the Java Virtual Machine imposes limits on the cryptographic strength that you can use by default with JCE providers, you must replace the default policy configuration files with the unlimited strength policy files. For information about how to install the unlimited strength jurisdiction files, see [Installing the nCipherKM JCA/JCE CSP](#) on page 157.

## System properties

You can use system properties to control the provider. You set system properties when starting the Java Virtual Machine using a command such as:

---

```
java -Dproperty=value MyJavaApplication
```

---

In this example command, *property* represents any system property, *value* represents the value set for that property, and *MyJavaApplication* is the name of the Java application you are starting. You can set multiple system properties in a single command, for example:

---

```
java -Dprotect=module -DignorePassphrase=true MyJavaApplication
```

---

The available system properties and their functions as controlled by setting different values for a property are described in the following table:

| Property                | Function for different values                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|-------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>JCECSP_DEBUG</b>     | This property is a bit mask for which different values specify different debugging functions; the default value is <b>0</b> . For details about the effects of setting different values for this property, see <a href="#">JCECSP_DEBUG property values</a> on page 164.                                                                                                                                                                                                                    |
| <b>JCECSP_DEBUGFILE</b> | This property specifies a path to the file to which logging output is to be written. Set this property if the <b>JCECSP_DEBUG</b> property is set to a value other than the default of <b>0</b> . For details about the effects of setting different values for this property, see <a href="#">JCECSP_DEBUG property values</a> on page 164. In a production environment, we recommend that you disable debug logging to prevent sensitive information being made available to an attacker. |
| <b>protect</b>          | This property specifies the type of protection to be used for key generation and nCipherKM KeyStore instances. You can set the value of this property to one of <b>module</b> , <b>softcard:IDENT</b> or <b>cardset</b> . OCS protection ( <b>cardset</b> ) uses the card from the first slot of the first usable hardware security device. To find the logical token hash <b>IDENT</b> of a softcard, run the command <b>nfkminfo --softcard-list</b> .                                    |
| <b>module</b>           | This property lets you override the default module and select a specific module to use for module and OCS protection. Set the value of this property as the ESN of the module you want to use.                                                                                                                                                                                                                                                                                              |
| <b>slot</b>             | This property lets you override the default slot for OCS-protection and select a specific slot to use. Set this the value of this property as the number of the slot you want to use.                                                                                                                                                                                                                                                                                                       |
| <b>ignorePassphrase</b> | If the value of this property is set to <b>true</b> , the nCipherKM provider ignores the pass phrase provided in its KeyStore implementation. This feature is included to allow the Sun or IBM <b>keytool</b> utilities to be used with module-protected keys. The <b>keytool</b> utilities require a pass phrase be provided; setting this property allows a dummy pass phrase to be used.                                                                                                 |
| <b>seeintegname</b>     | Setting the value of this property to the name of an SEE integrity key causes the provider to generate SEE application keys. These keys may only be used by an SEE application signed with the named key.                                                                                                                                                                                                                                                                                   |

| Property                                 | Function for different values                                                                                                                                                                                                                                                                                                                                                                                  |
|------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>com.ncipher.provider.announcemode</b> | The default value for this property is <b>auto</b> , which uses firmware auto-detection to disable algorithms in the provider that cannot be supported across all installed modules. Setting the value of this property to <b>on</b> forces the provider to advertise all mechanisms at start-up. Setting the value of this property to <b>off</b> forces the provider to advertise no mechanisms at start-up. |
| <b>com.ncipher.provider.enable</b>       | For the value of this property, you supply a comma-separated list of mechanism names that are to be forced on, regardless of the announce mode selected.                                                                                                                                                                                                                                                       |
| <b>com.ncipher.provider.disable</b>      | For the value of this property, you supply a comma-separated list of mechanism names that are to be forced off, regardless of the announce mode selected. Any mechanism supplied in the value for the <b>com.ncipher.provider.disable</b> property overrides the same mechanism if it is supplied in the value for the <b>com.ncipher.provider.enable</b> property.                                            |

## JCECSP\_DEBUG property values

The **JCECSP\_DEBUG** system property is a bit mask for which you can set different values to control the debugging functions. The following table describes the effects of different values that you can set for this property:

| JCECSP_DEBUG value | Function                                                                                                                                                                                                     |
|--------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>0</b>           | If this property has no bits set, no debugging information is reported. This is the default setting.                                                                                                         |
| <b>1</b>           | If this property has the bit 1 set, minimal debugging information (for example, version information and critical errors) is reported.                                                                        |
| <b>2</b>           | If this property has the bit 2 set, comprehensive debugging information is reported.                                                                                                                         |
| <b>4</b>           | If this property has the bit 3 set, debugging information relating to creation and destruction of memory and module resources is reported.                                                                   |
| <b>8</b>           | If this property has the bit 4 set, <b>debugFunc</b> and <b>debugFuncEnd</b> generate debugging information for functions that call them.                                                                    |
| <b>16</b>          | If this property has the bit 5 set, <b>debugFunc</b> and <b>debugFuncEnd</b> display the values for all the arguments that are passed in to them.                                                            |
| <b>32</b>          | If this property has the bit 6 set, context information is reported with each debugging message (for example, the <b>ThreadID</b> and the current time).                                                     |
| <b>64</b>          | If this property has the bit 7 set, the time elapsed during each logged function is calculated, and information on the number of times a function is called and by which function it was called is reported. |
| <b>128</b>         | If this property has the bit 8 set, debugging information for NFJAVA is reported in the debugging file.                                                                                                      |
| <b>256</b>         | If this property has the bit 9 set, the call stack is printed for every debug message.                                                                                                                       |

To set multiple logging functions, add up the **JCECSP\_DEBUG** values for the debugging functions you want to set, and specify the total as the value for **JCECSP\_DEBUG**. For example, if you want to set the debugging to use both function tracing (bit 4) and function tracing with parameters (bit 5), add the **JCECSP\_DEBUG** values shown in the table for these debugging functions ( $8 + 16 = 24$ ) and specify this total (**24**) as the value to use for **JCECSP\_DEBUG**.

## Compatibility

The nCipherKM JCA/JCE CSP supports both module-protected keys and OCS-protected keys. The CSP currently supports 1/*N* OCSs and a single protection type for each nCipherKM JCE KeyStore.

You can use the nCipherKM JCA/JCE CSP with Security Worlds that comply with FIPS 140-2 at either level 2 or level 3.

**Note** In a Security World that complies with FIPS 140-2 level 3, it is not possible to import keys generated by other JCE providers.

The nCipherKM JCA/JCE CSP supports load-sharing for keys that are stored in the nCipherKM KeyStore. This feature allows a server to spread the load of cryptographic operations across multiple connected modules, providing greater scalability.

**Note** We recommend that you use load-sharing unless you have existing code that is designed to run with multiple modules. To share keys with load-sharing, you must create a 1/*N* OCS with at least as many cards as you have modules. All the cards in the OCS must have the same pass phrase.

Keys generated or imported by the nCipherKM JCA/JCE CSP are not recorded into the Security World until:

- 1 The key is added to an nCipherKM KeyStore (by using a call to **setKeyEntry()** or **setCertificateEntry()**).
- 2 That nCipherKM KeyStore is then stored (by using a call to **store()**).

The pass phrase used with the KeyStore must be the pass phrase of the card from the OCS that protects the keys in the KeyStore.

## nCipher PKCS #11 library

To use the nCipher PKCS #11 library, you must tell the application the name and location of the library. The exact method for doing this depends on the application.

Instructions for using the nCipher PKCS #11 library with specific applications are available from the Thales Web site: <http://iss.thalesgroup.com/Resources.aspx?>. Alternatively, contact Support.

Depending on the application, you may need to set the path and library name `/opt/nfast/toolkits/pkcs11/libcknfast.so` in a dialog or configuration file.

The nCipher PKCS #11 library has security options which you must configure before you use the PKCS #11 library. For more information, see [PKCS #11 library with Security Assurance Mechanism](#) on page 172.

From version 1.7, the nCipher PKCS #11 library can be used with FIPS 140-2 level 3 compliant security worlds. This version of the library also introduces load-sharing. This feature provides support for multiple hardware security devices that are connected to a single server, spreading the load of cryptographic operations between the modules in order to provide scalability in terms of performance.

**Note** We recommend that you use load-sharing unless you have existing code that is designed to run with multiple hardware security devices.

To share keys with load-sharing, you must create a  $1/N$  OCS that contains at least as many cards as you have modules. All the cards on the OCS must have the same pass phrase.

**Note** If you are using the **preload** command-line utility in conjunction with the nCipher PKCS #11 library, you can create  $K/N$  OCSs.

## Choosing functions

Some PKCS #11 applications enable you to choose which functions you want to perform on the PKCS #11 token and which functions you want to perform in your application.

The following paragraphs in this section describe the functions that an nShield module can provide.

### Generating random numbers and keys

The nShield module includes a hardware random number generator. A hardware random number generator provides greater security than the pseudo-random number generators provided by host computers. Therefore, always use the nShield module to generate random numbers and keys.

### Digital signatures

The nCipher PKCS #11 library can use the nShield module to sign and verify messages using the following algorithms:

- DSA
- RSA
- DES3\_MAC
- AES
- ECDSA (if the appropriate feature is enabled)

A Thales hardware security device is specifically optimized for public key algorithms, and therefore it will provide significant acceleration for DSA and RSA signature generation and verification. You should always choose to perform asymmetric signature generation and verification with an nShield module (that is, RSA and DSA).

## Asymmetric encryption

The nCipher PKCS #11 library can use an nShield module to perform asymmetric encryption and decryption with the RSA algorithm.

The nShield module is specifically optimized for asymmetric algorithms, so you should always choose to perform asymmetric operations with the nShield module.

## Symmetric encryption

The nCipher PKCS #11 library can use the nShield module to perform symmetric encryption with the following algorithms:

- DES
- Triple DES
- AES

Because of limitations on throughput, these operations can be slower on the nShield module than on the host computer. However, although the nShield module may be slower than the host under a light load, you may find that under a heavy load the advantage gained from off-loading the symmetric cryptography (which frees the host CPU for other tasks) means that you achieve better overall performance.

## Message digest

The nCipher PKCS #11 library can perform message digest operations with MD2, MD5, SHA-1, SHA-224, SHA-256, SHA-384, and SHA-512 algorithms. However, for reasons of throughput, the library performs these operations on the host computer.

## Mechanisms

The following table lists the mechanisms currently supported by the nCipher PKCS #11 library. Thales also provides vendor-supplied mechanisms, see the *Cryptographic API Integration Guide*.

| Mechanism                            | Functions         |                |                |        |                   |               |                |
|--------------------------------------|-------------------|----------------|----------------|--------|-------------------|---------------|----------------|
|                                      | Encrypt & Decrypt | Sign & Verify  | SR & VR        | Digest | Gen. Key/Key Pair | Wrap & Unwrap | Derive Key     |
| CKM_RSA_PKCS_KEY_PAIR_GEN            |                   |                |                |        | Y                 |               |                |
| CKM_RSA_X9_31_KEY_PAIR_GEN           |                   |                |                |        | Y                 |               |                |
| CKM_RSA_PKCS                         | Y <sup>1</sup>    | Y <sup>1</sup> | Y <sup>1</sup> |        |                   | Y             |                |
| CKM_RSA_9796                         |                   | Y <sup>1</sup> | Y <sup>1</sup> |        |                   |               |                |
| CKM_RSA_X_509                        | Y <sup>1</sup>    | Y <sup>1</sup> | Y <sup>1</sup> |        |                   | X             |                |
| CKM_MD2_RSA_PKCS                     |                   | Y              |                |        |                   |               |                |
| CKM_MD5_RSA_PKCS                     |                   | Y              |                |        |                   |               |                |
| CKM_SHA_PKCS_OAEP <sup>10</sup>      | Y                 |                |                |        |                   | Y             |                |
| CKM_SHA1_RSA_PKCS                    |                   | Y              |                |        |                   |               |                |
| CKM_SHA256_RSA_PKCS                  |                   | Y              |                |        |                   |               |                |
| CKM_SHA384_RSA_PKCS                  |                   | Y              |                |        |                   |               |                |
| CKM_SHA512_RSA_PKCS                  |                   | Y              |                |        |                   |               |                |
| CKM_RSA_PKCS_PSS <sup>9</sup>        | Y                 | Y              |                |        |                   |               |                |
| CKM_SHA1_RSA_PKCS_PSS <sup>9</sup>   |                   | Y              |                |        |                   |               |                |
| CKM_SHA224_RSA_PKCS_PSS <sup>9</sup> |                   | Y              |                |        |                   |               |                |
| CKM_SHA256_RSA_PKCS_PSS <sup>9</sup> |                   | Y              |                |        |                   |               |                |
| CKM_SHA384_RSA_PKCS_PSS <sup>9</sup> |                   | Y              |                |        |                   |               |                |
| CKM_SHA512_RSA_PKCS_PSS <sup>9</sup> |                   | Y              |                |        |                   |               |                |
| CKM_DSA_KEY_PAIR_GEN                 |                   |                |                |        | Y                 |               |                |
| CKM_DSA                              |                   | Y <sup>1</sup> |                |        |                   |               |                |
| CKM_DSA_SHA1                         |                   | Y              |                |        |                   |               |                |
| CKM_DH_PKCS_KEY_PAIR_GEN             |                   |                |                |        | Y                 |               |                |
| CKM_DH_PKCS_DERIVE                   |                   |                |                |        |                   |               | Y              |
| CKM_EC_KEY_PAIR_GEN                  |                   |                |                |        | Y <sup>11</sup>   |               |                |
| CKM_ECDSA                            |                   | Y <sup>1</sup> |                |        |                   |               |                |
| CKM_ECDSA_SHA1                       |                   | Y              |                |        |                   |               |                |
| CKM_ECDH1_DERIVE                     |                   |                |                |        |                   |               | Y <sup>6</sup> |
| CKM_GENERIC_SECRET_KEY_GEN           |                   |                |                |        | Y                 |               |                |



| Mechanism                 | Functions         |                |         |        |                   |                |                |
|---------------------------|-------------------|----------------|---------|--------|-------------------|----------------|----------------|
|                           | Encrypt & Decrypt | Sign & Verify  | SR & VR | Digest | Gen. Key/Key Pair | Wrap & Unwrap  | Derive Key     |
| CKM_DES_KEY_GEN           |                   |                |         |        | Y <sup>7</sup>    |                |                |
| CKM_DES_ECB               | Y <sup>7</sup>    |                |         |        |                   | Y <sup>7</sup> |                |
| CKM_DES_CBC               | Y <sup>7</sup>    |                |         |        |                   | Y <sup>7</sup> |                |
| CKM_DES_CBC_PAD           | Y <sup>7</sup>    |                |         |        |                   | Y <sup>7</sup> |                |
| CKM_DES_MAC               |                   | Y <sup>7</sup> |         |        |                   |                |                |
| CKM_DES_ECB_ENCRYPT_DATA  |                   |                |         |        |                   |                | Y <sup>7</sup> |
| CKM_DES_MAC_GENERAL       |                   | Y <sup>7</sup> |         |        |                   |                |                |
| CKM_AES_KEY_GEN           |                   |                |         |        | Y                 |                |                |
| CKM_AES_ECB               | Y                 |                |         |        |                   | Y <sup>5</sup> |                |
| CKM_AES_CBC               | Y                 |                |         |        |                   | Y <sup>5</sup> |                |
| CKM_AES_CBC_PAD           | Y                 |                |         |        |                   | Y              |                |
| CKM_AES_MAC               |                   | Y              |         |        |                   |                |                |
| CKM_AES_MAC_GENERAL       |                   | Y              |         |        |                   |                |                |
| CKM_AES_CMAC              |                   | Y              |         |        |                   |                |                |
| CKM_AES_CMAC_GENERAL      |                   | Y              |         |        |                   |                |                |
| CKM_DES2_KEY_GEN          |                   |                |         |        | Y                 |                |                |
| CKM_DES3_KEY_GEN          |                   |                |         |        | Y                 |                |                |
| CKM_DES3_ECB              | Y                 |                |         |        |                   | Y <sup>5</sup> |                |
| CKM_DES3_EBC_ENCRYPT_DATA |                   |                |         |        |                   |                | Y              |
| CKM_DES3_CBC              | Y                 |                |         |        |                   | Y <sup>5</sup> |                |
| CKM_DES3_CBC_PAD          | Y                 |                |         |        |                   | Y              |                |
| CKM_DES3_CBC_ENCRYPT_DATA |                   |                |         |        |                   |                | Y              |
| CKM_DES3_MAC_GENERAL      |                   | Y              |         |        |                   |                |                |
| CKM_DES3_MAC              |                   | Y              |         |        |                   |                |                |
| CKM_MD2                   |                   |                |         | Y      |                   |                |                |
| CKM_MD5                   |                   |                |         | Y      |                   |                |                |
| CKM_MD5_HMAC_KEY_GEN      |                   |                |         |        | Y <sup>7</sup>    |                |                |
| CKM_MD5_HMAC              |                   | Y <sup>7</sup> |         |        |                   |                |                |
| CKM_MD5_HMAC_GENERAL      |                   | Y <sup>7</sup> |         |        |                   |                |                |
| CKM_SHA_1                 |                   |                |         | Y      |                   |                |                |
| CKM_SHA_1_HMAC            |                   | Y <sup>4</sup> |         |        |                   |                |                |
| CKM_SHA_1_HMAC_GENERAL    |                   | Y <sup>4</sup> |         |        |                   |                |                |
| CKM_SHA224                |                   |                |         | Y      |                   |                |                |

| Mechanism                    | Functions  |               |                   |        |         |                |                   |
|------------------------------|------------|---------------|-------------------|--------|---------|----------------|-------------------|
|                              | Derive Key | Wrap & Unwrap | Gen. Key/Key Pair | Digest | SR & VR | Sign & Verify  | Encrypt & Decrypt |
| CKM_SHA224_HMAC              |            |               |                   |        |         | Y <sup>4</sup> |                   |
| CKM_SHA224_HMAC_GENERAL      |            |               |                   |        |         | Y <sup>4</sup> |                   |
| CKM_SHA256                   |            |               |                   | Y      |         |                |                   |
| CKM_SHA256_HMAC              |            |               |                   |        |         | Y <sup>4</sup> |                   |
| CKM_SHA256_HMAC_GENERAL      |            |               |                   |        |         | Y <sup>4</sup> |                   |
| CKM_SHA384                   |            |               |                   | Y      |         |                |                   |
| CKM_SHA384_HMAC              |            |               |                   |        |         | Y <sup>4</sup> |                   |
| CKM_SHA384_HMAC_GENERAL      |            |               |                   |        |         | Y <sup>4</sup> |                   |
| CKM_SHA512                   |            |               |                   | Y      |         |                |                   |
| CKM_SHA512_HMAC              |            |               |                   |        |         | Y <sup>4</sup> |                   |
| CKM_SHA512_HMAC_GENERAL      |            |               |                   |        |         | Y <sup>4</sup> |                   |
| CKM_PBE_MD2_DES_CBC          |            |               | Y                 |        |         |                |                   |
| CKM_PBE_MD5_DES_CBC          |            |               | Y                 |        |         |                |                   |
| CKM_XOR_BASE_AND_DATA        |            |               |                   |        |         |                | Y <sup>3</sup>    |
| CKM_CONCATENATE_BASE_AND_KEY |            |               |                   |        |         |                | Y <sup>8</sup>    |
| CKM_RIPEMD160                |            |               |                   | Y      |         |                |                   |

The nCipher library supports some mechanisms that are defined in versions of the PKCS #11 standard later than 2.01, although the nCipher library does not fully support versions of the PKCS #11 standard later than 2.01. In the table above:

- Empty cells indicate mechanisms that are not supported by the PKCS #11 standard.
- The entry “Y” indicates that a mechanism is supported by the nCipher PKCS #11 library.
- The entry “X” indicates that a mechanism is not supported by the nCipher PKCS #11 library.

In the table above, annotations with the following numbers indicate:

- 1 Single-part operations only.
- 2 This mechanism uses the eight octets following the key as the initializing vector as specified in PKCS#5 v2.
- 3 The base key and the derived key are restricted to **DES**, **DES3**, **CAST5** or **Generic**, though they may be of different types.

- 4 This mechanism depends on the vendor-defined key generation mechanism **CKM\_SHA\_1\_HMAC\_KEY\_GEN**, **CKM\_NC\_SHA224\_HMAC\_KEY\_GEN**, **CKM\_NC\_SHA256\_HMAC\_KEY\_GEN**, **CKM\_NC\_SHA384\_HMAC\_KEY\_GEN**, or **CKM\_NC\_SHA512\_HMAC\_KEY\_GEN**. For more information, see the *Cryptographic API Integration Guide*.
- 5 Wrap secret keys only (private key wrapping must use **CBC\_PAD**).
- 6 The **CKM\_ECDH1\_DERIVE** mechanism is supported. However, the mechanism only takes a **CK\_ECDH1\_DERIVE\_PARAMS** struct in which **CK\_EC\_KDF\_TYPE** is **CKD\_NULL**, **CKD\_SHA1\_KDF**, **CKD\_SHA224\_KDF**, **CKD\_SHA256\_KDF**, **CKD\_SHA384\_KDF**, or **CKD\_SHA512\_KDF**. For more information on **CK\_ECDH1\_DERIVE\_PARAMS**, see the PKCS #11 standard.

For the **pPublicData\*** parameter, a raw octet string value (as defined in section A.5.2 of ANSI X9.62) and DER-encoded ECPoint value (as defined in section E.6 of ANSI X9.62) are now accepted.

- 7 These mechanisms are not supported in strict FIPS 140-2 Level 3 Security Worlds in firmware version 2.33.60 or later.
- 8 Before you can create a key for use with the derive mechanism **CKM\_CONCATENATE\_BASE\_AND\_KEY**, you must first specify the **CKA\_ALLOWED\_MECHANISMS** attribute in the template with the **CKM\_CONCATENATE\_BASE\_AND\_KEY** set. Specifying the **CKA\_ALLOWED\_MECHANISMS** in the template enables the setting of the nCore level ACL, which enables the key in this derive key operation. For more information about the Security Assurance Mechanisms (SAMs) on the **CKM\_CONCATENATE\_BASE\_AND\_KEY** mechanism and the **CKA\_ALLOWED\_MECHANISMS** attribute, see the *Cryptographic API Integration Guide*.
- 9 The **hashAlg** and the **mgf** that are specified by the **CK\_RSA\_PKCS\_PSS\_PARAMS** must have the same SHA hash size. If they do not have the same hash size, then the signing or verify fails with a return value of **CKR\_MECHANISM\_PARAM\_INVALID**.

The **sLen** value is expected to be the length of the message hash. If this is not the case, then the signing or verify again fails with a return value of **CKR\_MECHANISM\_PARAM\_INVALID**. The Security World Software implementation of **RSA\_PKCS\_PSS** salt lengths are as follows:

| Mechanism | Salt-length |
|-----------|-------------|
| SHA-1     | 160-bit     |
| SHA-224   | 224-bit     |
| SHA-256   | 256-bit     |
| SHA-384   | 384-bit     |
| SHA-512   | 512-bit     |

- 10 The **hashAlg** and the **mgf** that are specified by the **CK\_RSA\_PKCS\_OEAP\_PARAMS** must have the same SHA hash size. If they do not have the same hash size, then the signing or verify fails with a return value of **CKR\_MECHANISM\_PARAM\_INVALID**.

It is possible to specify a byte array using a data source if **CKZ\_DATA\_SPECIFIED** is set. If **CKZ\_DATA\_SPECIFIED** is not present, then **pSourceData** and **pSourceDataLen** are ignored. It is not a requirement to have source set, and the value can be zero.

- 11 For elliptic curve key pairs: when generating a key pair using **C\_GenerateKeyPair()**, you may specify either **CKA\_DERIVE** or **CKA\_SIGN** but not both. This means that your **CKK\_EC** key can only be used for either sign/verify or derive operations. If both types are included in the template, generation fails with **CKR\_TEMPLATE\_INCONSISTENT**. If nothing is specified in the template, then the default is sign/verify.

Key generation does calculate its own curves but, as shown in the PKCS #11 standard, takes the **CKA\_PARAMS**, which contains the curve information (similar to that of a discrete logarithm group in the generation of a DSA key pair). **CKA\_EC\_PARAMS** is a Byte array which is DER-encoded of an ANSI X9.62 Parameters value. It can take both named curves and custom curves.

Note Brainpool curves cannot be specified as named curves: they must be supplied as custom curves with all parameters DER-encoded.

The following PKCS #11-specific flags describe which curves are supported:

- **CKF\_EC\_P**: prime curve supported
- **CKF\_EC\_2M**: binary curve supported
- **CKF\_EC\_PARAMETERS**: supplying your own custom parameters is supported
- **CKF\_EC\_NAMECURVE**: supplying a named curve is supported
- **CKF\_EC\_UNCOMPRESS**: supports uncompressed form only, compressed form not supported.

## PKCS #11 library with Security Assurance Mechanism

It is possible for an application to use the PKCS #11 API in ways that do not necessarily provide the expected security benefits, or which might introduce additional weaknesses. For example, the PKCS #11 standard requires the nCipher library to be able to generate keys that are extractable from the module in plaintext. An application could use this ability in error, when a secure key would be more appropriate.

The PKCS #11 library with the Security Assurance Mechanism (SAM), **libcknfast**, can help users to identify potential weaknesses, and help developers create secure PKCS #11 applications more easily.

The SAM in the PKCS #11 library is intended to detect operations that reveal questionable behavior by the application. If these occur, the application fails with an explanation of the cause of failure.

After a review of your security policy and the way the application uses the PKCS #11 library with the SAM, if there are questionable operations that are considered to be acceptable and pose no security risk, the PKCS #11 library can be configured to permit some, or all, of them by means of the **CKNFAST\_OVERRIDE\_SECURITY\_ASSURANCES** environment variable (described in [CKNFAST\\_OVERRIDE\\_SECURITY\\_ASSURANCES](#) on page 180).

**Note** To ensure the security of your keys, you must review any messages returned by the PKCS #11 library before changing the settings of the **CKNFAST\_OVERRIDE\_SECURITY\_ASSURANCES** environment variable.

The **CKNFAST\_OVERRIDE\_SECURITY\_ASSURANCES** environment variable uses a semicolon separated list of parameters, with associated values, to explicitly allow operations that could compromise the security of cryptographic keys if the operations are not well understood.

If no parameters, or the **none** parameter, are supplied to the **CKNFAST\_OVERRIDE\_SECURITY\_ASSURANCES** #11 library fails to perform the operation in question, and issues a warning, when the following operations are detected:

- creating short-term session keys as long-term objects
- creating keys that can be exported as plain text
- importing keys from external sources
- creating or importing wrapping keys
- creating or importing unwrapping keys
- creating keys with weak algorithms (such as DES)
- creating keys with short key lengths.

For more information about parameters and diagnostic warnings, see [CKNFAST\\_OVERRIDE\\_SECURITY\\_ASSURANCES](#) on page 180.

## Key security

Questionable operations largely relate to the concept of a key being *secure*. A private or secret key is considered insecure if there is some reason for believing that its value may be available outside the module. Public keys are never considered insecure; by definition they are intended to be public.

An explicitly insecure PKCS #11 key is one where **CKA\_SENSITIVE** is set to false. If an application uses a key that is insecure but **CKA\_SENSITIVE** is not set to false, it is possible that the application is using an inadequate concept of key security, and that the library disallows use of that key by default. Use of insecure keys should, by default, be restricted to short-term session keys, and applications should explicitly recognize the insecurity.

## Using the nCipher PKCS #11 library

After you have loaded the nCipher PKCS #11 library, it is added to your application's list of cryptographic modules or PKCS #11 slots.

Whether or not the library uses load-sharing depends on the value of the **CKNFAST\_LOADSHARING** environment variable, described in [CKNFAST\\_LOADSHARING](#) on page 179.

### nCipher PKCS #11 library with load-sharing

The nCipher PKCS #11 library creates a virtual slot for every OCS in the security world (returning the name of the card set) unless you have set **CKNFAST\_CARDSET\_HASH** (as described in [CKNFAST\\_CARDSET\\_HASH](#) on page 178).

An additional virtual slot may be returned (with the label of **accelerator** ), depending on the value given to the variable **CKNFAST\_NO\_ACCELERATOR\_SLOTS** (described in [CKNFAST\\_NO\\_ACCELERATOR\\_SLOTS](#) on page 179). Accelerator slots can:

- be used to support session objects
- be used to create module protected keys
- not be used to create private objects.

When you insert a smart card from an OCS in the current security world, the nCipher PKCS #11 library treats this card as a PKCS #11 token that is present in the virtual slot for that OCS.

After the PKCS #11 token is present, you can open a session to that token. Until you log in, a session can only access public objects that belongs to that PKCS #11 token.

The PKCS #11 token is present until you remove the last card belonging to the OCS. When you remove the token, the nCipher PKCS #11 library closes any open sessions.

Logging in gives access to the private objects that are protected by the PKCS #11 token. Logging in requires the pass phrase for the OCS. The exact mechanism for supplying the pass phrase depends on the application that you are running.

The PKCS #11 token is shared across all the modules that have a smart card from the OCS in the reader at the point that you log in. After you have logged in, inserting additional cards from this OCS has no effect.

If you remove a smart card that belongs to a logged-in token, the nCipher PKCS #11 library closes any open sessions and marks the token as being not present (unless the OCS is persistent). Removing a card from a persistent OCS has no effect, and the PKCS #11 token remains present until you log out.

## nCipher PKCS #11 library without load-sharing

There will be two entries for each module, unless you have set **CKNFAST\_NO\_ACCELERATOR\_SLOTS**.

**Note** The entry called **accelerator** cannot be used to create private objects. It can be used to create module-protected keys.

Use the second of the two entries (which has the same name as the Operator Card that is currently in the smart card reader) to protect your keys or token objects.

PKCS #11 does not allow two tokens to be present in the same slot. Therefore, when you insert a smart card into a reader, the nCipher PKCS #11 library logs out any previously logged-in token from the slot and closes any open sessions.

## nCipher PKCS #11 library with the preload utility

You can use the **preload** command-line utility to preload *K/N* OCSs before actually using PKCS #11 applications. The **preload** utility loads the logical token and then passes it to the PKCS #11 utilities.

You must provide any required pass phrase for the tokens when using **preload** to load the card set. However, because the application is not aware that the card set has been preloaded, the application operates normally when handling the login activity (including prompting for a pass phrase), but the PKCS #11 library will not actually check the supplied pass phrase.

Normally, **preload** uses environment variables to pass information to the program using the preloaded objects, including the PKCS #11 library. Therefore, if the application you are using is one that clears its environment before the PKCS #11 library is loaded, you must set the appropriate values in the **cknfastrc** file (see [nCipher PKCS #11 library environment variables](#) on page 176). The current environment variables remain usable. The default setting for the **CKNFAST\_LOADSHARING** environment variable changes from specifying load-sharing as

disabled to specifying load-sharing as enabled. Moreover, in load-sharing mode, the loaded card set is used to set the environment variable **CKNFAST\_CARDSET\_HASH** so that only the loaded card set is visible as a slot.

The **NFAST\_NFKM\_TOKENSFILE** environment variable must also be set in the **cknfastrc** file to the location of the preload file (see [Environment variables](#) on page 245).

A logical token preloaded by **preload** for use with the nCipher PKCS #11 library is the only such token available to the application for the complete invocation of the library. You can use more than one module with the same card set.

If the loaded card set is non-persistent, then a card must be left in each module on which the set has been loaded during the start-up sequence. After a non-persistent card has been removed, the token is not present even if the card is reinserted.

If load-sharing has been specifically switched off, you see multiple slots with the same label.

## nCipher PKCS #11 library environment variables

The nCipher PKCS #11 library uses the following environment variables:

- **CKNFAST\_ASSUME\_SINGLE\_PROCESS**
- **CKNFAST\_ASSURANCE\_LOG**
- **CKNFAST\_CARDSET\_HASH**
- **CKNFAST\_DEBUG**
- **CKNFAST\_DEBUGDIR**
- **CKNFAST\_DEBUGFILE**
- **CKNFAST\_FAKE\_ACCELERATOR\_LOGIN**
- **CKNFAST\_LOADSHARING**
- **CKNFAST\_NO\_ACCELERATOR\_SLOTS**
- **CKNFAST\_NO\_SYMMETRIC**
- **CKNFAST\_NO\_UNWRAP**
- **CKNFAST\_NONREMOVABLE**
- **CKNFAST\_NVRAM\_KEY\_STORAGE**
- **CKNFAST\_OVERRIDE\_SECURITY\_ASSURANCES.**



- **CKNFAST\_SESSION\_THREADSafe**
- **CKNFAST\_TOKENS\_PERSISTENT**
- **CKNFAST\_USE\_THREAD\_UPCALLS**
- **CKNFAST\_LOAD\_KEYS**
- **CKNFAST\_WRITE\_PROTECTED**

If you used the default values in the installation script, you should not need to change any of these environment variables.

You can set environment variables in the file **cknfastrc**. This file must be in the **/opt/nfast/** directory.

Each line of the file **cknfastrc** must be of the following form:

---

```
variable=value
```

---

**Note** Variables set in the environment are used in preference to those set in the resource file.

Changing the values of these variables after you start your application has no effect until you restart the application.

If the description of a variable does not explicitly state what values you can set, the values you set are normally **1** or **0**, **Y** or **N**.

**Note** For more information concerning Security World Software environment variables that are not specific to PKCS #11 and which are used to configure the behavior of your nCipher installation, see the Security World Software installation instructions.

## CKNFAST\_ASSUME\_SINGLE\_PROCESS

By default, this variable is set to **1**. This specifies that only token objects that are loaded at the time **C\_Initialize** is called are visible.

Setting this variable to **0** means that token objects created in one process become visible in another process when it calls **C\_FindObjects**. Existing objects are also checked for modification on disc; if the key file has been modified, then the key is reloaded. Calling **C\_SetAttributeValues** or **C\_GetAttributeValues** also checks whether the object to be changed has been modified in another process and reloads it to ensure the most recent copy is changed.

Setting the variable to **0** can slow the library down because of the additional checking needed if a large number of keys are being changed and a large number of existing objects must be reloaded.

## CKNFAST\_ASSURANCE\_LOG

This variable is used to direct all warnings from the Security Assurance Mechanism to a specific log file.

## CKNFAST\_CARDSET\_HASH

This variable enables you to specify a specific card set to be used in load-sharing mode. If this variable is set, only the virtual smart card slot that matches the specified hash is present (plus the accelerator slot). The hash that you use to identify the card set in **CKNFAST\_CARDSET\_HASH** is the SHA-1 hash of the secret on the card. Use the **nfkminfo** command-line utility to identify this hash for the card set that you want to use: it is listed as **hkltu**. For more information about using **nfkminfo**, see [nfkminfo: information utility](#) on page 259.

## CKNFAST\_DEBUG

This variable is set to enable PKCS #11 debugging. The values you can set are in the range **0** - **11**. If you are using **NFLOG\_\*** for debugging, you must set **CKNFAST\_DEBUG** to **1**.

| Value     | Description                                      |
|-----------|--------------------------------------------------|
| <b>0</b>  | None (default setting)                           |
| <b>1</b>  | Fatal error                                      |
| <b>2</b>  | General error                                    |
| <b>3</b>  | Fix-up error                                     |
| <b>4</b>  | Warnings                                         |
| <b>5</b>  | Application errors                               |
| <b>6</b>  | Assumptions made by the nCipher PKCS #11 library |
| <b>7</b>  | API function calls                               |
| <b>8</b>  | API return values                                |
| <b>9</b>  | API function argument values                     |
| <b>10</b> | Details                                          |
| <b>11</b> | Mutex locking detail                             |

## CKNFAST\_DEBUGDIR

If this variable is set to the name of a writeable directory, log files are written to the specified directory. The name of each log file contains a process ID. This can make debugging easier for applications that fork a lot of child processes.

## CKNFAST\_DEBUGFILE

You can use this variable to write the output for **CKNFAST\_DEBUG** (**Path name > file name**).

## CKNFAST\_FAKE\_ACCELERATOR\_LOGIN

If this variable is set, the nCipher PKCS #11 library accepts a PIN for a module-protected key, as required by Sun Java Enterprise System (JES), but then discards it. This means that a Sun JES user requesting a certificate protected by a load-shared module can enter an arbitrary PIN and obtain the certificate.

## CKNFAST\_LOADSHARING

Load-sharing is determined by the state of the **CKNFAST\_LOADSHARING** environment variable.

To enable load-sharing mode, set the environment variable **CKNFAST\_LOADSHARING** to a value that starts with something other than **0**, **N**, or **n**. The virtual slot behavior then operates. When this variable is not set (or is set to a value that starts with **0**, **N**, or **n**), you see two slots for every module connected.

Applications that enable the user to select the slot, or that dynamically select a slot based on its capabilities, normally work without requiring the backward capability mode.

**Note** To use softcards with PKCS #11, you must have **CKNFAST\_LOADSHARING** set to a nonzero value. When using pre-loaded softcards or other objects, the PKCS #11 library automatically sets **CKNFAST\_LOADSHARING=1** (load-sharing mode on) unless it has been explicitly set to **0** (load-sharing mode off).

## CKNFAST\_NO\_ACCELERATOR\_SLOTS

If this variable is set, the nCipher PKCS #11 library does not create the accelerator slot, and thus the library only presents the smart card slots (real or virtual, depending on whether load-sharing is in use).

Do not set this environment variable if you want to use the accelerator slot to create or load module protected keys.

**Note** Setting this environment variable has no effect on **ckcheckinst** because **ckcheckinst** needs to list accelerator slots.

## CKNFAST\_NO\_SYMMETRIC

If this variable is set, the nCipher PKCS #11 library does not advertise any symmetric key operations.

## CKNFAST\_NO\_UNWRAP

If this variable is set, the nCipher PKCS #11 library does not advertise the **c\_wrap** and **c\_unwrap** commands. You should set this variable if you are using Sun Java Enterprise System (JES) or Netscape Certificate Management Server as it ensures that a standard SSL handshake is carried out. If this variable is not set, Sun JES or Netscape Certificate Management Server make extra calls, which reduces the speed of the library.

## CKNFAST\_NONREMOVABLE

When this environment variable is set, the state changes of the inserted card set are ignored by the nCipher PKCS #11 library.

**Note** Since protection by non-persistent cards is enforced by the HSM, not the library, this variable does not make it possible to use keys after a non-persistent card is removed, or after a timeout expires.

## CKNFAST\_NVRAM\_KEY\_STORAGE

When this environment variable is set, the PKCS #11 library generates only keys in nonvolatile memory (NVRAM). You must also ensure this environment variable is set in order to delete NVRAM-stored keys.

## CKNFAST\_OVERRIDE\_SECURITY\_ASSURANCES

This variable can be assigned one or more of the following parameters, with an associated value where appropriate, to override the specified security assurances in key operations where this is deemed acceptable:

- **all**
- **none**

- **tokenkeys**
- **longterm** [=days]
- **explicitness**
- **import**
- **unwrap\_mech**
- **unwrap\_kek**
- **derive\_kek**
- **derive\_xor**
- **derive\_concatenate**
- **weak\_algorithm**
- **shortkey\_algorithm=bitlength**
- **silent.**

Each parameter specified is separated by a semicolon. On the command line, enter the following to set the variable:

---

```
CKNFAST_OVERRIDE_SECURITY_ASSURANCES="token1;token2=value3"
```

---

In the configuration file, enter the following to set the variable:

---

```
CKNFAST_OVERRIDE_SECURITY_ASSURANCES=token1;token2=value3
```

---

Unknown parameters generate a warning; see [Diagnostic warnings about questionable operations](#) on page 186.

The meaning of these parameters is described in the rest of this section.

**all**

The **all** parameter overrides all security checks and has the same effect as supplying all the other **CKNFAST\_OVERRIDE\_SECURITY\_ASSURANCES** parameters except the **none** parameter. Using the **all** parameter prevents the library from performing any of the security checks and allows the library to perform potentially insecure operations. This parameter cannot be used with any other parameters.

## none

The **none** parameter does not override any of the security checks and has the same effect as supplying no parameters. Using the **none** parameter allows the library to perform all security checks and warn about potentially insecure operations without performing them. This parameter cannot be used with any other parameters.

## tokenkeys

The **tokenkeys** parameter permits applications to request that insecure keys are stored long-term by the cryptographic hardware and library.

Some PKCS #11 applications create short-term session keys as long-term objects in the cryptographic provider, for which strong protection by the module is not important. Therefore, provided that you intend to create long-term keys, the need to set this token does not always indicate a potential problem because the **longterm** keys restriction is triggered automatically. If you set the **tokenkeys** parameter, ensure that your Quality Assurance process tests all of your installation's functionality at least 48 hours after the system was set up to check that the key lifetimes are as expected.

When the **tokenkeys** parameter is set, the effect on the PKCS #11 library is to permit insecure Token keys. By default, any attempts to create, generate, or unwrap insecure keys with **CKA\_TOKEN=true** fails with **CKR\_TEMPLATE\_INCONSISTENT** and a log message that explains the insecurity. When **tokenkeys** is included as a parameter for **CKNFAST\_OVERRIDE\_SECURITY\_ASSURANCES**, attempts to create, generate, or unwrap insecure keys with **CKA\_TOKEN=true** are allowed.

## longterm[=*days*]

The **longterm** parameter permits an insecure key to be used for *days* after it was created. Usually insecure keys may not be used more than 48 hours after their creation. If *days* is not specified, there is no time limit.

**Note** A need to set this variable usually means that some important keys that should be protected by the module's security are not secure.

When the **longterm** parameter is set, the PKCS #11 API permits the use of the following functions with an insecure key up to the specified number of *days* after its creation:

- **C\_Sign** and **C\_SignUpdate**
- **C\_Verify** and **C\_VerifyUpdate**
- **C\_Encrypt** and **C\_EncryptUpdate**
- **C\_Decrypt** and **C\_DecryptUpdate**.

By default these functions fail with **CKR\_FUNCTION\_FAILED**, or **CKR\_KEY\_FUNCTION\_NOT\_PERMITTED**, and a log message that explains the insecurity of these functions when used with an insecure private or secret key more than 48 hours after the creation of the key as indicated by **time()** on the host.

When the **longterm** parameter is set, the functions **C\_SignInit**, **C\_VerifyInit**, **C\_EncryptInit**, and **C\_DecryptInit** check the **CKA\_CREATION\_DATE** against the current time.

## explicitness

The **explicitness** parameter permits applications to create insecure keys without explicitly recognizing that they are insecure by setting the flag which allows export as plain text. An insecure key is one whose plain text is available to an attacker on the host; thus it makes no sense to restrict legitimate users' access to the plain text of the key value.

**Note** A need to set the **explicitness** parameter does not necessarily indicate a problem, but does usually indicate that a review of the application's security policies and use of the PKCS #11 API should be carried out.

Unless the **explicitness** parameter is set, attempts to create, generate, or unwrap insecure keys with **CKA\_SENSITIVE=true**, or to set **CKA\_SENSITIVE=true** on an existing key, fail by default with **CKR\_TEMPLATE\_INCONSISTENT** and a log message explaining the insecurity. However, when the **explicitness** parameter is set, these operations are allowed.

## import

The **import** parameter allows keys that are to be imported into the module's protection from insecure external sources to be treated as secure, provided that the application requests security for them. Usually, the library treats imported keys as insecure for the purposes of checking the security policy of the application. Even though the imported copy may be secure, insecure copies of the key may still exist on the host and elsewhere.

If you are migrating from software storage to hardware protection of keys, you must enable the **import** parameter at the time of migration. You can disable **import** again after migrating the keys.

**Note** Setting this variable at any other time indicates that the library regards the key as secure, even though it is not always kept within a secure environment.

When the **import** parameter is set, the PKCS #11 API treats keys that are imported through **C\_CreateObject** or **C\_UnwrapKey** as secure (provided there is no other reason to treat them as insecure). By default, keys which are imported through **C\_CreateObject** or **C\_UnwrapKey** without this option in effect are marked as being insecure. Only the setting of the parameter at the time of import is relevant.

## unwrap\_mech

The **unwrap\_mech** parameter allows keys transferred into the module in an insecurely encrypted form to be treated as if the encryption had been secure. This parameter allows you to use key-decryption keys for insecure decryption mechanisms as well as for raw decryption.

There are no key decryption or wrapping mechanisms that are both secure and suitable for long keys. Set the **unwrap\_mech** parameter to use PKCS #11 **unwrap** to create keys that are treated as secure. Set the **unwrap\_mech** parameter at the time that the wrapping key is created or imported.

When the **unwrap\_mech** parameter is set, the PKCS #11 API adds the **CKA\_DECRYPT** permission on decryption, even if the template has **CKA\_DECRYPT=false**. By default, trying to create a key with **CKA\_UNWRAP=true** and **CKA\_DECRYPT=false** fails with **CKR\_TEMPLATE\_INCONSISTENT**. If **unwrap\_mech** is supplied as a parameter for **CKNFAST\_OVERRIDE\_SECURITY\_ASSURANCES**, then when the **CKA\_UNWRAP** permission is requested on a key, the library automatically adds the **CKA\_DECRYPT** permission, even if the template has **CKA\_DECRYPT** false, because abuse of the decryption mechanisms would allow a program to use the library to decrypt with the key.

## unwrap\_kek

When a key is transferred into the module in encrypted form, the key is usually treated as insecure unless the key that was used for the decryption only allows the import and export of keys and not the decryption of arbitrary messages. This behavior is necessary to prevent an unauthorized application from simply decrypting the encrypted key instead of importing it. However, because PKCS #11 wrapping mechanisms are insecure, all unwrapping keys have **CKA\_DECRYPT=true**.

By default, keys that are unwrapped with a key that has **CKA\_DECRYPT** permission are considered insecure. When the **unwrap\_kek** parameter is set, the PKCS #11 API considers keys that are unwrapped with a key that also has **CKA\_DECRYPT** permission as secure (provided there is no other reason to treat them as insecure).

## derive\_kek

By default, keys that have been derived by using **CKM\_DES3\_ECB\_ENCRYPT\_DATA** with a key that has **CKA\_ENCRYPT** permission are considered insecure. However, when the **derive\_kek** parameter is set, the PKCS #11 API considers keys that are derived with a key that has **CKA\_ENCRYPT** permission as secure (provided that there is no other reason to treat them as insecure).



## derive\_xor

Normally, you can only use only extractable keys with **CKM\_XOR\_BASE\_AND\_DATA** and, on unextractable keys, only **CKM\_DES3\_ECB\_ENCRYPT\_DATA** is allowed by **CKA\_DERIVE**. However, when the **derive\_xor** parameter is set, the PKCS #11 API also allows such functions with keys that are not extractable and treats them as secure (provided that there is no other reason to treat them as insecure).

## derive\_concatenate

Normally, you can only use session keys with **CKM\_CONCATENATE\_BASE\_AND\_KEY** for use with the operation **C\_DeriveKey**. However, when the **derive\_concatenate** parameter is set, the PKCS#11 API also allows such functions with keys that are long term (token) keys. The PKCS#11 API treats these keys as secure, provided there is no other reason to treat them as insecure. Even if the **all** parameter is set, if you do not include the **CKA\_ALLOWED\_MECHANISMS** with **CKM\_CONCATENATE\_BASE\_AND\_KEY**, this **C\_DeriveKey** operation will not be allowed.

## weak\_algorithm

The **weak\_algorithm** parameter allows you to treat keys used with a weak algorithm as secure. For example, DES is not secure, but setting the parameter **weak\_des** means that such keys are considered secure. You can apply the **weak\_algorithm** parameter to all keys that have a short fixed key length or whose algorithms have other security problems. As a guide, weak algorithms are those whose work factor to break is less than approximately 80 bits.

## shortkey\_algorithm=bitlength

The **shortkey\_algorithm=bitlength** parameter permits excessively short keys for the specified **algorithm** to be treated as secure. The parameter **bitlength** specifies the minimum length, in bits, that is to be considered secure. For example, RSA keys must usually be at least 1024 bits long in order to be treated as secure, but **shortkey\_rsa=768** would allow 768-bit RSA keys to be treated as secure.

## silent

The **silent** parameter turns off the warning output. Checks are still performed and still return failures correctly according to the other variables that are set.

## Diagnostic warnings about questionable operations

When the **CKNFAST\_OVERRIDE\_SECURITY\_ASSURANCES** environment variable is set to a value other than **all**, diagnostic messages are always generated for questionable operations. Each message contains the following elements:

- the PKCS #11 label of the key, if available
- the PKCS #11 identifier of the key, if available
- the hash of the key
- a summary of the problem.

If the problem is not that a questionable operation has been permitted because of a setting in **CKNFAST\_OVERRIDE\_SECURITY\_ASSURANCES** it could be that an operation has failed. In such a case, the setting required to authorize the operation is noted.

By default, these messages are sent to **stderr**. If a file name has been specified in the **CKNFAST\_ASSURANCE\_LOG** environment variable, diagnostic messages are also written to this file.

If **CKNFAST\_DEBUG** is **1** or greater and a file is specified in **CKNFAST\_DEBUGFILE**, the PKCS #11 library Security Assurance Mechanism log information is sent to the specified file. These variables must be set whenever **generatekey** or **KeySafe** are used.

**Note** If a file is specified in **CKNFAST\_ASSURANCES\_LOG** and no file is specified in **CKNFAST\_DEBUGFILE** (or if **CKNFAST\_DEBUG** is **0**), diagnostic messages are sent to **stderr** as well as to the file specified in **CKNFAST\_ASSURANCES\_LOG**.

## CKNFAST\_SESSION\_THREADSafe

You must set this environment variable to **yes** if you are using the Sun PKCS #11 provider when running nCipherKM JCA/JCE code.

## CKNFAST\_TOKENS\_PERSISTENT

This variable controls whether or not the Operator Cards that are created by your PKCS #11 application are persistent. If this variable is set when your application calls the PKCS #11 function that creates tokens, the Operator Card created is persistent.

**Note** Use of the nCipher PKCS #11 library to create tokens is deprecated, because it can only create 1/1 tokens in FIPS 140-2 level 2 security worlds. Use **KeySafe** or one of the command-line utilities to create OCSs.

## CKNFAST\_USE\_THREAD\_UPCALLS

If this variable is set and **CKF\_OS\_LOCKING\_OK** is passed to **C\_Initialize**, **NFastApp\_SetThreadUpcalls** is called by means of **nfast\_usencthread**s and only a single **NFastApp\_Connection** is used, shared between all threads.

If this variable is set and mutex callbacks are passed to **C\_Initialize** but **CKF\_OS\_LOCKING\_OK** is not passed, **C\_Initialize** fails with **CKR\_FUNCTION\_FAILED**. (**NFastApp\_SetThreadUpcalls** requires more callbacks than just the mutex ones that PKCS #11 supports.)

If neither mutex callbacks nor **CKF\_OS\_LOCKING\_OK** is passed, this variable is ignored. Only a single connection is used because the application must be single threaded in this case.

## CKNFAST\_LOAD\_KEYS

This variable will load private objects at **C\_Login** time, rather than at the first cryptographic operation.

## CKNFAST\_WRITE\_PROTECTED

Set this variable to make your OCS or softcard (token) write-protected. If a token is write-protected, you cannot:

- Generate certificate, data, and key objects for that token.
- Modify attributes of an existing object.

**Note** This environment variable does not prevent you from deleting an object from your token.

## Checking the installation of the nCipher PKCS #11 library

After you have created a security world, ensure that the nCipher PKCS #11 library has been successfully installed by using the **ckcheckinst** command-line utility. In a FIPS 140-2 level 3 compliant Security World, you need an Operator Card from the Security World to run this utility. If you did not install your Security World in FIPS 140-2 level 3 mode, you can run the utility by using either an Operator Card or a fixed token.

To verify the installation of the nCipher PKCS #11 library, follow these steps:

# 1 Give the command **ckcheckinst**.

If you have an invalid security world (for example, if all your modules are in the initialization state), **ckcheckinst** quits with the following error message:

---

```
ckcheckinst: C_Initialize failed rv = 00000006
Is the security world initialized? (Use nfkminfo to check)
```

---

If your Security World is valid, **ckcheckinst** displays information similar to the following:

---

```
PKCS#11 library interface version 2.01
 flags 0
 manufacturerID "nCipher Corp. Ltd"
 libraryDescription "nFast PKCS#11 1.1.14"
 implementation version 1.1.14
 Strictfips 140 enabled
 Load sharing and Failover enabled

slot Status Label
=====
1 Operator card "card2"
2 Operator card "card3"
Select slot Number to run library test or 'R'etry or to 'E'xit:
```

---

In this example output:

- **PKCS#11 library interface version 2.01** refers to the version of the PKCS #11 specification supported
- **implementation version 1.1.14** refers to the version of the nCipher PKCS #11 library
- **Loadsharing and Failover enabled** is shown if load-sharing has been enabled.

Slots that contain a valid Operator Card are indicated by the status **Operator card** and the card's label. In a FIPS 140-2 level 2 compliant security world, the available fixed tokens are also listed for selection.

If you insert a blank card or an unrecognized card (for example, an Operator Card from a different Security World or an Administrator Card), this is indicated in the **Status** column. The corresponding slot number is not available.

**Note** If you are using the **preload** command-line utility in conjunction with the nCipher PKCS #11 library, you can only see the token that you loaded with the **preload** utility. In load-sharing mode, the loaded card set is used to set the environment variable **CKNFAST\_CARDSET\_HASH**, so only this card set is visible as a slot.

If you have no available slots, **ckcheckinst** displays **No token present** beside the relevant slot numbers.

**ckcheckinst** gives you the following choices:

---

```
No removable tokens present.
Please insert an operator card into at least one available slot and
enter 'R' retry.
If you have not created a an operator card, enter a fixed token slot
number (if available)
Or 'E' to exit this program and create a card set before continuing.
```

---

2 If there are no available slots with cards in them, you can choose one of the following actions:

- insert a valid Operator Card, and press **␣**
- choose a fixed token slot
- press **␣** to quit, then create an OCS, and run **ckcheckinst** again.

When there is at least one slot with a valid token, input a slot number, and press **␣**. In a FIPS 140-2 level 3 compliant Security World, **ckcheckinst** prompts you to enter the pass phrase for the selected Operator Card.

3 Type the pass phrase, and press **␣**.

**ckcheckinst** displays the results of the tests:

---

| Test                            | Pass/Failed |
|---------------------------------|-------------|
| ----                            | -----       |
| 1 Generate RSA key pair         | Pass        |
| 2 Generate DSA key pair         | Pass        |
| 3 Encryption/Decryption         | Pass        |
| 4 Signing/Verify                | Pass        |
| Deleted test keys               | ok          |
| PKCS11 Library test successful. |             |

---

If any tests fail, **ckcheckinst** displays a message indicating the failure and quits. It does not run any subsequent tests.

If **ckcheckinst** fails:

- check that the hardserver is running
- use the **enquiry** and **nfkminfo** world.

If all seems in order, reinstall the nCipher library.

## How the nCipher PKCS #11 library protects keys

Session objects are created on a module and never leave that module. The following table lists the protection for different types of PKCS #11 token objects:

|                      | Smart card Slot       | Accelerator Slot      |
|----------------------|-----------------------|-----------------------|
| Private Token Object | Operator Card Set     | not supported         |
| Public Token Object  | security world        | security world        |
| Public key           | well known module key | well known module key |

### Operator Card Set

The object is stored as an nCipher key blob encrypted by the OCS key. You must log in to this OCS before you can load this object.

### security world

The object is stored as an nCipher key blob encrypted by the security world key. This object can be loaded on to any module in the security world. The nCipher PKCS #11 library only allows access if a card from this OCS is present.

### well-known module key

Public keys are encrypted under a well-known module key. This encryption is for programming convenience only and does not provide security. These keys can be loaded on any nShield module.

## Restrictions on function calls in load-sharing mode

The following function calls are not supported in load-sharing mode:

- **C\_LoginBegin** (Security World Software-specific call to support *K/N* card sets)
- **C\_LoginNext** (Security World Software-specific call to support *K/N* card sets)
- **C\_LoginEnd** (Security World Software-specific call to support *K/N* card sets).

The following function calls are supported in load-sharing mode *only* when using softcards:

- **C\_InitToken**
- **C\_InitPin**

- **C\_SetPin.**

**Note** To use **C\_InitToken**, **C\_InitPin**, or **C\_SetPin** in load-sharing mode, you must have created a softcard with the command **ppmk -n** before selecting the corresponding slot.

**Note** The **C\_InitToken** function is *not* supported for use in non-load-sharing FIPS 140-2 level 3 Security Worlds.

## Vendor specific error codes

Security World Software defines the following vendor specific error codes:

### **CKR\_FIPS\_TOKEN\_NOT\_PRESENT**

This error code indicates that an Operator Card is required even though the card slot is not in use.

### **CKR\_FIPS\_MECHANISM\_INVALID**

This error code indicates that the current mechanism is not allowed in FIPS 140-2 level 3 mode.

### **CKR\_FIPS\_FUNCTION\_NOT\_SUPPORTED**

This error code indicates that the function is not supported in FIPS 140-2 level 3 mode (although it is supported in FIPS 140-2 level 2 mode).

## nShield native and custom applications

Use the **nShield native** option for applications that were written using nShield key management software and that expect keys to be both protected by the Security World and stored in the Security World data structure.

Use the **custom** external application option for applications that were written using nShield key management software and that expect their keys to be in standalone files.

**Note** KeySafe does not place any restrictions on the OCS that is used to protect nShield native or **custom** application keys. You must make sure that your application is capable of loading the card set.

## CodeSafe applications

If you have enabled the Secure Execution Engine (SEE), your system can run CodeSafe applications that implement special functionality.

**Note** If you wish to use the SEE to run applications, it must have been ordered and enabled as described in [Enabling optional features on the module](#) on page 60.

An SEE application is typically a standalone SEE machine that is loaded automatically by the **hardserver** (for example, a CodeSafe C application).

Check the documentation that your application vendor supplies for information about any signatures that you may require to set up and use the application, as well as for any other installation and configuration information.

CodeSafe applications are standalone applications, but each CodeSafe C application can consist of multiple parts, and its installation can include several configuration steps. For instructions on installing and configuring each application, see your application vendor's documentation.

You may need to use the **hardserver**, **loadmache**, and **tct2** utilities when configuring and loading an application; see the *CodeSafe Developer Guide* for more information.

**Note** To use **see-sock-serv** directly, you must select **BSDlib sockserv**.





## Chapter 8: Remote Operator Card Sets

This chapter explains:

- the concept of Remote Operator
- how to configure Remote Operator.

**Note** If you wish to use the Remote Operator feature, you must have enabled it as described in [Enabling optional features on the module](#) on page 60. The Remote Operator feature must have been ordered for, and enabled on, the nShield module that you intend to use as the remote, unattended module.

### About Remote Operator

The Remote Operator feature enables the contents of a smart card inserted into the slot of one module (the *attended module*, such as a client module) to be securely transmitted and loaded onto another module (an *unattended module*, such as the nShield Connect or netHSM). This is useful when you need to load an OCS-protected key onto a machine to which you do not have physical access (because, for example, it is in a secure area).

For Remote Operator to work, the modules must be in the same Security World. You insert the required cards from the OCS into a slot in the attended module. From this module, the contents of the OCS are transmitted over secure channels to the unattended module, which then loads them. You do not need physical access to the unattended module in order to load the OCS onto it.

The following limitations apply to Remote Operator:

- you cannot access non-persistent card sets remotely
- you cannot use the **createocs** command-line utility to write new cards or card sets remotely.

You can export a slot from an attended module and import a slot to any (unattended) module in the Security World. The unattended module for import may be a local module connected to the host or a network-connected module such as the nShield Connect or netHSM. (You can import or export slots between the nShield Connect or netHSM and local modules only if the local module uses firmware version 2.6.10 or higher.) Before you can import a slot to one module, you must first export it from another module.

# Configuring Remote Operator

This section explains how to configure Remote Operator.

## Overview of configuring Remote Operator

Before you can use Remote Operator, you must perform the following initial configuration tasks:

- 1 Configure the modules for Remote Operator.

The modules must be in the same Security World, must have firmware version 2.6.10 or greater, and must have been initialized with remote card set reading enabled.

Both the attended and the unattended module must be in operational mode before they can import or export slots. For more information about changing module modes, see [Checking and changing module mode](#) on page 302.

- 2 Configure the module hardservers on their respective host machines for slot import and export, as appropriate.

After the initial configuration is complete, to use Remote Operator you must:

- 1 Create a Remote OCS (that is, an OCS with the correct permissions for Remote Operator).
- 2 Generate keys that are protected by the Remote OCS.
- 3 Ensure your application is configured to use keys protected by the Remote OCS.

## Configuring modules for Remote Operator

- 1 Ensure both modules are initialized into the same Security World; see [Adding or restoring a module to the Security World](#) on page 94.

**Note** By default, modules are initialized with remote card-set reading enabled. If you do not want a module to be able to read remote card sets, you can initialize it by running the `new-world` with the `-S MODULE` (where **MODULE** is the module's ID number).

- 2 Check whether firmware version 2.6.10 or greater is installed in both the attended and unattended modules by running the **enquiry** command-line utility.

If an earlier version of firmware is installed, you must upgrade the module firmware as necessary; see Appendix J: [Upgrading firmware](#).

- If you upgrade the attended module's firmware, you must thereafter reinitialize it into the Security World (by using KeySafe or the **new-world** command-line utility).
- If you upgrade the unattended module's firmware, you do not need to reinitialize it afterwards.

- 3 For the unattended module:

- a Check whether the Remote Operator feature is enabled by running the **enquiry** command-line utility. The output for the module must include **Remote Share** in its list of Features.
- b Check whether the correct software, with permission to receive remote shares, is present by running the **nfkminfo** command-line utility. The output for the module must show that **flags** are set to include **ShareTarget**, as in the following example:

---

```
Module #1
generation 2
state 0x2 Usable
flags 0x10000 ShareTarget
n_slots 3
esn 8851-43DF-3795
hkml 391eb12cf98c112094c1d3ca06c54bfe3c07a103
```

---

## Configuring hardservers for Remote Operator

Before you can configure hardservers for Remote Operator, ensure that you have configured the attended and unattended modules for Remote Operator as described in [Configuring modules for Remote Operator](#) on page 194.

When the modules have been configured, follow these steps to configure the hardserver:

- 1 On the attended module's host machine, configure the hardserver to allow slot 0 of the local module (with ESN *AAAA-AAAA-AAAA* to be exported to a remote module (with ESN *BBBB-BBBB-BBBB*, hosted by the machine with the IP address *222.222.222.222*). Follow these steps:

- a Edit the **slot\_exports** section of the hardserver configuration file by adding the lines of the form:

---

```
local_esn=AAAA-AAAA-AAAA local_slotid=0 remote_ip=222.222.222.222 remote_esn=BBBB-BBBB-BBBB
```

---

**Note** For more information about the parameters controlled by the **slot\_exports** section of the hardserver configuration file, see [slot\\_exports](#) on page 287.

- b Run the **cfg-reread** command-line utility to prompt the hardserver to read the configuration changes.
- 2 On the unattended module's host machine, configure the hardserver to import slot 0 from the remote attended module (with ESN *AAAA-AAAA-AAAA*, hosted by the machine with the IP address *111.111.111.111*) to the local module (with ESN *BBBB-BBB-BBBB*):

- a Edit the **slot\_imports** section of the hardserver configuration file by adding the lines of the form:

---

```
local_esn=BBBB-BBBB-BBBB local_slotid=2 remote_ip=111.111.111.111 remote_esn=AAAA-AAAA-AAAA remote_slotid=0
```

---

This example assigns the imported slot to ID 2.

**Note** For more information about the parameters controlled by the **slot\_imports** section of the hardserver configuration file, see [slot\\_imports](#) on page 286.

- b Run the **cfg-reread** command-line utility to prompt the hardserver to read the configuration changes.

You can check that the slot was imported successfully by, on the unattended machine, running the command:

---

```
slotinfo -m 1
```

---

If slot importation was successful, the output from this command includes the line:

| Slot | Type         | Token   | IC | Flags | Details |
|------|--------------|---------|----|-------|---------|
| #0   | Smartcard    | present | 3  | A     |         |
| #1   | Software Tkn | -       | 0  |       |         |
| #2   | smartcard    | -       | 0  | AR    |         |

The **R** in the **Flags** column indicates that slot **#2** is a remote slot.

Applications running on the unattended machine can now use slot **#2** to load OCSs that are presented to slot **#0** on the attended machine. If any of the cards require a pass phrase, the application must pass this to the unattended module in the usual way.

For the application to be able to load the OCS onto the unattended module, it must be able to read the card set files associated with the OCS from the local Key Management Data directory. If the OCS was created on a different machine, you must copy the card set files in the Key Management Data directory onto the unattended machine (either manually or by using client cooperation; for more information, see [Setting up client cooperation](#) on page 51).

The same applies for any keys that an application on an unattended module needs to load but that were not generated on that machine.

## Creating OCSs and keys for Remote Operator

When you have configured the modules and hardservers for Remote Operator, you can create Remote OCSs and generate keys protected by them. These Remote OCSs and keys can be used by applications running on the unattended module.

For the most part, card sets and keys intended to be used with Remote Operator are similar to their ordinary, non-Remote counterparts.

### Creating OCSs for use with Remote Operator

You can generate Remote OCSs by using KeySafe or by running the **createocs** command-line utility with the **-q|--remotely\_readable** option specified. The cards in a Remote OCS must be created as persistent; see [Persistent Operator Card Sets](#) on page 116.

To check whether the card in a slot is from a Remote OCS, run the **nfkminfo** command-line utility. The output displays slot section information similar to the following:

---

```
Module #1 Slot #0 IC 1
generation 1
phystype SmartCard
slotlistflags 0x2
state 0x5
Operator flags 0x20000 RemoteEnabled
shareno 1
shares LTU (Remote)
error OK
```

---

In this example output, the **RemoteEnabled** flag indicates the card in the slot is from a Remote OCS.

**Note** A module must have firmware version 2.0.2 or greater in order to create Remote OCSs.

**Note** If you create a Remote OCS on the attended machine, then you must copy the Key Management Data files on the attended machine to the unattended machine.

**Note** Both the attended and unattended modules must be in the same Security World before you generate a Remote OCS. If you are not using client cooperation, the Key Management Data directories must be manually synchronized after you generate the Remote OCS.

**Note** If you already have recoverable keys protected by a non-Remote OCS, you can transfer them to a new Remote OCS by using KeySafe or the **replaceocs** command-line utility.

## Loading Remote Operator Card Sets

After the hardserver has been configured, the remote slots can be used by all the standard nShield libraries. A remote slot can be used to load any OCSs that have been created to allow remote loading. For more information about the applications to use with remote cards, see Chapter 7: [Application interfaces](#). For more information about remote slots, see Chapter 8: [Remote Operator Card Sets](#).

**Note** After an OCS has been inserted into a remote slot, for each time a given card is inserted, the module only allows each share on that card to be read one time. If there is a second attempt to read shares from that card before the card is reinserted, the operation fails with a **UseLimitsUnavailable** error.

## Generating keys for use with Remote Operator

After you have created a Remote OCS, to generate keys protected by it you can run **KeySafe** or the **generatekey** and **preload** command-line utilities on the unattended module, inserting cards to the slot attached to the attended module. For more information about generating and working with keys, see Chapter 9: [Working with keys](#).

**Note** If you generate keys protected by a Remote OCS on the attended module, then you must copy the files in the Key Management Data directory on the attended machine to the unattended module.

**Note** **KeySafe** can list imported slots, but cannot use them.

If you already have an OCS-protected key that you want to use, but the protecting OCS is not a Remote OCS, you can use **KeySafe** to protect the key under a new Remote OCS if the key was originally generated with the key recovery option enabled.

However, if the key was not generated with key recovery enabled, you cannot protect it under a different OCS. In such a case, you must generate a new key to be protected by a Remote OCS.

## Configuring the application

After you have configured the modules and hardservers for Remote Operator, created a Remote OCS, and generated keys protected by the Remote OCS, configure the application with which you want to use these keys as appropriate for the particular application.

After you have configured the application, start it remotely from the attended machine. Insert cards from the OCS into the attended machine's exported slot as prompted.



## Chapter 9: Working with keys

This chapter explains how to use the facilities we provide to work with keys. There is often more than one way of performing a particular task. The methods available for working with keys are:

- `KeySafe`
- `generatekey` and related utilities

### Generating keys

Whenever possible, generate a new key instead of importing an existing key. Because existing keys have been stored in a known format on your hard disk, there is a risk that the existing key has been compromised. Key material can also persist on backup media.

**Note** Some applications can generate keys directly.

When you attempt to generate keys for a Security World that complies with FIPS 140-2 level 3, you are prompted to insert an Administrator Card or Operator Card. You may need to specify to the application, the slot you are going to use to insert the card. You need to insert the card only once in a session.

**Note** For softcard protected key generation, you must use an Operator Card Set.

Generating a key creates both a key and a certificate request for the following application types:

- `embed` (OpenSSL)
- `kpm`
- `seessl`

These requests are generated in PKCS #10 format with base-64 encoding.



## Generating keys on the command line

Keys are generated on the command line with the **generatekey** utility. The **--generate** option creates a new key on the host computer that is protected either by the module or by an Operator Card set from the Security World. No key material is stored in an unencrypted form on the host computer.

When you generate a key with **generatekey**, choose a new identifier for the key and use whichever application type is appropriate. The key identifier can only contain digits and lowercase letters; it cannot contain spaces, underscores (`_`), or hyphens (`-`).

You can use **generatekey** in two ways:

- in interactive mode, by issuing commands without parameters and supplying the required information when prompted by the utility
- in batch mode, by supplying some or all of the required parameters on the command line (**generatekey** prompts interactively for any missing but required parameters).

In interactive mode, you can input **abort** at any prompt to terminate the process.

Batch mode is useful for scripting. In batch mode, if any required parameters are omitted, **generatekey** does not prompt for the missing information but instead will either use available defaults or fail. If you specify one or more parameters incorrectly, an error is displayed and the command fails.

To generate a key, use the command:

---

```
generatekey --generate [OPTIONS] APPNAME [NAME=VALUE ...]
```

---

In this command:

- **--generate** option specifies that this instance of **generatekey** is generating a key. Other options can be specified to perform tasks such as importing or retargeting keys. To see a list of options run the command **generatekey --help**.
- the **APPNAME** parameter specifies the name of the application for which the key is to be generated. For details of the available application types (**APPNAME**), see [Key application type \(APPNAME\)](#) on page 295 in Appendix H: [Key generation options and parameters](#).
- The **NAME=VALUE** syntax is used to specify the properties of the key being generated. For details of the available application types (**APPNAME**), see [Key properties \(NAME=VALUE\)](#) on page 296 in Appendix H: [Key generation options and parameters](#)

For details of the available application types (**APPNAME**) and parameters that control other key properties (**NAME=VALUE**), see Appendix H: [Key generation options and parameters](#)

In interactive mode, **generatekey** prompts you for any required parameters or actions that have not been included in the command. When you give the command:

- 1 Enter parameters for the command, as requested. If you enter a parameter incorrectly, the request for that information is repeated and you can re-enter the parameter.
- 2 For **jcecs** keys, you must input the key store password. In interactive mode, this is not echoed to the screen. When all the parameters have been collected, **generatekey** displays the final settings. In a FIPS 140-2 level 3 compliant Security World, you are prompted to insert a card for FIPS authorization if no such card is present.
- 3 If prompted, insert an Administrator Card or an Operator Card from the current Security World.
- 4 If you want to protect the key with an OCS, you are prompted to insert the relevant cards and input pass phrases, as required.

## Example of key generation with generatekey

To generate in batch mode an RSA key protected by an OCS named **operatorone** and for an application that uses the Cryptographic Hardware Interface Library (CHIL), use the command:

---

```
generatekey --generate --batch --cardset=operatorone hwcrhk protect=token type=rsa size=1024 plainname=keya
ident=abc
```

---

The **generatekey** utility prompts you to insert a quorum of Operator Cards from the **operatorone** OCS. After you have inserted the appropriate number of cards, **generatekey** generates the key.

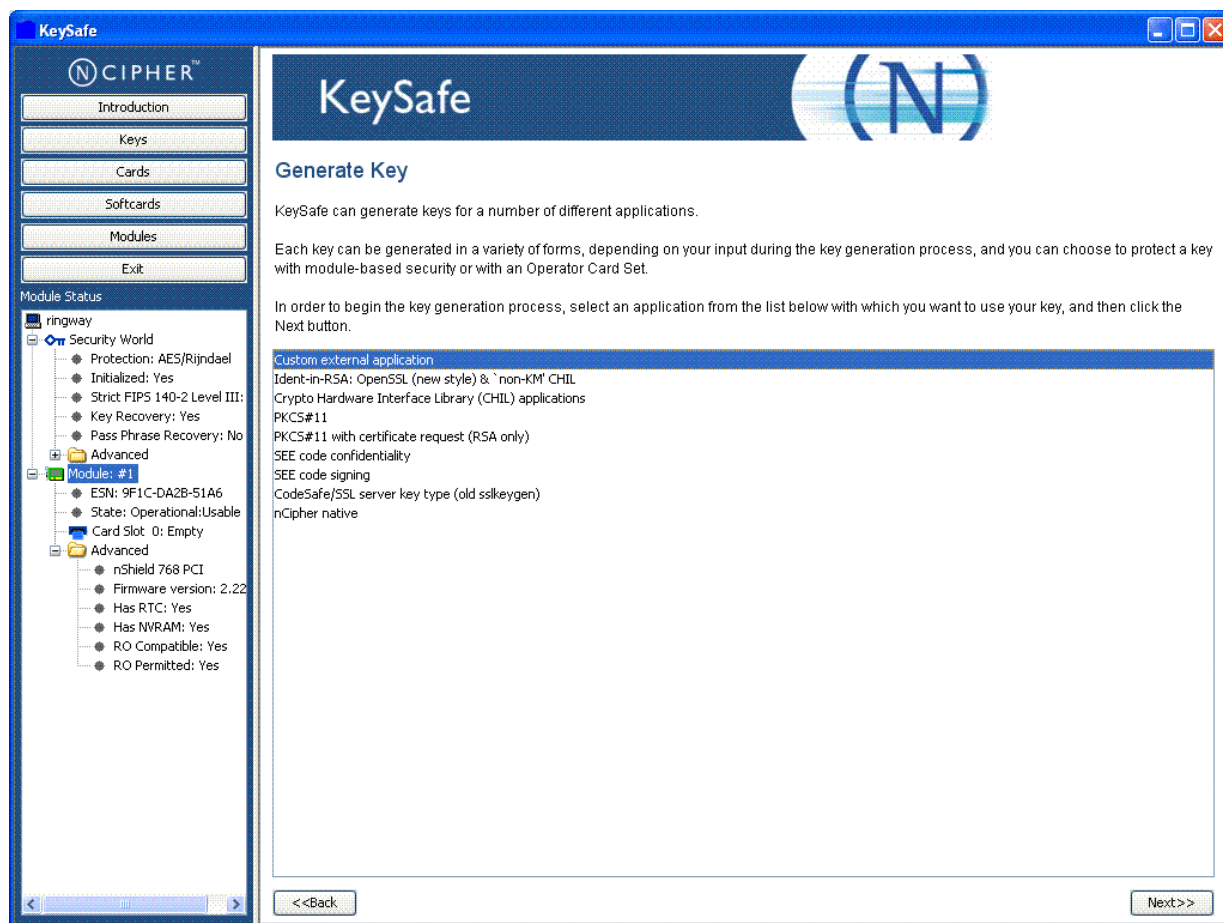
Although it is not explicitly specified, the created key is recoverable by default if OCS and softcard replacement is enabled for the Security World.

## Generating keys with KeySafe

In order to generate a key with KeySafe, follow these steps:

- 1 Start KeySafe. (For an introduction to KeySafe and for information on starting the software, see Appendix A: [Using KeySafe](#).)
- 2 Click the **Keys** menu button. KeySafe takes you to the **Key Operations** panel.

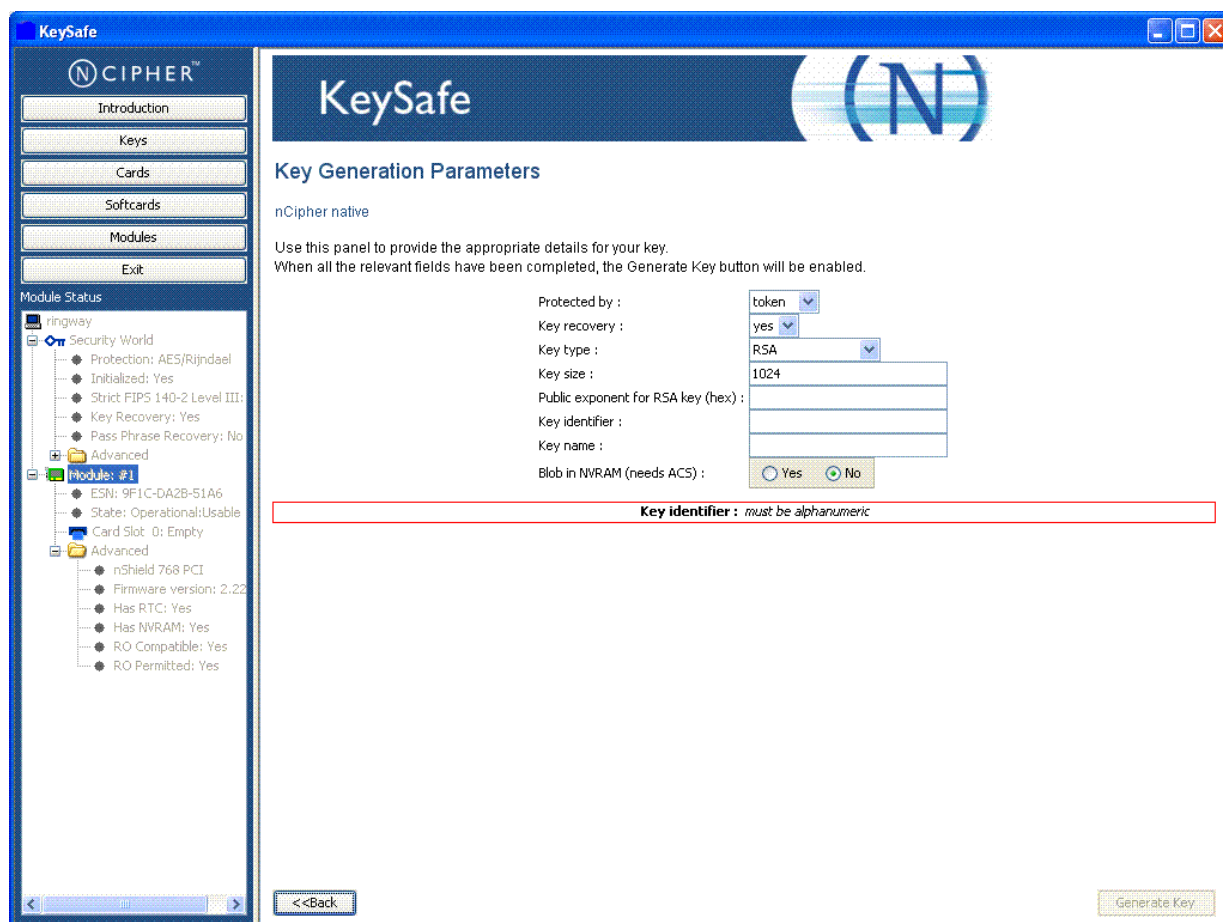
- 3 Click the **Generate Key** navigation button. KeySafe takes you to the **Generate Key** panel:



- 4 Select an application with which you want to use your key from the list, and then click the **Next** button. KeySafe takes you to the **Key Generation Parameters** panel.

- 5 Select and enter your desired parameters for key generation.

The types of information that you need to provide on the **Key Generation Parameters** panel differs slightly depending on the application you selected on the **Generate Key** panel. Below, as an example, is the **Key Generation Parameters** panel associated with nShield native applications:



- 6 When you have supplied your desired key generation parameters, click the **Next** button.

**Note** In order to generate a key protected by a FIPS 140-2 level 3 compliant Security World, you need authorization from an Operator Card or Administrator Card from the current Security World. Follow the onscreen instructions.

- 7 If you choose to generate a key that is protected by a smart card or softcard, KeySafe takes you to a panel from which you can load the protecting card or softcard. Follow the onscreen instructions, inserting any necessary Operator Cards and supplying any pass phrases as needed.
- 8 KeySafe displays a message indicating that the key has been successfully generated. Click the **OK** button.

- 9 KeySafe returns you to the Generate Key panel, from which you can generate another key or choose another operation.

## Generating NVRAM-stored keys

NVRAM key storage provides a mechanism for generating keys stored in a module's nonvolatile memory and hence within the physical boundary of an nShield module. You can store only a few keys in this way: the number depends on the memory capacity of the module, the size of the key and whether the key has recovery data associated with it.



We recommend that you *do not store keys in NVRAM unless you must do so to satisfy regulatory requirements*. NVRAM key storage was introduced only for users who must store keys within the physical boundary of a module to comply with regulatory requirements. NVRAM-stored keys provide no additional security benefits and their use exposes your ACS to increased risk. Storing keys in nonvolatile memory also reduces load-balancing and recovery capabilities. Because of these factors, we recommend you always use standard Security World keys unless explicitly required to use NVRAM-stored keys.

Before you can generate an NVRAM-stored key you must:

- ensure that you have installed the latest version of the Security World Software server software. You can use the **enquiry** command-line utility to identify the installed version number.
- ensure that your module has the latest version of firmware installed. See Appendix J: [Upgrading firmware](#) for information on upgrading your module firmware.
- create a new Security World.

When you generate an NVRAM-stored key, you must have sufficient nonvolatile memory available in the module or the command fails.



You need backup and recovery procedures to protect your NVRAM-stored keys.

**Note** An NVRAM-stored key can only be loaded successfully by using the **with-nfast** command-line utility on the generating module. Attempts to load such a key on other modules that have NVRAM fail with **UnknownID** errors.

We provide the **nvr-~~am~~-backup** utility to enable the copying of files, including NVRAM-stored keys, between a module's nonvolatile memory and a smart card.

## Importing keys

Importing a key takes an unprotected key stored on the host and stores it in the Security World in encrypted form.



We recommend generating a new key (or retargeting a key from within the Security World) instead of importing an existing key whenever possible. The import operation does not delete any copies of the key material from the host, and because existing keys have been stored in a known format on your hard disk (and key material can persist on backup media), there is a risk that an existing key has been compromised. It is your responsibility to ensure any unprotected key material is deleted. If a key was compromised before importation, then importing it does not make it secure again.

The following key types can be imported by the tools we provide:

- RSA keys in PEM-encoded PKCS #1 format (from a file). The PEM key that contains the key to import must not require a pass phrase.
- DES, DES2 and Triple DES keys (entered in hex).

**Note** You cannot import keys into a Security World that complies with FIPS 140-2 level 3. Attempting to import keys into a strict FIPS 140-2 Level 3 Security World returns an error.

This request is a PKCS #10 format request in base-64 encoding.

## Importing keys from the command line

You can import keys using the **generatekey** command-line utility. To import a key, give the command:

---

```
generatekey --import [OPTIONS] APPNAME [NAME=VALUE ...]
```

---

This command uses the following options:

| Option            | Description                                                                                                                                                        |
|-------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>--import</b>   | This option specifies key importation.                                                                                                                             |
| <b>OPTIONS</b>    | You can specify particular options when running <b>generatekey</b> that control details of key importation.                                                        |
| <b>APPNAME</b>    | This option specifies the name of the application for which the key is to be imported. This must be an application for which <b>generatekey</b> can generate keys. |
| <b>NAME=VALUE</b> | This specifies a list of parameters for the application.                                                                                                           |

For RSA keys, you can include **pemreadfile=filename** on the command line to specify the file name of the PEM file that contains the key. Otherwise, you are prompted for this information during import.

In interactive mode, you are prompted for any required parameters or actions that have not been included in the command:

- Enter parameters, as requested. If you enter a parameter incorrectly, the request for that information is repeated and you can re-enter the parameter.
- If you want to protect the key with an OCS, you are prompted to insert the relevant cards and input pass phrases, as required.
- If prompted, insert an Administrator Card or an Operator Card from the current Security World.

## Example of key importation with generatekey

To import an RSA key stored in **/opt/projects/key.pem** for use with an nShield native application and protect it with the Security World, use the command:

---

```
generatekey --import simple pemreadfile=/opt/projects/key.pem plainname=importedkey ident=abc
protect=module
```

---

In this example, **generatekey** requires you to input **RSA** for the key type.

Although not explicitly specified, this key is, by default, recoverable if OCS and softcard replacement is enabled for the Security World.

## Importing keys with KeySafe

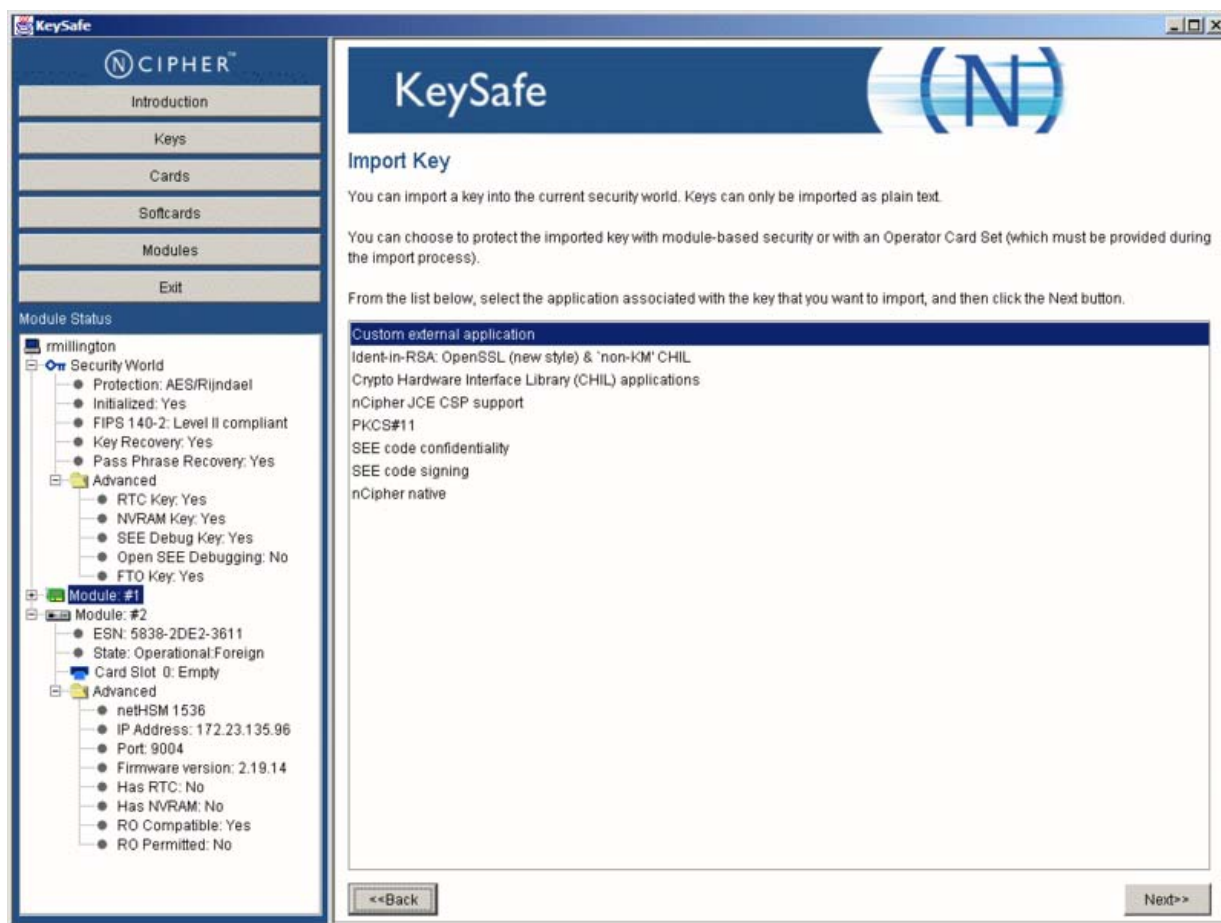
Any user who has write access to the directory that contains the Security World can import a key.

In order to import a key with KeySafe, follow these steps:

- 1 Start KeySafe. (For an introduction to KeySafe and for information on starting the software, see Appendix A: [Using KeySafe](#).)
- 2 Click the **Keys** menu button. KeySafe takes you to the **Key Operations** panel.



- 3 Click the **Import Key** navigation button. KeySafe takes you to the **Import Key** panel:

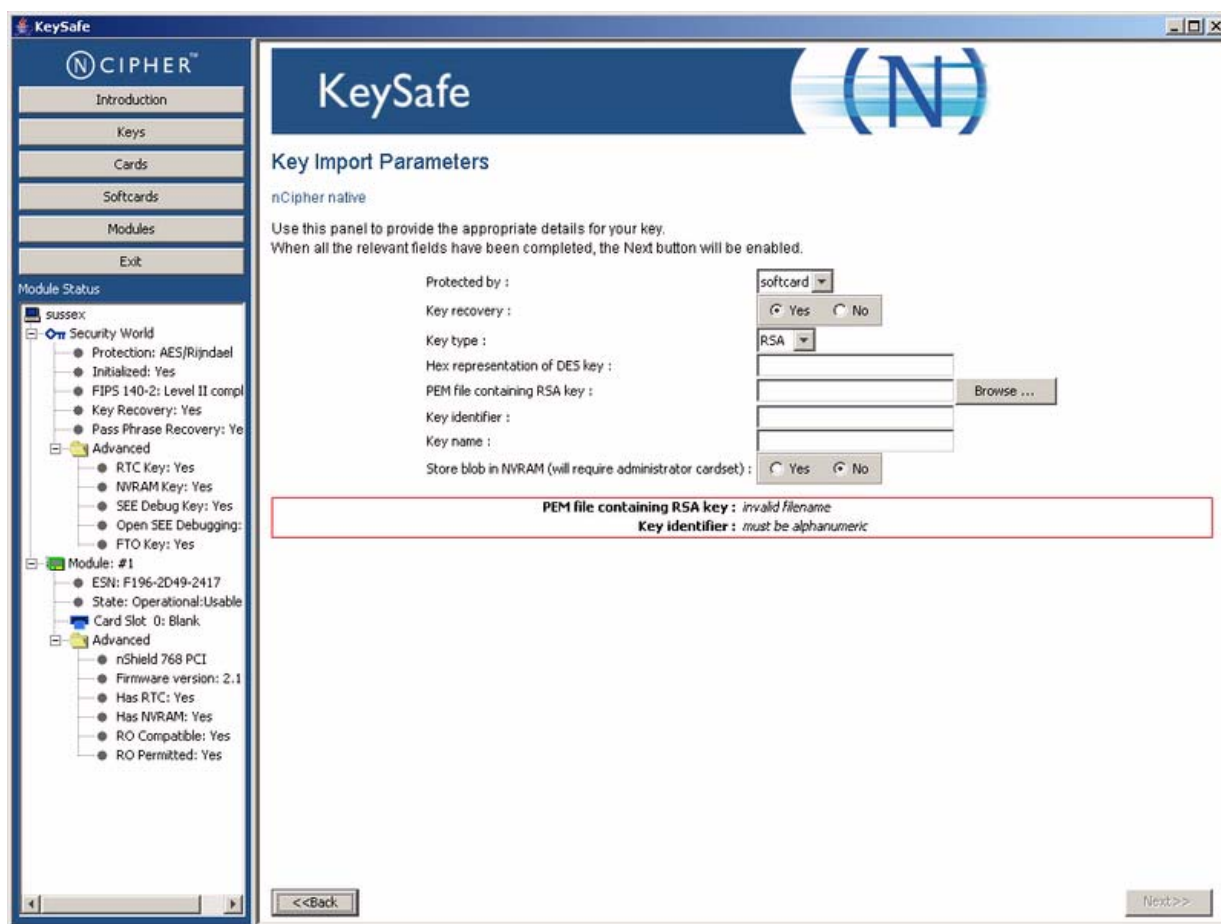


- 4 Select the application associated with the key that you want to import, and then click the **Next** button. KeySafe takes you to the **Key Import Parameters** panel.



- 5 Select and enter the desired parameters for the key that you want to import.

The types of information that you need to provide on the **Key Import Parameters** panel will differ slightly depending on the application you selected on the **Import Key** panel. Below, as an example, is the **Key Import Parameters** panel associated with nShield native applications:



- 6 When you have supplied parameters for the key that you want to import, click the **Next** button.
- 7 If you choose to import a key that is protected by a smart card, KeySafe takes you to the **Load Operator Card Set** panel. Follow the onscreen instructions, inserting the required number of Operator Cards and supplying any pass phrases as needed.
- 8 KeySafe displays a message indicating that the key has been successfully imported. Click the **OK** button.
- 9 KeySafe returns you to the **Import Key** panel, from which you can import another key or choose another operation.

## Listing supported applications with generatekey

To list supported applications, use the command:

---

```
generatekey --list-apps
```

---

## Retargeting keys with generatekey

The **--retarget** option takes an existing key in the Security World and makes it available for use by another application as if it had been expressly generated for use by that application. Because no key material is exposed during retargeting, this operation is as secure as generating a new key.

**Note** When you retarget a key, **generatekey** does not remove the original key from the Security World. If required, you can use KeySafe to discard the original key.

When you retarget a key, you cannot change its protection method. You cannot change the key from module-protected to card-protected, or from card-protected to module-protected.

To retarget a key, use the command:

---

```
generatekey --retarget [OPTIONS] APPNAME [from-application=appname]
[from-ident=keyident]
```

---

In this command:

| Option                          | Description                                                                                                                                                         |
|---------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>--retarget</b>               | This option specifies key importation.                                                                                                                              |
| <b>OPTIONS</b>                  | This option specifies any options to include when the command is run. Run the command <b>generatekey --help</b> for details about the available options.            |
| <b>APPNAME</b>                  | This option specifies the name of the application for which the key is to be generated. This must be an application for which <b>generatekey</b> can generate keys. |
| <b>from-application=appname</b> | This option specifies the name of the application with which the key is currently associated.                                                                       |
| <b>from-ident=keyident</b>      | This option specifies the identifier of the key to be retargeted. You can find this identifier by using the <b>nfkminfo</b> command-line utility.                   |

If **generatekey** cannot identify the key type for retargeting, you are prompted to specify the key type. Input the key type and press **Enter**.

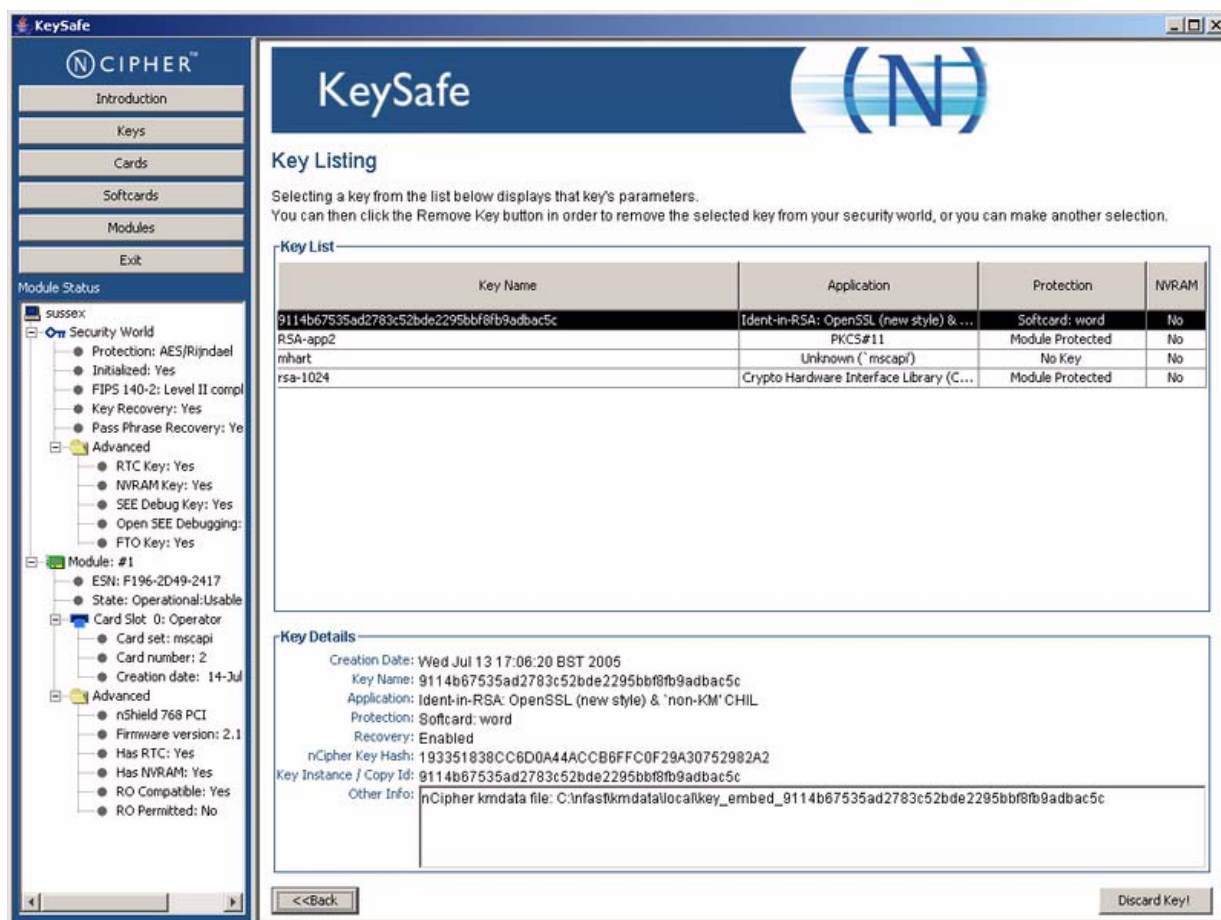
## Viewing keys

You can view existing keys in the Security World using KeySafe or the **nfkminfo** command-line utility.

### Viewing keys with KeySafe

In order to view a list of keys with KeySafe, follow these steps:

- 1 Start KeySafe. (For an introduction to KeySafe and for information on starting the software, see Appendix A: [Using KeySafe](#).)
- 2 Click the **Keys** menu button. KeySafe takes you to the **Key Operations** panel.
- 3 Click the **List Keys** navigation button. KeySafe takes you to the **Key Listing** panel:



The **Key Listing** panel displays all the keys that have been created in (or imported in to) the current security world. It displays the name of the key, the application for which it was created, the protection method that was used and whether the key is stored in NVRAM.

If you click a key's listing, KeySafe displays additional information about that key, for example, the application with which the key is used, whether or not the key is recoverable, and the key's name, creation date, hash, instance, and copy ID.

From the **Key Listing** panel, you can choose to discard a key from the security world by clicking the **Discard Key** button. (See [Discarding keys](#) on page 214 ).

## Viewing keys on the command line

Keys are listed on the command line using the **nfkminfo** command-line utility. To list the keys that have been created in the current security world, use one of the following commands:

---

```
nfkminfo -k [APPNAME[IDENT]]
```

---



---

```
nfkminfo -l [APPNAME[APPNAME...]]
```

---

The **-k|--key-list** option lists keys only. The **-l|--name-list** option lists keys and their names.

With either option, **APPNAME** is an optional application name. If you specify an application name, **nfkminfo** lists only the keys for that application. Commonly, **APPNAME** is often one of:

- **custom**
- **embed**
- **hwcrhk**
- **pkcs11**
- **kpm**
- **kps**
- **seeconf**
- **seeinteg**
- **simple**

You can also specify your own application names for **APPNAME** as appropriate to your system.

**Note** For example, user-defined application names can be created by using the **nfkml** library to generate arbitrary keys.

With the **--name-list** option, **IDENT** is the key identifier.

The command **nfkminfo --key-list** returns output of the form:

---

```
Key summary - 4 keys:
 AppName appname Ident ident AppName appname Ident ident AppName appname
Ident ident AppName appname Ident ident
```

---

To list information about a specific key, specify the **--key-list** option with an application and key identifier:

---

```
nfkminfo --key-list appname ident
```

---

This command returns output of the form:

---

```
Key AppName appname Ident ident BlobKA length 752
BlobPubKA length 316
BlobRecoveryKA length 868
name "name"
hash hash recovery Enabled
protection CardSet
other flags PublicKey +0x0
cardset hash_ktBlobKA
format 6 Token
other flags 0x0
hkm hash_km hkt hash_kt hkr none
BlobRecoveryKA
format 8 Indirect
other flags 0x0
hkm none
hkt none
hkr hash_krBlobPubKA
format 5 Module
other flags 0x0
hkm hash_km hkt none
hkr none
No extra entries
```

---

To list keys and names, specify the **--name-list** option. The command **nfkminfo --name-list** returns output of the form:

---

```
Key summary - 30 keys
in format key_<appname>_<ident> '<name>')
 key_appname_ident 'name '
 key_appname_ident 'name '
 key_appname_ident 'name '
 key_appname_ident 'name '
 key_appname_ident 'name '
 key_appname_ident 'name '
 key_appname_ident 'name '
 key_appname_ident 'name '
```

---

## Discarding keys

Discarding a key deletes the information about the key from the host disk. This option is only available in KeySafe.

If you have backup media, you can restore the information and the key becomes usable again. Likewise, if you have copies of the Security World data on several computers, erasing the data on one computer does not remove it from any other computer.

To destroy a key permanently you must either erase the OCS that is used to protect it or erase the Security World completely. There are no other ways to destroy a key permanently.

## Restoring keys

We do not supply tools for restoring a key that has been discarded. However if you have kept a backup of the host data for the Security World, you can restore a key from the backup data.

**Note** If you have NVRAM-stored keys, you must additionally ensure you have a backup of the key data stored on the relevant modules.



## Appendix A: Using KeySafe

This appendix describes KeySafe, the Security World management tool. It includes information about:

- starting KeySafe
- using the graphical user interface (GUI) for KeySafe
- using buttons to select and run operations
- using the keyboard to navigate KeySafe
- KeySafe error reporting.

To perform Security World management, card-set management, and key management tasks using KeySafe, see the relevant chapters of this guide.

**Note** By default, KeySafe uses the same mechanisms and supports the same features and applications as the **generatekey** command-line utility.

### Prerequisites for using KeySafe

To use KeySafe, you must have installed JRE/JDK 1.4.2, 1.5, or 1.6. We recommend that you install Java before you install the Security World Software. The Java executable must be on your path.

Java software is available from <http://java.sun.com/>. If your security policy does not allow the use of downloaded software, these components can be obtained on removable media from Sun or from your operating system vendor.

**Note** The IBM Java SDK, for AIX only, can be downloaded from <http://www.ibm.com/java/>.

The HP Java SDK & RTE, for HP-UX can be downloaded from <http://www.hp.com/java/>.

The **keysafe.jar** file must be specified in the Java class path.

For KeySafe to work, you must ensure that KeySafe and the hardserver are communicating on the same sockets. In the hardserver configuration file, set **priv\_port** (the port on which the hardserver listens for local privileged TCP connections) to 9001 and **nonpriv\_port** (the port on which the hardserver listens for local nonprivileged TCP connections) to 9000. You must then restart the hardserver. For more information, see [Stopping and restarting the hardserver](#) on page 70.

Note For information about hardserver configuration file settings, see [server\\_startup](#) on page 284.

Note The file **nCipherKM.jar**, if present, is located in the extensions folder of your local Java Virtual Machine. The uninstall process may not delete this file. Before reinstalling over an old installation, remove the **nCipherKM.jar** file. To locate the Java Virtual Machine extensions folder, see [Installing the nCipherKM JCA/JCE CSP](#) on page 157.

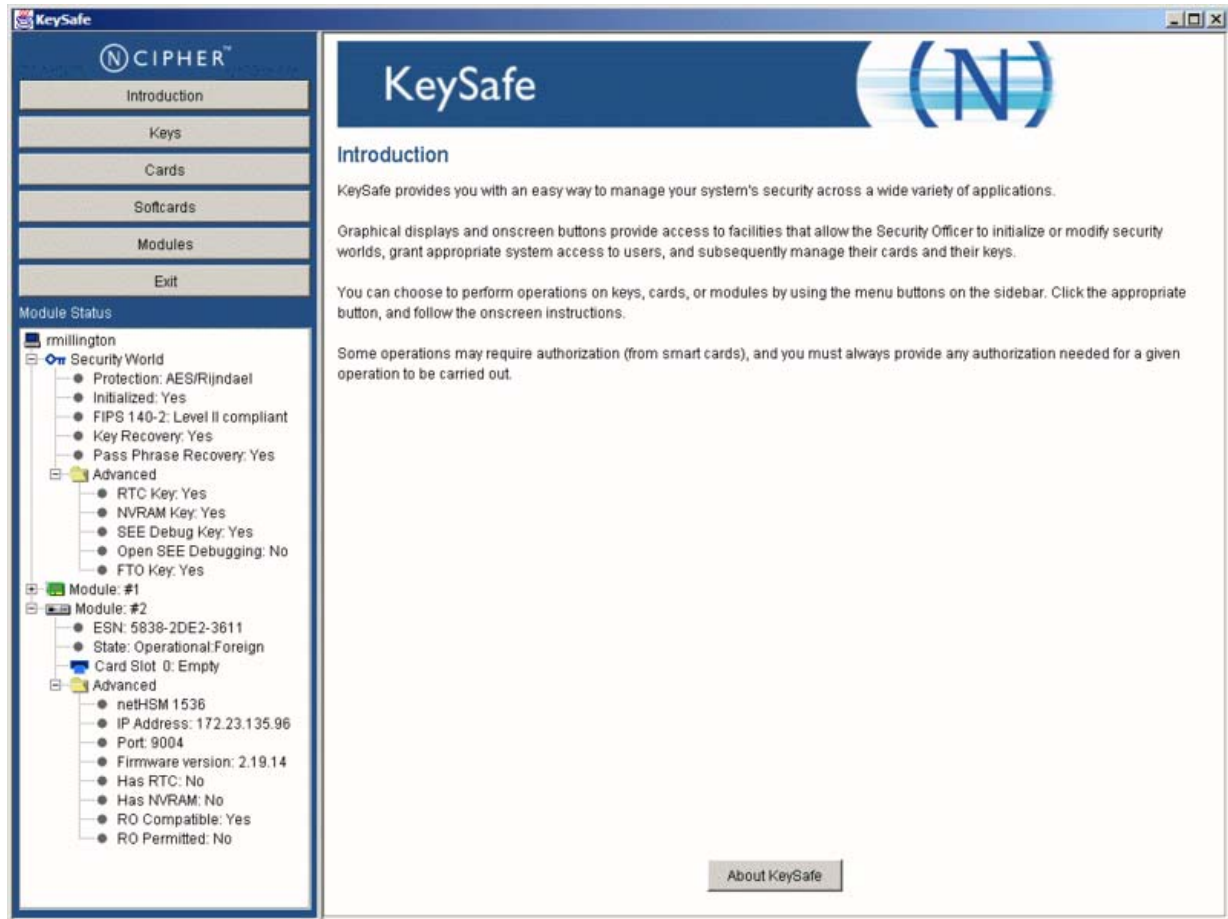
## Starting KeySafe

Note Ensure that Xwindows is properly configured and running before starting KeySafe.

Start KeySafe by running the **/opt/nfast/bin/ksafe** script (assuming you installed the Security World Software in the default **/opt/nfast/** directory).



When it starts, KeySafe displays a window similar to the one shown below:



**Note** We recommend using a 16-bit or 24-bit color depth to ensure accurate color reproduction. In environments with 8-bit color depth, some colors may be represented incorrectly, although KeySafe still operates.

## About the KeySafe window

The KeySafe window is divided into two areas:

- the sidebar (on the left), subdivided into:
  - the menu buttons (at the top of the sidebar)
  - the Module Status tree (at the bottom of the sidebar)
- the main panel area (on the right).

This layout is consistent throughout the KeySafe application.

## Sidebar

The sidebar provides access to different parts of the KeySafe application (with the menu buttons) and also displays information about both the current Security World and your module or modules (with the Module Status tree).

### Menu buttons

There are six menu buttons at the top of the sidebar:

| Menu button         | Description                                                                                                                                                                                                                                                                                                                                                                   |
|---------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Introduction</b> | Clicking the <b>Introduction</b> menu button takes you to the introductory panel that KeySafe displays at startup.                                                                                                                                                                                                                                                            |
| <b>Keys</b>         | Clicking the <b>Keys</b> menu button takes you to the Key Operations panel, from which you can choose to create, import, view, or destroy keys.                                                                                                                                                                                                                               |
| <b>Cards</b>        | Clicking the <b>Cards</b> menu button takes you to the Card Operations panel, from which you can: <ul style="list-style-type: none"> <li>• create, view, or erase Operator Cards</li> <li>• change the pass phrase on an Operator or Administrator Card, or recover the pass phrase from an Operator Card</li> <li>• replace an OCS or ACS.</li> </ul>                        |
| <b>Softcards</b>    | Clicking the <b>Softcards</b> menu button takes you to the Softcard Operations panel, from which you can: <ul style="list-style-type: none"> <li>• create, view or erase softcards</li> <li>• change or recover the pass phrase on a softcard</li> </ul>                                                                                                                      |
| <b>Modules</b>      | Clicking the <b>Modules</b> menu button takes you to the Module Operations panel, from which you can initialize a Security World, add modules to a Security World, or remove modules from a Security World. You cannot perform these operations on a module that is not in the pre-initialization mode.                                                                       |
| <b>Exit</b>         | Clicking the <b>Exit</b> button displays a dialog asking whether you are sure you want to exit KeySafe. Click <b>Yes</b> (or press <b>Enter</b> ) to exit KeySafe; click <b>No</b> to continue using KeySafe. Clicking the <b>Exit</b> dialog's close-window button will take you back to your KeySafe session. These features prevent you from closing KeySafe accidentally. |

While KeySafe is executing a command, the menu buttons are disabled. Their normal functionality returns when the command is completed.

### Module Status tree

The Module Status tree, in the lower part of the KeySafe sidebar, displays information about the current Security World and your modules in the form of a tree diagram.

At the top of the Module Status tree is an icon representing the computer on which the running copy of KeySafe is installed. The name of this computer is spelled out to the right of the icon.

Below the computer icon in the Module Status tree are icons and text identifiers representing the current Security World and your module(s). To the left of each icon is an expand/collapse toggle. By default, when KeySafe starts, these toggles are on their collapsed setting. Click the toggle to display expanded information about the Security World or module. Click the toggle again to collapse this information.

**Note** For clarity, screenshots of KeySafe in this guide are sometimes shown with certain Module Status tree toggles expanded.

## Security World information

When you toggle the Module Status tree view of Security World information to its expanded setting, KeySafe reports **Yes** or **No** for each of the following conditions:

- whether the Security World is initialized
- whether OCS and softcard replacement is enabled for this Security World (for more information, see [OCS and softcard replacement](#) on page 77)
- whether pass phrase replacement is enabled for a Security World (for more information, see [Pass phrase replacement](#) on page 77).

It also gives the following information:

- the Cipher suite—the type of key protecting the Security World
- the FIPS 140-2 level at which the Security World is operating (level 2 or level 3).

The expanded Security World information includes a folder labeled **Advanced**, which itself can be toggled into an expanded view. This folder displays advanced Security World attributes related to the nCipher Secure Execution Engine (SEE). When expanded, the **Advanced** folder displays **Yes** or **No** for each of the following:

- whether there is a real-time clock (RTC) authorization key for this Security World
- whether there is nonvolatile memory (NVRAM) authorization key for this Security World
- whether there is a Secure Execution Engine (SEE) debugging key for this Security World
- whether the Foreign Token Open key is enabled for this Security World

It also displays the level of SEE debugging that is enabled for this Security World.

## Module information

When you toggle the Module Status tree view of module information to its expanded setting for a given module, KeySafe displays:

- the module's electronic serial number (ESN), which is a unique identifier. You must quote a module's ESN if you need to contact Support. Keep a record of the ESN(s) associated with your module(s).
- the module's mode, which is one of the following:

| Mode                             | Description                                                                                                                                                         |
|----------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>PreInitMode</b>               | The module is in pre-initialization mode.                                                                                                                           |
| <b>InitMode</b>                  | The module is in initialization mode.                                                                                                                               |
| <b>Operational:Useable</b>       | The module is in the current Security World and useable for key operations.                                                                                         |
| <b>Operational:Unknown</b>       | The mode of the module cannot be determined.                                                                                                                        |
| <b>Operational:Uninitialized</b> | The module key is set and the module must be initialized before use.                                                                                                |
| <b>Operational:Factory</b>       | The module key is set to the factory default.                                                                                                                       |
| <b>Operational:Foreign</b>       | The module is from an unknown Security World.                                                                                                                       |
| <b>Operational:AcceOnly</b>      | The module is an acceleration-only module.                                                                                                                          |
| <b>Operational:Unchecked</b>     | Although the module appears to be in the current Security World, KeySafe cannot find a module initialization certificate (a <b>module_ESN</b> file) for this module |
| <b>Failed</b>                    | The module has failed.                                                                                                                                              |
| <b>PreMaintMode</b>              | The module is in the pre-maintenance mode.                                                                                                                          |
| <b>MaintMode</b>                 | The module is in the maintenance mode.                                                                                                                              |

You cannot initialize a Security World from KeySafe unless you have at least one module in the pre-initialization mode (**PreInitMode**). Likewise, if you want to add a module to an existing Security World, that module must be in the pre-initialization mode.

**Note** For information about putting a module into the pre-initialization mode, see [Putting a module in pre-initialization mode](#) on page 303. After you have initialized a Security World, in order for a module to be usable, you must put it into the operational mode, as described in [Putting a module in operational mode](#) on page 306.

- the status of the smart card reader slot(s).

**Note** For strict FIPS 140-2 level 3 Security Worlds, a **FIPS Auth Loaded** entry shows if an Administrator Card or Operator Card has been inserted to authorize an operation that requires a FIPS key.

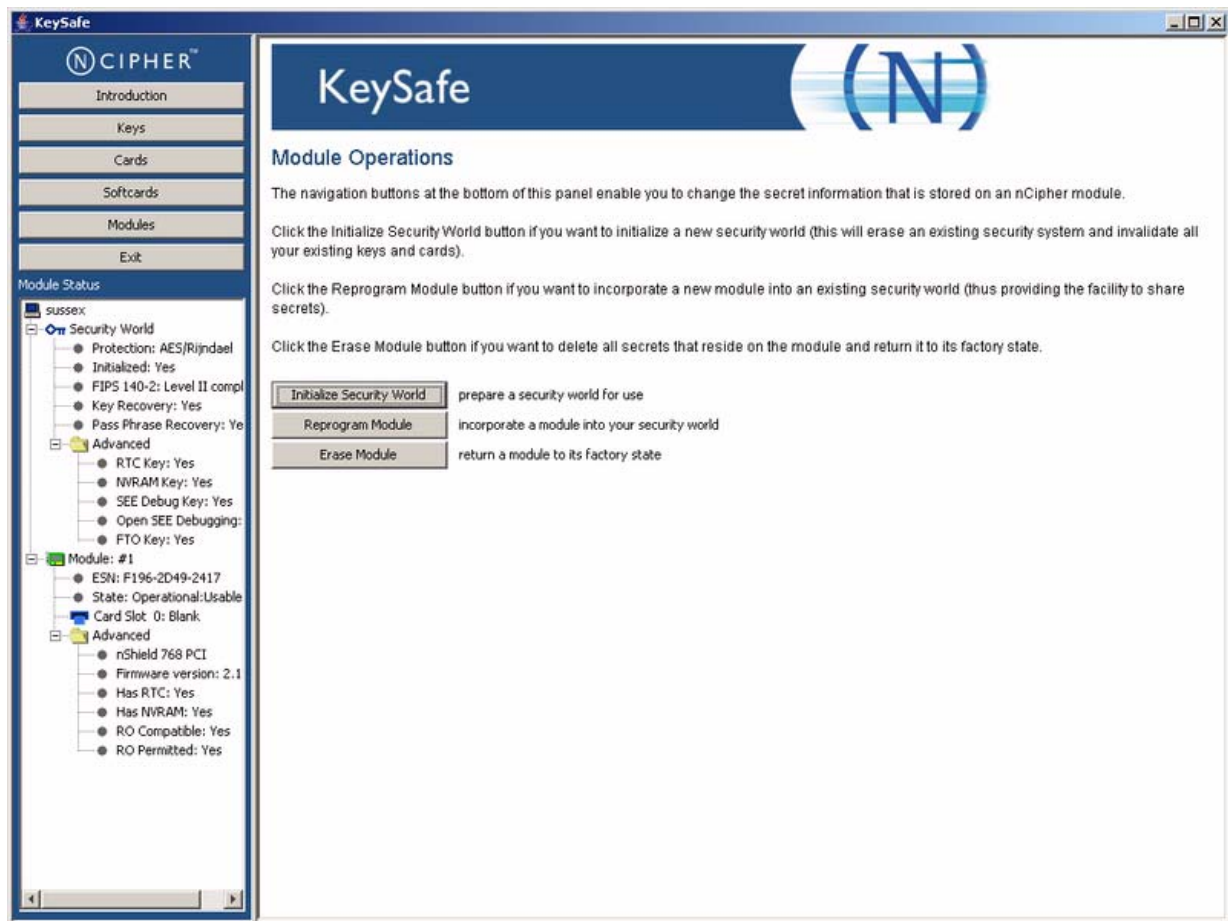
The Module status tree has an **Advanced** folder that shows the following details when expanded:

- the version of the module's firmware
- whether the module has a Real Time Clock (RTC)

- whether the module has nonvolatile memory (NVRAM).

## Main panel area

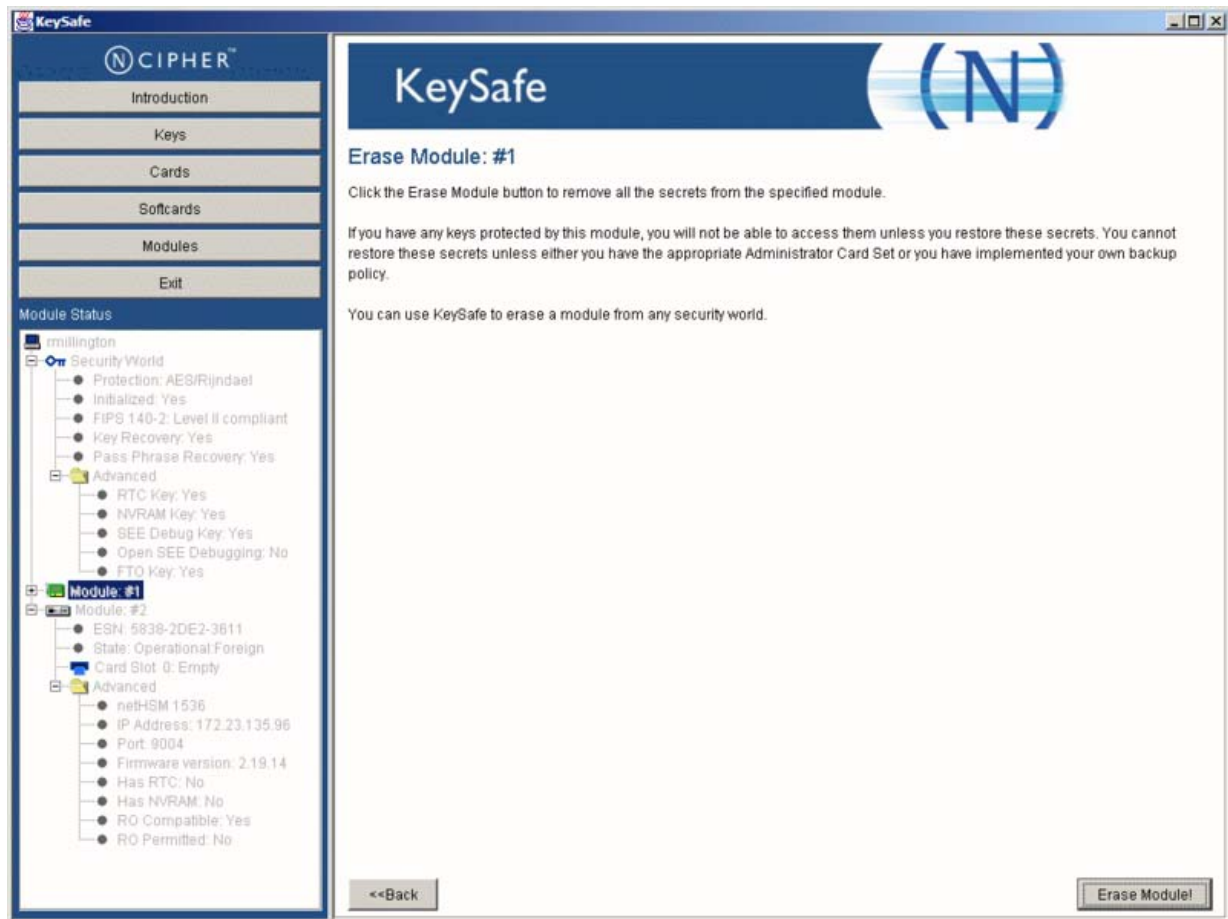
The KeySafe main panel area is used to display information and options pertaining to a chosen operation. For example, clicking the **Modules** menu button takes you to the **Module Operations** panel in the main panel area:



## Navigation buttons

At the bottom left of the **Module Operations** panel are three *navigation buttons*. Clicking a navigation button does *not* commit you to an action, but instead selects an operation and loads another panel of additional information and options related to the selected operation.

From the **Module Operations** panel, for example, clicking the **Erase Module** navigation button does not itself erase a module, but rather loads the **Erase Module** panel:

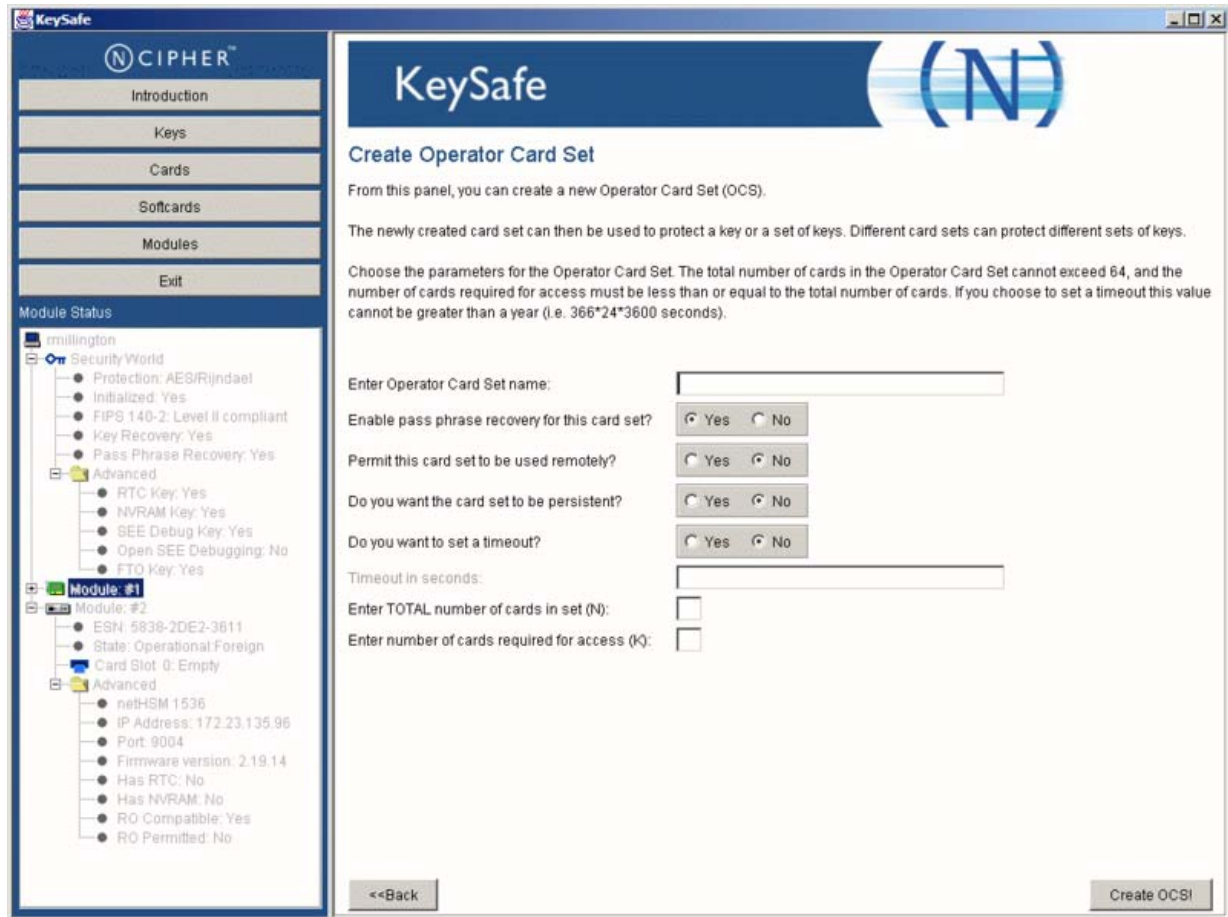


## Command buttons

At the bottom right of the **Erase Module** panel is a *command button* (the **Erase Module!** command button, in this case). Unlike clicking a navigation button, clicking a command button does commit you to an operation. Clicking a command button tells KeySafe to write or delete data: it is not necessarily possible to reverse such changes even if you subsequently cancel the operation.

For these reasons, command buttons are usually marked with an exclamation point (!) if there is a danger that a command could overwrite data that you may not want deleted. In some cases, clicking a command button causes KeySafe to display a dialog asking you to confirm your command. Such features help prevent you from accidentally destroying your data or keys.

Some panels require that, before you can click a command button, you select options by means of radio buttons or that you enter data into text fields:



If you click a command button without having entered data into a mandatory text field or if KeySafe detects that the information you provided is inconsistent or invalid, KeySafe displays an error message. Click the message's **OK** button, and then provide or correct the necessary information.

After you successfully issue a command by clicking a command button, the sidebar's menu buttons are disabled until the requested command is completed.

## Back buttons

Some KeySafe panels have **Back** buttons. Clicking a **Back** button takes you back to the previous screen.



## Navigating with the keyboard

The **Tab** key always takes you to the next field or button. If the cursor is not currently active in a text field, pressing the space bar or the **Enter** key activates the currently selected button (as if you had clicked it). Pressing the **Shift-Tab** button combination takes you to the previous field (if any) or deselects an automatically selected button (if any).

## Errors

If KeySafe detects an error from which it cannot recover, it may display a Fatal Error message, such as:



In this case, it could be that the hardserver is unable to receive TCP connections. The server program communicates with clients by using named pipes or TCP sockets.

**Note** For more information about these hardserver configuration file settings, see [server\\_startup](#) on page 284.

The error message shown could also indicate that the hardserver is not running (or that your module is physically disconnected). If the hardserver is not running, take the following steps:

- 1 Exit KeySafe.
- 2 Restart the server (as described in [Stopping and restarting the hardserver](#) on page 70).
- 3 Restart KeySafe.

Another fatal error from which KeySafe cannot recover is:



This error may mean that your hardserver or firmware is not current. To update your firmware, take the following steps:

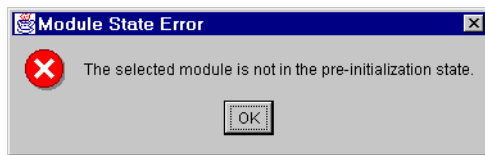
- 1 Exit KeySafe.



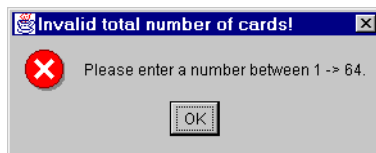
- 2 Update the firmware as described in Appendix J: [Upgrading firmware](#).
- 3 Restart KeySafe.

**Note** The firmware upgrade process destroys all persistent data held in a key-management module. If your security system requires that the persistent data held in a key-management module must survive intact during the upgrade or initialization of the key-management module, a backup and recovery mechanism of your **kmdata** directory must be implemented.

Other, more trivial errors include:

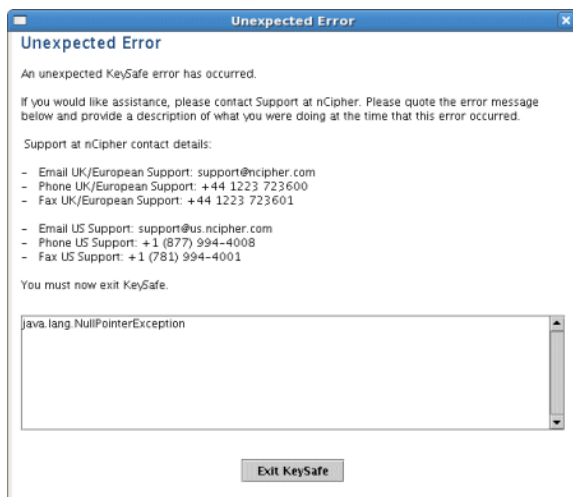


This error occurs if you are attempting to initialize, reprogram, or erase a module that is not in the pre-initialization mode.



This error occurs if you attempt to create a card set that would contain more than 64 cards.

However, if you receive any error message that looks similar to the following example, contact Support:





## Appendix B: Supplied utilities

This appendix describes the executable command-line utilities (utilities) that you can use for performing various configuration and administrative tasks related to your module.

These utilities exist in the **bin** subdirectory of your Security World Software installation. Unless noted, all utilities have the following standard help options:

- **-h|--help** displays help for the utility.
- **-v|--version** displays the version number of the utility.
- **-u|--usage** displays a brief usage summary for the utility.

### Utilities for general operations

Use the utilities described in this section to:

- Check the module configuration and verify that it functions as expected.
- Obtain statistics for checking the performance of the module.

| Utility              | Enables you to...                                                                                                                                                                                                                                                                                                                                                                                                        |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>enquiry</b>       | Obtain information about the hardserver (Security World Software server) and the modules connected to it.<br><br>Use this utility to: <ul style="list-style-type: none"><li>• Check if the software has been installed correctly.</li><li>• Check the firmware version.</li><li>• Check if the Remote Operator feature is enabled.</li></ul> For more information, see <a href="#">Test the installation</a> on page 48. |
| <b>checkmod</b>      | Check modulo exponentiations performed on the module, against the test data located in the <b>opt/nfast/testdata</b> directory.                                                                                                                                                                                                                                                                                          |
| <b>cfg-mkdefault</b> | Create a default client configuration file for the hardserver configuration sections.<br><br><div>Note The configuration file created by using this utility is not suitable for transfer to a netHSM.</div>                                                                                                                                                                                                              |

| Utility                             | Enables you to...                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|-------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>cfg-reread</b><br><br><b>fet</b> | <p>Load the hardserver configuration from the configuration file.</p> <ul style="list-style-type: none"> <li>• Activate features on an nShield module connected to the host.</li> <li>• View the status of features on a connected module.</li> <li>• Verify that a feature has been successfully enabled on a connected module.</li> </ul> <p>To view the status of features, run the tool without a smart card. If a FEM card is not present, or if any of the features are not enabled successfully, the utility prompts you to indicate what to do next.</p> <p><b>Note</b> To enable features, and view the status of or verify features on an nShield Connect or a netHSM unit, use the front panel rather than the <b>fet</b> utility.</p> <p>For more information, see <a href="#">Enabling features with a smart card</a> on page 66.</p> |
| <b>ncdate</b>                       | View, set, and update the time on a module's real-time clock.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| <b>ncversions</b>                   | <p>Obtain and verify the versions of the Security World Software components that are installed. This utility lists the following information:</p> <ul style="list-style-type: none"> <li>• Versions of all components, irrespective of whether they are installed individually or as part of a component bundle.</li> <li>• Version of each component bundle.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| <b>nfdiag</b>                       | <p>Obtain information about the module and the host on which it is installed. This diagnostic utility can save information to either a <b>ZIP</b> file or a text file.</p> <p>For more information, see <a href="#">nfdiag: diagnostics utility</a> on page 256.</p> <p><b>Note</b> Run this utility only if requested to do so by Support.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| <b>nopclearfail</b>                 | <p>Clear a module, put a module into the error state, or retry a failed module.</p> <p>For more information, see <a href="#">Checking and changing module mode</a> on page 302.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| <b>nvr-am-backup</b>                | Copy files between a module's NVRAM and a smart card, allowing files to be backed up and restored.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| <b>nvr-am-sw</b>                    | View and modify information about NVRAM areas.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <b>pubkey-find</b>                  | Obtain information of the public key from a certificate or certificate request (in a Base-64 encoded PEM file).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| <b>randchk</b>                      | Run a universal statistical test on random numbers returned by the module.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| <b>rtc</b>                          | View and set the module's real-time clock.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |

| Utility            | Enables you to...                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|--------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>slotinfo</b>    | <ul style="list-style-type: none"> <li>Obtain information about tokens in a module.</li> <li>Format a token.</li> </ul> <p>For more information, see <a href="#">Configuring hardservers for Remote Operator</a> on page 195.</p>                                                                                                                                                                                                                                       |
| <b>snmp *-serv</b> | <ul style="list-style-type: none"> <li>Test SNMP installations.</li> <li>Obtain module-related information from the SNMP agent.</li> </ul> <p>The utilities <b>snmpbulkwalk</b>, <b>snmpget</b>, <b>snmpgetnext</b>, <b>snmptable</b>, <b>snmpset</b>, <b>snmpstat</b>, <b>snmptranslate</b>, and standard, <b>snmpwalk</b> are general-purpose SNMP utilities.</p> <p>For more information, see <a href="#">Using the SNMP command-line utilities</a> on page 336.</p> |
| <b>stattree</b>    | <p>Obtain statistics gathered by the Security World Software server and modules.</p> <p>For more information, see <a href="#">stattree: information utility</a> on page 274.</p>                                                                                                                                                                                                                                                                                        |

## Hardware utilities

Use the following utilities to manage the firmware installed on an nShield module.

**Note** To upgrade or verify the firmware on an nShield Connect or a netHSM unit, use the front panel.

| Utility          | Enables you to...                                                                                                                                                                                                                                                                                                 |
|------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>fwcheck</b>   | Verify the firmware installed on a module.                                                                                                                                                                                                                                                                        |
| <b>loadrom</b>   | <ul style="list-style-type: none"> <li>Upgrade the module firmware.</li> <li>Obtain information about the firmware installed on a module.</li> </ul> <p>To determine the version security number of the firmware in a file and for more information, see <a href="#">Firmware on the DVD-ROM</a> on page 308.</p> |
| <b>nfloadmon</b> | <p>Upgrade the module monitor and firmware of nShield Edge and nShield Solo modules.</p> <p>For more information, see <a href="#">Upgrading firmware</a> on page 308.</p>                                                                                                                                         |

## Test analysis tools

Use the following utilities to test the cryptographic operational behavior of a module.

**Note** All the listed utilities, except the **floodtest** utility, are supported only on non-FIPS Security Worlds.

| Utility              | Enables you to...                                                                                                                                                                      |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>crypttest</b>     | Test all defined symmetric cryptographic mechanisms.                                                                                                                                   |
| <b>des_kat</b>       | Perform DES known-answer tests. This utility indicates if any of them fail.                                                                                                            |
| <b>floodtest</b>     | Perform hardware speed-testing by using modular exponentiation.                                                                                                                        |
| <b>kptest</b>        | Test the consistency of encryption and decryption, or of signature and verification, with the RSA and DSA algorithms.                                                                  |
| <b>ncthread-test</b> | Stress test modules and test nCore API concurrent connection support.                                                                                                                  |
| <b>perfcheck</b>     | Run various tests to measure the cryptographic performance of a module.<br><br>For more information, see <a href="#">perfcheck: performance measurement checking tool</a> on page 271. |
| <b>sigtest</b>       | Measure module speed using RSA or DSA signatures or signature verifications.                                                                                                           |

## Security World utilities

Use the utilities described in this section to:

- Set up and manage Security Worlds.
- Create and manage card sets and pass phrases.

- Generate keys and transfer keys between Security Worlds.

| Utility            | Enables you to...                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|--------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>bulkerase</b>   | <p>Erase multiple smart cards including Administrator Cards, Operator Cards, and FEM activation cards, in the same session.</p> <p><b>Note</b> Do not use the <b>bulkerase</b> utility to erase Administrator Cards from the current Security World.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| <b>cardpp</b>      | <p>Change, verify, and recover a pass phrase of an Operator Card.</p> <p>For more information, see:</p> <ul style="list-style-type: none"> <li>• <a href="#">Verifying the pass phrase of a card with cardpp</a> on page 134.</li> <li>• <a href="#">Changing known pass phrases with cardpp</a> on page 137.</li> <li>• <a href="#">Changing unknown or lost pass phrases</a> on page 138.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                              |
| <b>createocs</b>   | <p>Create and erase an OCS.</p> <p>For more information, see:</p> <ul style="list-style-type: none"> <li>• <a href="#">Creating an Operator Card Set from the command line</a> on page 118.</li> <li>• <a href="#">Erasing cards from the command line</a> on page 129.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| <b>initunit</b>    | <p>Initialize an nShield module.</p> <p><b>Note</b> To initialize an nShield Connect or a netHSM unit, use the front panel.</p> <p>For more information, see <a href="#">Erasing a module with initunit</a> on page 113.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| <b>generatekey</b> | <p>Generate, import, or retarget keys. This utility is included in the <b>Core Tools</b> bundle, which contains all the Security World Software utilities.</p> <p>For more information, see:</p> <ul style="list-style-type: none"> <li>• <a href="#">Generating keys on the command line</a> on page 201.</li> <li>• <a href="#">Importing keys from the command line</a> on page 206.</li> <li>• <a href="#">Example of key generation with generatekey</a> on page 202, for an example of key generation in batch mode.</li> <li>• <a href="#">Example of key importation with generatekey</a> on page 207, for an example of importing an RSA key.</li> <li>• <a href="#">Listing supported applications with generatekey</a> on page 210.</li> <li>• <a href="#">Retargeting keys with generatekey</a> on page 210.</li> </ul> |

| Utility              | Enables you to...                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>key-xfer-im</b>   | <p>Transfer keys between Security Worlds, when used in conjunction with the <b>mk-reprogram</b> utility. During the key transfer process, the <b>key-xfer-im</b> utility imports the module key that has been programmed into a new Security World.</p> <p>For more information, see <a href="#">Transferring keys between Security Worlds</a> on page 98.</p> <p><b>Note</b> You can also use the <b>migrate-world</b> utility to transfer keys between Security Worlds.</p>                                        |
| <b>kmfile-dump</b>   | Obtain key management information from a Security World's key management data file.                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| <b>migrate-world</b> | <p>Migrate existing keys to a destination Security World.</p> <p>For more information, see <a href="#">Security World migration</a> on page 103.</p>                                                                                                                                                                                                                                                                                                                                                                 |
| <b>mkaclx</b>        | <p>Generate non-standard cryptographic keys that can be used to perform specific functions, for example, to wrap keys and derive mechanisms. This utility includes options that are not available with the <b>generate-key</b> utility.</p> <p><b>Note</b> Ensure that you run the <b>mkaclx</b> utility with the options that are appropriate for your security infrastructure. If the appropriate options are not chosen, the security of existing keys might potentially be compromised.</p>                      |
| <b>mk-reprogram</b>  | <p>Transfer keys between Security Worlds, when used in conjunction with the <b>key-xfer-im</b> utility. During the key transfer process, the <b>mk-reprogram</b> utility programs the module key of a Security World to another Security World.</p> <p>For more information, see <a href="#">Transferring keys between Security Worlds</a> on page 98.</p> <p><b>Note</b> You can also use the <b>migrate-world</b> utility to transfer keys between Security Worlds.</p>                                            |
| <b>new-world</b>     | <p>Create and manage Security Worlds on nShield modules.</p> <p><b>Note</b> To create and manage Security Worlds on an nShield Connect or a nethSM device, use the front panel.</p> <p>For more information, see:</p> <ul style="list-style-type: none"> <li>• <a href="#">Creating a Security World by using new-world</a> on page 79.</li> <li>• <a href="#">Adding a module to a Security World with new-world</a> on page 97.</li> <li>• <a href="#">Erasing a module with new-world</a> on page 111.</li> </ul> |

| Utility          | Enables you to...                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>nfmcheck</b>  | Check Security World data for consistency.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| <b>nfkminfo</b>  | <p>Obtain information about a Security World and its associated cards and keys.</p> <p>For more information, see:</p> <ul style="list-style-type: none"> <li>• <a href="#">Displaying information about a Security World with nfkminfo on page 94.</a></li> <li>• <a href="#">Viewing card sets from the command line on page 132.</a></li> <li>• <a href="#">Viewing softcards with nfkminfo on page 133.</a></li> <li>• <a href="#">Viewing keys on the command line on page 212.</a></li> <li>• <a href="#">nfkminfo: information utility on page 259.</a></li> </ul>                                                                                                                                                                                                                                         |
| <b>nfmverify</b> | Perform Security World verification.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| <b>postrocs</b>  | <p>Transfer PKCS #11 keys to a new card set in the new Security World. When transferring keys by using either the <b>key-xfer-im</b> utility or the <b>migrate-world</b> utility, run the <b>postrocs</b> utility if there are any PKCS #11 keys that are protected by OCSs.</p> <p>Note PKCS #11 keys either have <b>keys_pkcs_um</b> or <b>key_pkcs_uc</b> as the prefix.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| <b>ppmk</b>      | <p>Create and manage softcards. Use this utility to:</p> <ul style="list-style-type: none"> <li>• View details of a softcard.</li> <li>• Create and delete a softcard.</li> <li>• View, change, and recover the pass phrase of a softcard.</li> </ul> <p>For more information, see:</p> <ul style="list-style-type: none"> <li>• <a href="#">Creating a softcard with ppmk on page 125.</a></li> <li>• <a href="#">Erasing softcards with ppmk on page 130.</a></li> <li>• <a href="#">Viewing softcards with ppmk on page 134.</a></li> <li>• <a href="#">Verifying the pass phrase of a softcard with ppmk on page 135.</a></li> <li>• <a href="#">Changing known softcard pass phrases with ppmk on page 138.</a></li> <li>• <a href="#">Replacing unknown pass phrases with ppmk on page 139.</a></li> </ul> |
| <b>preload</b>   | <p>Load keys into a module before an application is run in another session.</p> <p>For more information, see <a href="#">Transferring keys between Security Worlds on page 98.</a></p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| <b>racs</b>      | <p>Create a new ACS to replace an existing ACS.</p> <p>For more information, see <a href="#">Replacing an Administrator Card Set with racs on page 154.</a></p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| <b>rocs</b>      | <ul style="list-style-type: none"> <li>• Restore an OCS from a quorum of its cards.</li> <li>• Restore softcards.</li> </ul> <p>For more information, see:</p> <ul style="list-style-type: none"> <li>• <a href="#">Replacing OCSs or softcards with rocs on page 143.</a></li> <li>• <a href="#">Using rocs from the command line on page 149.</a></li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                   |



## CodeSafe utilities

Use the following helper utilities to develop and sign SEE machines. For more information about these utilities, see the *CodeSafe Developer Guide*.

| Utility                                                                                               | Enables you to...                                                                                                                 |
|-------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------|
| <b>elftool</b>                                                                                        | Convert ELF format executables into a format suitable for loading as an SEE machine.                                              |
| <b>hsc_loadseemachine</b>                                                                             | Load an SEE machine into each module that is configured to receive one, then publishes a newly created SEE World, if appropriate. |
| <b>loadsee-setup</b>                                                                                  | Set up the configuration of auto-loaded SEE machines.                                                                             |
| <b>modstate</b>                                                                                       | View the signed module state.                                                                                                     |
| <b>see-sock-serv</b><br><b>see-stdioe-serv</b><br><b>see-stdioesock-serv</b><br><b>see-stdoe-serv</b> | Activate or enable standard IO and socket connections for SEE machines using the <b>bsdlib</b> architecture.                      |
| <b>tct2</b> (Trusted Code Tool)                                                                       | Sign, pack, and encrypt file archives so that they can be loaded onto an SEE-ready nShield module.                                |

## PKCS #11

Use the following utilities to manage the interfaces between the PKCS #11 library and the module.

| Utility                | Enables you to...                                                                                                                                                                                                                                                          |
|------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>ckcerttool</b>      | Import a certificate as a PKCS #11 <b>CKO_CERTIFICATE</b> object of type <b>CKC_X_509</b> , and optionally, associate it with the corresponding private key.                                                                                                               |
| <b>ckcheckinst</b>     | Verify the installation of the nCipher PKCS #11 libraries.<br><br>For more information, see <a href="#">Checking the installation of the nCipher PKCS #11 library</a> on page 187.                                                                                         |
| <b>ckimportbackend</b> | Generate keys for use with PKCS #11 applications. When you run the <b>generatekey</b> utility to generate PKCS #11 keys, the <b>ckimportbackend</b> utility is executed in the background.<br><br><b>Note</b> Do not run this utility unless directed to do so by Support. |
| <b>cknfkmid</b>        | View values of attributes of PKCS #11 objects.                                                                                                                                                                                                                             |
| <b>ckshahmac</b>       | Perform a PKCS #11 test for vendor-defined <b>SHA1_HMAC</b> key signing and verification capabilities.                                                                                                                                                                     |
| <b>cksigtest</b>       | Measure module signing or encryption speed when used with nCipher PKCS #11 library calls.                                                                                                                                                                                  |

If you have installed **Cipher Tools**, you can use the following additional PKCS #11 utilities. For more information about these utilities, see the *Cryptographic API Integration Guide*.

| Utility           | Enables you to...                                                                                                                                                                                                                                                                                                                                                   |
|-------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>ckinfo</b>     | View PKCS #11 library, slot and token information. Use this utility to verify that the library is functioning correctly.                                                                                                                                                                                                                                            |
| <b>cklist</b>     | View details of objects on all slots. If invoked with a PIN argument, the utility lists public and private objects. If invoked with the <b>-n</b> ( <b>--nopin</b> ) option, the utility lists only the public objects.<br><br>This utility does not output any potentially sensitive attributes, even if the object has <b>CKA_SENSITIVE</b> set to <b>FALSE</b> . |
| <b>ckmechinfo</b> | View details of the supported PKCS #11 mechanisms provided by the module.                                                                                                                                                                                                                                                                                           |
| <b>ckrsagen</b>   | Test RSA key generation. You can use specific PKCS #11 attributes for generating RSA keys.                                                                                                                                                                                                                                                                          |

## nShield Connect and netHSM utilities

The utilities described in this section are used with nShield Connect and netHSM units only. Use these utilities to:

- Create and manage client configuration files.
- Enroll nTokens with an nShield Connect or a netHSM.
- Set up a Remote File System (RFS) and synchronize Security World data between an nShield Connect or a netHSM and the RFS.

| Utility                     | Enables you to...                                                                                                                                                                                                                                               |
|-----------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>anonkneti</b>            | View the ESN and <b>HK<sub>NETI</sub></b> key hash from a module identified by its IP address.                                                                                                                                                                  |
| <b>cfg-pushnethsm</b>       | Copy a specified configuration file from a remote file system to the file system on a specified module.                                                                                                                                                         |
| <b>config-serverstartup</b> | Edit the <b>[server_startup]</b> section of the configuration file for the client's hardserver to enable or disable TCP sockets.                                                                                                                                |
| <b>nethsmenroll</b>         | Edit the local hardserver configuration file to add the specified nShield Connect or netHSM unit. As an alternative to hand-editing a client's configuration file, you can run this utility on a client to configure it to access an nShield Connect or netHSM. |

| Utility             | Enables you to...                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|---------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>ntokenenroll</b> | Enroll a locally attached nToken with an nShield Connect or a netHSM unit. This utility installs the Electronic Serial Number (ESN) of the nToken within the client configuration file and displays the module's ESN and the hash of the key to be used in nToken authentication.                                                                                                                                                                                                                                                                                                             |
| <b>rfs-setup</b>    | Create a default RFS hardserver configuration on the client. Run this utility when you first configure a client.                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| <b>rfs-sync</b>     | <ul style="list-style-type: none"> <li>• Synchronize the Security World data of a client with the RFS, when you run the utility on the client.</li> <li>• Synchronize the Security World data of the RFS with the client, when you run the utility on the RFS.</li> </ul> <p>For more information, see:</p> <ul style="list-style-type: none"> <li>• <a href="#">Setting up client cooperation</a> on page 51.</li> <li>• <a href="#">rfs-sync</a> on page 54.</li> </ul> <p><b>Note</b> You can use this utility with nShield modules if an nShield Connect or a netHSM unit is present.</p> |

## Developer-specific utilities

Use the following utilities to ensure that the modules are functioning as expected and to test the cryptographic functionality at the nCore level.

| Utility         | Enables you to...                                                                                                                                                                                                                                                                |
|-----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>pollbare</b> | Obtain information about state changes. The functionality of this test utility depends on whether the server or a module supports nCore API poll commands. <p><b>Note</b> To know if your server or module supports nCore API poll commands, run the <b>enquiry</b> utility.</p> |
| <b>trial</b>    | Test the nCore API commands. You can use this utility interactively or from a script file.                                                                                                                                                                                       |



## Appendix C: Components on Security World Software DVD-ROMs

This appendix lists the contents of the component bundles and the additional software supplied on your Security World Software DVD-ROM. For information on installing the supplied software, see Chapter 3: [Support software installation](#).

We supply the hardserver and associated software as bundles of common components that provide much of the required software for your installation. In addition to the component bundles, we provide individual components for use with specific applications and features supported by certain Thales modules.

To list installed components, use the **ncversions** command-line utility.

### Security World Software component bundles

We supply component bundles containing many of the necessary components for your installation. Certain standard component bundles are offered for installation on all standard Security World Software DVD-ROMs, while additional component bundles are found on CipherTools and CodeSafe DVD-ROMs.

**Note** **nhsign Signed netHSM firmware files** are supplied for use with nShield Connect and netHSM.

### Standard component bundles

You are always offered the following standard component bundles on all standard Security World Software DVD-ROMs:

| Standard component bundles              |
|-----------------------------------------|
| hwsp Hardware Support (mandatory)       |
| ctls Core Tools (recommended)           |
| javasp Java Support (including KeySafe) |

The component bundles consist of various individual components.

The **hwsp Hardware Support (mandatory)** bundle contains the hardserver and kernel device drivers:

|                                                   |
|---------------------------------------------------|
| <b>hwsp Hardware Support (mandatory) bundle</b>   |
| nfserv Hardserver process executables and scripts |
| sdrv nFast Win 2000 driver signatures             |
| cfgall Hardserver config file support             |
| nflog Logging library support                     |

The **ctls Core Tools (recommended)** bundle contains all the Security World Software command-line utilities, including **generatekey**, low level utilities, and test programs:

|                                               |
|-----------------------------------------------|
| <b>ctls Core Tools (recommended) bundle</b>   |
| convrt Command line key conversions           |
| nftcl Command line key management (Tcl)       |
| nftcl Command line key generation and import  |
| nfuser Low level utilities and test programs  |
| nfuser Command line remote server management  |
| opensl nftcl certificate generation utility   |
| sworld Command line key management (C)        |
| tclsrc Tcl run time                           |
| tclstf Small Tcl utilities                    |
| nftcl Command line key generation and import  |
| tct2 Trusted Code Tool 2 command-line utility |
| pysrc Python source for developers            |
| nfpy nFPython header files                    |

We recommend that you always install the **ctls Core Tools bundle**.

**Note** The Core Tools bundle includes the **tclsrc Tcl run time**'s tools for creating the Security World, **KeySafe**, and **new-world**. This does not affect any other installation of Tcl on your computer.

The **javasp Java Support (including KeySafe)** bundle contains Java applications, including KeySafe:

|                                                      |
|------------------------------------------------------|
| <b>javasp Java Support (including KeySafe)bundle</b> |
| jutils Java utilities                                |
| jutils JNI shared library for jutils.jar             |
| kmjava Java Key Management classes                   |
| ksafe KeySafe 2                                      |
| nfjava nFast Java generic stub classes               |
| nftcl Java Key Management Support                    |

## Additional component bundles

We supply the following additional component bundles on CipherTools DVD-ROMs:

|                                                 |
|-------------------------------------------------|
| <b>Additional CipherTools component bundles</b> |
| ctd CipherTools Developer                       |
| jd Java Developer                               |

We supply the following additional component bundles on CodeSafe DVD-ROMs:

|                                              |
|----------------------------------------------|
| <b>Additional CodeSafe component bundles</b> |
| csd CodeSafe Developer                       |
| jd Java Developer                            |

The **ctd CipherTools Developer** bundle contains components supplied with the CipherTools Developer Kit:

|                                                                                  |
|----------------------------------------------------------------------------------|
| <b>ctd CipherTools Developer bundle</b>                                          |
| emvspj JNI library for payShield Java                                            |
| emvspp payShield developer library                                               |
| hwcrhk Crypto Hardware Interface (CHIL) dev kit                                  |
| nflibs nCipher libraries and headers, and example C source for utility functions |
| nfuser nCore & KM tools and example source                                       |
| pkcs11 nFast PKCS#11 developer's library                                         |
| sworld Key Management C library developers kit                                   |
| tcsrc Tcl run time - Headers and Libraries                                       |
| cutils C utilities library                                                       |
| nflog Logging library                                                            |
| hilibs GS libs & headers                                                         |
| pysrc Python source for developers                                               |
| nfpy nFPython header files                                                       |

The **csd CodeSafe Developer** bundle contains components supplied with the CodeSafe Developer Kit:

|                                                                                       |
|---------------------------------------------------------------------------------------|
| <b>csd CodeSafe Developer bundle</b>                                                  |
| csee Codesafe-C moduleside example code                                               |
| csee Codesafe-C hostside example code                                                 |
| module Firmware test scripts                                                          |
| nflibs Generic stub libraries and headers, and example C source for utility functions |
| nfuser nCore & KM tools and example source                                            |
| sworld Key Management C library developers kit                                        |
| tcsrc Tcl run time - Headers and Libraries                                            |
| cutils C utilities library                                                            |
| nflog Logging library                                                                 |
| hilibs GS libs & headers                                                              |
| jhsee Java hostside developer's kit                                                   |
| jhsee Java hostside SEE examples                                                      |
| ssllib Codesafe-SSL hostside code                                                     |
| ssllib Codesafe-SSL moduleside code                                                   |
| pysrc Python source for developers                                                    |
| nfpy nFPython header files                                                            |
| nfpy Libs and headers for codesafe/python                                             |

The **jd Java Developer** bundle contains components to support development of Java applications:

|                                             |
|---------------------------------------------|
| <b>jd Java Developer bundle</b>             |
| jcecs Java Key Management developer         |
| jutils Java utilities source and javadocs   |
| kmjava Java Key Management developer        |
| nfjava Java Generic Stub examples & javadoc |

## Security World for nShield User DVD-ROM

We supply the following component bundles and additional components on the Security World for nShield User DVD-ROM:

|                                         |
|-----------------------------------------|
| <b>Component Bundles</b>                |
| hwsp Hardware Support (mandatory)       |
| ctls Core Tools (recommended)           |
| javasp Java Support (including KeySafe) |

|                                                |
|------------------------------------------------|
| <b>Individual Components</b>                   |
| hwcrhk Crypto Hardware Interface (CHIL) plugin |
| jcecs nCipherKM JCA/JCE provider classes       |
| ncsnmp nCipher SNMP monitoring agent           |
| nhfw nShield Connect and netHSM firmware       |
| pkcs11 nCipher pkcs11 library                  |



## CipherTools DVD-ROM

We supply the following component bundles and additional components on the CipherTools DVD-ROM:

| Component Bundles                       |
|-----------------------------------------|
| hwsp Hardware Support (mandatory)       |
| ctls Core Tools (recommended)           |
| javasp Java Support (including KeySafe) |
| ctd CipherTools Developer               |
| jd Java Developer                       |

| Individual Components                          |
|------------------------------------------------|
| devref nCore API Documentation                 |
| hwcrhk Crypto Hardware Interface (CHIL) plugin |
| jcecsn nCipherKM JCA/JCE provider classes      |
| ncsnmp nCipher SNMP monitoring agent           |
| nhfw nShield Connect and netHSM firmware       |
| pkcs11 nCipher pkcs11 library                  |
| sslyp Open SSL source code patch file          |

## CodeSafe DVD-ROM

We supply the following component bundles and additional components on the CodeSafe DVD-ROM:

| Component Bundles                       |
|-----------------------------------------|
| hwsp Hardware Support (mandatory)       |
| ctls Core Tools (recommended)           |
| javasp Java Support (including KeySafe) |
| csd CodeSafe Developer                  |
| jd Java Developer                       |

| Individual Components                          |
|------------------------------------------------|
| csdref nCore CodeSafe API Documentation        |
| devref nCore API Documentation                 |
| gccsrc Prebuilt arm-gcc for Codesafe/C         |
| gccsrc Prebuilt powerpc-m-gcc for Codesafe/C   |
| hwcrhk Crypto Hardware Interface (CHIL) plugin |
| jceesp nCipherKM JCA/JCE provider classes      |
| ncsnmp nCipher SNMP monitoring agent           |
| nhfw nShield Connect and netHSM firmware       |
| pkcs11 nCipher pkcs11 library                  |

## Components required for particular functionality

Some functionality requires particular component bundles or individual components to be installed.

Ensure that you have installed the **hwsp Hardware Support (mandatory)** and **ctls Core Tools (recommended)** bundles.

If you have a CipherTools DVD-ROM, we recommend that you install the **ctd CipherTools Developer** bundle.

If you have a CodeSafe DVD-ROM, we recommend that you install the **csd CodeSafe Developer** bundle.

If you have a CodeSafe DVD-ROM and you are developing in C:

- if your module has a part code of the form nC4nn2 or Bn1nnn, install the **gccsrc Prebuilt arm-gcc for Codesafe/C** component.

- if your module has a part code of the form **nC4nn3**, **Bn2nnn**, **BN2nnn(-E)**, or **NH2nnn**, install the **gccsrc Prebuilt powerpc-gcc for Codesafe/C** component.

In these part codes, *n* represents any integer.

If you have a CipherTools DVD-ROM or a CodeSafe DVD-ROM and you are developing in Java, install the **jd Java Developer** and **javasp Java Support (including KeySafe)** bundles; after installation, ensure that you have added the **.jar** files to your **CLASSPATH**.

You must install the **nfdrvk** component if you are using a Thales PCI card.

## KeySafe

To use KeySafe, install the **ctls Core Tools** and the **javasp Java Support (including KeySafe)** bundles.

## PKCS #11 applications

If you want to use the module with PKCS #11 applications, including release 4.0 or later of Netscape Enterprise Server, Sun Java Enterprise System (JES), or Netscape Certificate Server 4, install the **pkcs11 nCipher PKCS11 library**. For detailed PKCS #11 configuration options, see:

- [nCipher PKCS #11 library](#) on page 165
- the appropriate third-party integration guide for your application.

Integration guides for third-party applications are available from the Thales web site:  
<http://iss.thalesgroup.com/Resources.aspx>.

## Cryptographic Hardware Interface Library applications

If you want to use the module with the Cryptographic Hardware Interface Library (CHIL) applications, install the **hwcrhk Crypto Hardware Interface (CHIL) plugin** component and, if required, the **sslyp OpenSSL source code patch file** component.

Note Security World Software supports OpenSSL 0.9.7g and later.

## nCipherKM JCA/JCE cryptographic service provider

If you want to use the nCipherKM JCA/JCE cryptographic service provider, you must install both:

- the **javasp Java Support (including KeySafe)** bundle
- the **jceesp nCipherKM JCA/JCE provider classes** component.

An additional JCE provider **nCipherRSAPrivateEncrypt** is supplied that is required for RSA encryption with a private key. Install this provider and ensure that the file **rsaprivenc.jar** is in your **CLASSPATH**.

For more information about configuring the nCipherKM JCA/JCE cryptographic service provider, see [nCipher JCA/JCE CSP](#) on page 156.

## nCipher SNMP monitoring agent

If you want to use the nCipher SNMP monitoring agent to monitor your modules, install the **ncsnmp nCipher SNMP monitoring agent** component. During the first installation process of the nCipher SNMP agent, the agent displays the following message:

---

```
If this is a first time install, the
nCipher SNMP Agent will not run by default. Please see the manual for further
instructions.
```

---

For instructions on how to activate the agent after installation, see Appendix K: [nCipher SNMP monitoring agent](#).



## Appendix D: Environment variables

This appendix describes the environmental variables used by Security World Software:

| Variable                     | Description                                                                                                                                                                                                                                                                                                                                                                                                                  |
|------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>KERNEL_HEADERS</b>        | This variable allows you to specify the path to kernel headers (if, for example, they are not in the default directory). It is necessary for the configuration script to be able to find the kernel headers when building the PCI driver during software installation.                                                                                                                                                       |
| <b>NFAST_CERTDIR</b>         | This variable specifies the path to the dynamic feature enabling Feature Certificates directory. You only need to change the value of this variable if you move the Installation directory. See <b>NFAST_HOME</b> , <b>NFAST_KMDATA</b> , and <b>NFAST_LOGDIR</b> .                                                                                                                                                          |
| <b>NFAST_DEBUG</b>           | This variable enables debug logging for the hardserver and the PKCS #11 library. You must set <b>NFAST_DEBUG</b> equal to a value in the range 1 – 7 for debug messages to be logged (see Appendix E: <a href="#">Logging, debugging, and diagnostics</a> ). For more information, see also <a href="#">Hardserver debugging</a> on page 253 and <a href="#">Logging and debugging information for PKCS #11</a> on page 253. |
| <b>NFAST_DEBUG_SYSLOG</b>    | This variable redirects debug logging to syslog. The value of the environment variable should be one of the syslog facilities to be used. Prefixing the facility name with + enables traditional logging and syslog simultaneously.                                                                                                                                                                                          |
| <b>NFAST_HOME</b>            | This variable specifies the path to the Installation directory, which is set by the Security World Software installation script. You only need to change the value of this variable if you move the Installation directory. See <b>NFAST_KMDATA</b> , <b>NFAST_CERTDIR</b> , and <b>NFAST_LOGDIR</b> .                                                                                                                       |
| <b>NFAST_HWCRRHK_LOGFILE</b> | This variable sets the name of the file to which the CHIL (Cryptographic Hardware Interface Library) writes its log from applications. This is in addition to the logging done according to the mechanisms in the published <b>hwcrhk</b> API, which vary according to the application in use.                                                                                                                               |

| Variable                                                       | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|----------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>NFAST_KMDATA</b>                                            | This variable sets the location of the Key Management Data directory. You only need to change the value of this variable if you move the Key Management Data directory. See <b>NFAST_HOME</b> , <b>NFAST_CERTDIR</b> , <b>NFAST_LOGDIR</b> , and <b>NFAST_KMLOCAL</b> .                                                                                                                                                                                                           |
| <b>NFAST_KMLOCAL</b>                                           | This variable specifies the location of the Key Management and Security World Data directory. If this environment variable is not set, by default the module looks for the Security World data in the <b>local</b> subdirectory of the Key Management Data directory. See <b>NFAST_KMDATA</b> .                                                                                                                                                                                   |
| <b>NFAST_LOGDIR</b>                                            | This variable specifies the location of the Log Files directory. You only need to change the value of this variable if you move the Log Files directory. See <b>NFAST_HOME</b> , <b>NFAST_KMDATA</b> , and <b>NFAST_CERTDIR</b> .                                                                                                                                                                                                                                                 |
| <b>NFAST_NFKM_TOKENSFILE</b><br><b>NFAST_NFKM_TOKENSSELECT</b> | This variable sets the default values for a file in which <b>ClientID</b> and <b>KeyIDs</b> are stored by the <b>preload</b> command-line utility.                                                                                                                                                                                                                                                                                                                                |
| <b>NFAST_SEE_MACHINEENCKEY_DEFAULT</b>                         | This variable is the name of the <b>SEEConf</b> key needed to decrypt SEE-machine images. Running the command <b>loadmache --encryptionkey=IDENT</b> (or <b>loadmache --unencrypted</b> ) overrides any value set by this variable.                                                                                                                                                                                                                                               |
| <b>NFAST_SEE_MACHINEENCKEY_module</b>                          | This variable is the name of the <b>SEEConf</b> key needed to decrypt the SEE-machine image targeted for the specified module. It overrides <b>NFAST_SEE_MACHINEENCKEY_DEFAULT</b> for the specified module. Running the command <b>loadmache --encryptionkey=IDENT</b> (or <b>loadmache --unencrypted</b> ) overrides any value set by this variable.                                                                                                                            |
| <b>NFAST_SEE_MACHINEIMAGE_DEFAULT</b>                          | This variable is the path of the SEE machine image to load on to any module for which a specific image is not defined. Supplying the <i>machine-filename</i> parameter when running the <b>loadmache</b> command-line utility overrides this variable. This variable is not affected when running the <b>loadsee-setup</b> or <b>hsc_loadseemachine</b> utilities.                                                                                                                |
| <b>NFAST_SEE_MACHINEIMAGE_module</b>                           | This variable is the path of the SEE machine image to load on to the specified module. If set, this variable overrides the use of <b>NFAST_SEE_MACHINEIMAGE_DEFAULT</b> for the specified module. Supplying the <i>machine-filename</i> parameter when running the <b>loadmache</b> command-line utility overrides the <b>NFAST_SEE_MACHINEIMAGE_module</b> variable. This variable is not affected when running the <b>loadsee-setup</b> or <b>hsc_loadseemachine</b> utilities. |

| Variable                                                 | Description                                                                                                                                                                                                                                                                                                                                                                                                 |
|----------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>NFAST_SEE_MACHINESIGHASH_DEFAULT</b>                  | This variable is the default key hash of the vendor signing key ( <b>seeinteg</b> ) that signs SEE machine images. This variable is only required if you are using a dynamic SEE feature with an encrypted SEE machine. Running the command <b>loadmache --sighash=HASH</b> any value set in this variable.                                                                                                 |
| <b>NFAST_SEE_MACHINESIGHASH_module</b>                   | This variable is the key hash of the vendor signing key ( <b>seeinteg</b> ) that signs SEE machine images for the specified module. It overrides <b>NFAST_SEE_MACHINESIGHASH_DEFAULT</b> for the specified module. This variable is only required if you are using a dynamic SEE feature with an encrypted SEE machine. Running the command <b>loadmache --sighash=HASH</b> any value set in this variable. |
| <b>NFAST_SERVER</b><br><b>NFAST_PRIV_SERVER</b>          | These variables specify the path to the hardserver. <b>NFAST_SERVER</b> specifies the path to a non-privileged hardserver. <b>NFAST_PRIVSERVER</b> specifies the path to a privileged hardserver. See <a href="#">Hardserver start-up settings</a> on page 59 and <a href="#">server_startup</a> on page 284.                                                                                               |
| <b>NFAST_SERVER_PORT</b><br><b>NFAST_SERVER_PRIVPORT</b> | This variable sets the TCP port numbers that the hardserver uses for connections over TCP sockets. This variable is available for this purpose for backward compatibility only; you should configure ports in the configuration file, as described in <a href="#">server_startup</a> on page 284. If you set this variable, it overrides the values in the configuration file.                              |
| <b>NFAST_SERVER_SYSLOG</b>                               | This variable sets server logging to syslog. The value of the environment variable should be one of the syslog facilities to be used. Prefixing the facility name with <b>+</b> enables traditional logging and syslog simultaneously.                                                                                                                                                                      |
| <b>NFLOG_CATEGORIES</b>                                  | This variable is used to filter log messages by supplying a colon-separated list of allowable message categories; see Appendix E: <a href="#">Logging, debugging, and diagnostics</a> . If no value is supplied, all message categories are logged.                                                                                                                                                         |
| <b>NFLOG_SEVERITY</b>                                    | This variable is used to filter log messages by supplying a minimum severity level to be logged; see <a href="#">Logging, debugging, and diagnostics</a> on page 249. If no value is supplied, the default severity level is <b>WARNING</b> .                                                                                                                                                               |
| <b>NFLOG_DETAIL</b>                                      | This variable is used to filter log messages by supplying a bitmask of detail flags; see Appendix E: <a href="#">Logging, debugging, and diagnostics</a> . The default is <b>time+severity+writeable</b> .                                                                                                                                                                                                  |

| Variable           | Description                                                                                                                                                                                                                                                      |
|--------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| NFLOG_FILE         | This variable is used to specify a filename (or file descriptor) in which log messages are to be written; see Appendix E: <a href="#">Logging, debugging, and diagnostics</a> . The default is <b>stderr</b> (the equivalent of file descriptor <b>&amp;2</b> ). |
| OPENSSL_HWCRHK_LOG | This variable sets the directory in which the CHIL (Cryptographic Hardware Interface Library) creates its log file. This must be a directory for which the user (that the CHIL-enabled application runs as) has write permission.                                |





## Appendix E: Logging, debugging, and diagnostics

This appendix describes the settings and tools you can use to access the logging and debugging information generated by the Security World Software. You are also shown how to obtain system information using the **nfdiag** command-line utility.

### Logging and debugging

**Note** The current release of Security World Software uses controls for logging and debugging that differ from those used in previous releases. However, settings you made in previous releases to control logging and debugging are still generally supported in the current release, although in some situations the output is now formatted differently.

**Note** Some text editors, such as Notepad, can cause NFLOG to stop working if the NFLOG file is open at the same time as the hardserver is writing the logs.

### Environment variables to control logging

The Security World for nShield generates logging information that is configured through a set of four environment variables:

#### NFLOG\_FILE

This environment variable specifies the name of a file (or a file descriptor, if prefixed with the **&** character) to which logging information is written. The default is **stderr** (the equivalent of **&2**).

Ensure that you have permissions to write to the file specified by **NFLOG\_FILE**.

#### NFLOG\_SEVERITY

This environment variable specifies a minimum severity level for logging messages to be written (all log messages less severe than the specified level are ignored). The level can be one of (in order of greatest to least severity):

- 1 **FATAL**

- 2 **SEVERE**
- 3 **ERROR**
- 4 **WARNING**
- 5 **NOTIFICATION**
- 6 **DEBUG $n$**  (where  $n$  can be an integer from 1 to 10 inclusive that specifies increasing levels of debugging detail, with 10 representing the greatest level of detail, although the type of output is depends on the application being debugged).

**Note** The increasingly detailed information provided by different levels of **DEBUG $n$**  is only likely to be useful during debugging, and we recommend not setting the severity level to **DEBUG $n$**  unless you are directed to do so by Support.

The default severity level is **WARNING**.

## NFLOG\_DETAIL

This environment variable takes a hexadecimal value from a bitmask of detail flags as described in the following table (the **logdetail** flags are also used in the **hardserver** configuration file to control **hardserver** logging; see [server\\_settings](#) on page 282):

| Hexadecimal flag | Function                                                                                                                                                   | logdetail flags |
|------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------|
| 0x00000001       | This flag shows the external time (that is, the time according to your machine's local clock) with the log entry. It is on by default.                     | external_time   |
| 0x00000002       | This flag shows the external date (that is, the date according to your machine's local clock) with the log entry.                                          | external_date   |
| 0x00000004       | This flag shows the external process ID with the log entry.                                                                                                | external_pid    |
| 0x00000008       | This flag shows the external thread ID with the log entry.                                                                                                 | external_tid    |
| 0x00000010       | This flag shows the external <b>time_t</b> (that is, the time in machine clock ticks rather than local time) with the log entry.                           | external_time_t |
| 0x00000020       | This flag shows the stack backtrace with the log entry.                                                                                                    | stack_backtrace |
| 0x00000040       | This flag shows the stack file with the log entry.                                                                                                         | stack_file      |
| 0x00000080       | This flag shows the stack line number with the log entry.                                                                                                  | stack_line      |
| 0x00000100       | This flag shows the message severity (a severity level as used by the <b>NFLOG_SEVERITY</b> environment variable) with the log entry. It is on by default. | msg_severity    |

| Hexadecimal flag | Function                                                                                                                                                                                                                | logdetail flags |
|------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------|
| 0x00000200       | This flag shows the message category (a category as used by the <b>NFLOG_CATEGORIES</b> environment variable) with the log entry.                                                                                       | msg_categories  |
| 0x00000400       | This flag shows message writeables, extra information that can be written to the log entry, if any such exist. It is on by default.                                                                                     | msg_writeable   |
| 0x00000800       | This flag shows the message file in the original library. This flag is likely to be most useful in conjunction with Security World Software-supplied example code that has been written to take advantage of this flag. | msg_file        |
| 0x00001000       | This flag shows the message line number in the original library. This flag is likely to be most useful in conjunction with example code we have supplied that has been written to take advantage of this flag.          | msg_line        |
| 0x00002000       | This flag shows the date and time in UTC (Coordinated Universal Time) instead of local time.                                                                                                                            | options_utc     |

## NFLOG\_CATEGORIES

This environment variable takes a colon-separated list of categories on which to filter log messages (categories may contain the wild-card characters \* and ?). If you do not supply any values, then all categories of messages are logged. This table lists the available categories:

| Category      | Description                                                                                                                                                                                         |
|---------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| nflog         | Logs all general messages relating to nflog.                                                                                                                                                        |
| nflog-stack   | Logs messages from <b>StackPush</b> and <b>StackPop</b> functions.                                                                                                                                  |
| memory-host   | Logs messages concerning host memory.                                                                                                                                                               |
| memory-module | Logs messages concerning module memory.                                                                                                                                                             |
| gs-stub       | Logs general generic stub messages. (Setting this category works like using the <b>dbg_stub</b> flag with the logging functionality found in previous Security World Software releases.)            |
| gs-stubbignum | Logs bignum printing messages. (Setting this category works like using the <b>dbg_stubbignum</b> flag with the logging functionality found in previous Security World Software releases.)           |
| gs-stubinit   | Logs generic stub initialization routines. (Setting this category works like using the <b>dbg_stubinit</b> flag with the logging functionality found in previous Security World Software releases.) |
| gs-dumpenv    | Logs environment variable dumps. (Setting this category works like using the <b>dbg_dumpenv</b> flag with the logging functionality found in previous Security World Software releases.)            |
| nfkm-getinfo  | Logs <b>nfkm-getinfo</b> messages.                                                                                                                                                                  |
| nfkm-newworld | Logs messages about world generation.                                                                                                                                                               |

| Category          | Description                                                                                              |
|-------------------|----------------------------------------------------------------------------------------------------------|
| nfm-admin         | Logs operations using the Administrator Card Set.                                                        |
| nfm-kmdata        | Logs file operations in the <b>kmdata</b> directory.                                                     |
| nfm-general       | Logs general NFM library messages.                                                                       |
| nfm-keys          | Logs key loading operations.                                                                             |
| nfm-preload       | Logs <b>preload</b> operations.                                                                          |
| nfm-ppmk          | Logs softcard operations.                                                                                |
| serv-general      | Logs general messages about the local hardserver.                                                        |
| serv-client       | Logs messages relating to clients or remote hardservers.                                                 |
| serv-internal     | Logs severe or fatal internal errors.                                                                    |
| serv-startup      | Logs fatal startup errors.                                                                               |
| servdbg-stub      | Logs all generic stub debugging messages.                                                                |
| servdbg-env       | Logs generic stub environment variable messages.                                                         |
| servdbg-underlay  | Logs messages from the OS-specific device driver interface                                               |
| servdbg-statemach | Logs information about the server's internal state machine.                                              |
| servdbg-perf      | Logs messages about the server's internal queuing.                                                       |
| servdbg-client    | Logs external messages generated by the client.                                                          |
| servdbg-messages  | Logs server command dumps.                                                                               |
| servdbg-sys       | Logs OS-specific messages.                                                                               |
| hwcrhk            | Logs messages from the CHIL (Cryptographic Hardware Interface Library).                                  |
| pkcs11-sam        | Logs all security assurance messages from the PKCS #11 library.                                          |
| pkcs11            | Logs all other messages from the PKCS #11 library.                                                       |
| rqcard-core       | Logs all card-loading library operations that involve standard message passing (including slot polling). |
| rqcard-ui         | Logs all card-loading library messages from the current user interface.                                  |
| rqcard-logic      | Logs all card-loading library messages from specific logics.                                             |

You can set a minimum level of hardserver logging by supplying one of the values for the **NFLOG\_SEVERITY** environment variable in the hardserver configuration file, and you can likewise specify one or more values for the **NFLOG\_CATEGORIES** environment variable. For detailed information about the hardserver configuration file settings that control logging, see [server\\_settings](#) on page 282 in Appendix F: [Hardserver configuration files](#).

**Note** If none of the four environment variables are set, the default behavior is to log nothing, unless this is overridden by any individual library. If any of the four variables are set, all unset variables are given default values.

## Logging and debugging information for PKCS #11

In order to get PKCS #11 logging and debugging output, you must set the **CKNFAST\_DEBUG** environment variable equal to 1 and specify any appropriate **pks11** categories using the **NFLOG\_CATEGORIES** environment variable.

This environment variable takes a colon-separated list of categories on which to filter log messages (categories may contain the wildcards characters \* and ?).

The following table maps PKCS #11 debug level numbers to the corresponding **NFLOG\_SEVERITY** value:

| PKCS #11 debug level | PKCS #11 debug meaning | NFLOG_SEVERITY value | Output in log        |
|----------------------|------------------------|----------------------|----------------------|
| 0                    | DL_None                | NONE                 |                      |
| 1                    | DL_EFatal              | FATAL                | "Fatal error:"       |
| 2                    | DL_EError              | ERROR                | "Error:"             |
| 3                    | DL_Fixup               | WARNING              | "Fixup:"             |
| 4                    | DL_Warning             | WARNING              | "Warning:"           |
| 5                    | DL_EApplic             | ERROR                | "Application error:" |
| 6                    | DL_Assumption          | NOTIFICATION         | "Unsafe assumption:" |
| 7                    | DL_Call                | DEBUG2               | ">> "                |
| 8                    | DL_Result              | DEBUG3               | "< "                 |
| 9                    | DL_Arg                 | DEBUG4               | "> "                 |
| 10                   | DL_Detail              | DEBUG5               | "D "                 |
| 11                   | DL_DetailMutex         | DEBUG6               | "DM "                |

## Hardserver debugging

Hardserver debugging is controlled by specifying one or more **servdbg-\*** categories (from the **NFLOG\_CATEGORIES** environment variable) in the hardserver configuration file; see [server\\_settings](#) on page 282 in Appendix F: [Hardserver configuration files](#). However, unless you also set the **NFAST\_DEBUG** environment variable to a value in the range 1 – 7, no debugging is produced (regardless of whether or not you specify **servdbg-\*** categories in the hardserver configuration file). This behavior helps guard against the additional load debugging places on the CPU usage; you can set the desired **servdbg-\*** categories in the hardserver configuration file, and then enable or disable debugging by setting the **NFAST\_DEBUG** environment variable.

The **NFAST\_DEBUG** environment variable controls debugging for the general stub or hardserver. The value is an octal number, in the range 1 – 7. It refers bitwise to a number of flags:

| Flag | Result                                                                       |
|------|------------------------------------------------------------------------------|
| 1    | Generic stub debugging value.                                                |
| 2    | Show bignum values.                                                          |
| 4    | Show initial <b>NewClient</b> or <b>ExistingClient</b> command and response. |

For example, if the **NFAST\_DEBUG** environment variable is set to 6, flags 2 and 4 are used.

**Note** If the **NFAST\_DEBUG** environment variable value includes flag 1 (Generic stub debugging value), the **logdetail** value in the hardserver configuration file (one of the values for the **NFLOG\_DETAIL** environment variable) controls the level of detail printed.

Do not set the **NFAST\_DEBUG** environment variable to a value outside the range 1 – 7. If you set it to any other value, the hardserver does not start.

## Debugging information for Java

This section describes how you can specify the debugging information generated by Java.

**Note** Do not set **NFJAVA\_DEBUG** or **NFJAVA\_DEBUGFILE** in the environment because Java does not pick up variables from the environment.

### Setting the Java debugging information level

In order to make the Java generic stub output debugging information, set the Java property **NFJAVA\_DEBUG**. The debugging information for **NFJAVA**, **NFAST**, and other libraries (for example, **KMJAVA**) can all use the same log file and have their entries interleaved.

You set the debugging level as a decimal number. To determine this number:

- 1 Select the debugging information that you want from the following list:

---

|                      |                                                             |
|----------------------|-------------------------------------------------------------|
| NONE =               | 0x00000000 (debugging off)                                  |
| MESS_NOTIFICATIONS = | 0x00000001 (occasional messages including important errors) |
| MESS_VERBOSE =       | 0x00000002 (all messages)                                   |
| MESS_RESOURCES =     | 0x00000004 (resource allocations)                           |
| FUNC_TRACE =         | 0x00000008 (function calls)                                 |
| FUNC_VERBOSE =       | 0x00000010 (function calls + arguments)                     |
| REPORT_CONTEXT =     | 0x00000020 (calling context e.g ThreadID and time)          |
| FUNC_TIMINGS =       | 0x00000040 (function timings)                               |
| NFJAVA_DEBUGGING =   | 0x00000080 (Output NFJAVA debugging info)                   |

---

- 2 Add together the hexadecimal value associated with each type of debugging information.

For example, to set **NFJAVA\_DEBUGGING** and **MESS\_NOTIFICATIONS**, add 0x00000080 and 0x00000001 to make 0x00000081.

- 3 Convert the total to a decimal and specify this as the value for the variable.

For example, to set **NFJAVA\_DEBUGGING** and **MESS\_NOTIFICATIONS**, include the line:

---

```
NFJAVA_DEBUG=129
```

---

For **NFJAVA** to produce output, **NFJAVA\_DEBUG** must be set to at least **NFJAVA\_DEBUGGING** + **MESS\_NOTIFICATIONS**. Other typical values are:

- **255**: All output
- **130**: nfjava debugging and all messages (**NFJAVA\_DEBUGGING** and **MESS\_VERBOSE**)
- **20**: function calls and arguments and resource allocations (**FUNC\_VERBOSE** and **MESS\_RESOURCES**)

## Setting the Java debugging file

You can set the **NFJAVA\_DEBUGFILE** property to direct output to a given file name, for example:

---

```
java -DNFJAVA_DEBUGFILE=myfile -classpath classname
```

---

If **NFJAVA\_DEBUGFILE** is not set, the standard error stream **System.err** is used.

**Note** Set these variables only when developing code or at the request of Support.



Debug output contains all commands and replies sent to the hardserver in their entirety, including all plain texts and the corresponding cipher texts as applicable.

## Diagnostics and system information

**Note** Besides the diagnostic tools described in this section, we also supply a performance tool that you can use to test Web server performance both with and without a Thales module. This tool is supplied separately. If you require a copy, contact your Sales representative.

## **nfdiag: diagnostics utility**

The **nfdiag** command-line utility is a diagnostics tool that gathers information about the system on which it is executed. It can save this information to either a **.zip** file or a text file.

Under normal operating conditions, you do not need to run **nfdiag**. You can run **nfdiag** before contacting Support and include its output file with any problem report.

### Usage

**Note** Run **nfdiag** with the standard **-h|--help** option to display information about the options and parameters that control the program's behavior.



The **nfdiag** command-line utility is an interactive tool. When you run it, it prompts you to supply the following information:

| Option                                    | Actions to take                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|-------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| which application(s) you are using        | Identify all application software installed on the machine on which any problem with the Thales product occurs.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| what APIs you are using                   | Describe any custom software, especially any interaction it has with the nShield security system.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| a description of the problem              | Include as much detail as possible, including any error messages you have seen.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| a Support ticket number (if you have one) | When you contact Support you are supplied with a Support ticket number. Enter this number to help Support expedite the collection of any information you have sent.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| a contact email address                   | Supply an email address that has as few e-mail/spam filters as possible so that any additional files that Support sends to you are not blocked. We use the e-mail address you supply here <i>only</i> for communication directly related to your problem report.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| a contact name                            | Enter your name (or the name of an appropriate person for contact by Support).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| a contact telephone number                | Include the appropriate country and any region code for your contact telephone number.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| any additional diagnostic file(s)         | <p>By default, <b>nfdiag</b> asks if you want to supply any additional diagnostic files. If you do not want to supply additional diagnostic files, type <b>(N)</b>, or <b>(Enter)</b> to continue.</p> <p>If you want to supply additional diagnostic files:</p> <ol style="list-style-type: none"> <li>1 Type <b>(Y)</b>.</li> <li>2 When prompted, supply the name of the file to be attached. Attached files must be in plain text format.</li> <li>3 After you have supplied the name of a file to attach, <b>nfdiag</b> asks again if you want to supply any additional diagnostic files. Repeat the process to attach additional files.</li> </ol> <p>When you do not want to attach any more files, type <b>(N)</b>, or <b>(Enter)</b> to continue.</p> |

**Note** Except for a Support ticket number, **nfdiag** requires non-NULL answers to all its prompts for information.

Supplying this information helps **nfdiag** capture as much relevant information as possible for any problem report to Support. As you supply information at each prompt in turn, press **(Enter)** to confirm the information and continue to the next prompt. Information you supply cannot extend over multiple lines, but if you need to supply this level of information, you can include it in additional attached files, as described.

By default, **nfdiag** runs in verbose mode, providing feedback on each command that it executes and which log files are available. If the system is unable to execute a command, the verbose output from **nfdiag** shows where commands are stalling or waiting to time out.

At any time while **nfdiag** is running, you can type **Ctrl-C** to cancel its current commands and re-run it.

## Output

After you have finished supplying information for each required prompt, **nfdiag** generates a plain text output file and displays its file name.

If the file **opt/nfast/log/logfile** exists, **nfdiag** automatically includes this file in its output. If the file **opt/nfast/log/logfile** does not exist, **nfdiag** warns you that it could not process this file. This warning does not affect the validity of the generated output file.

When complete, this output file contains the following:

- the information supplied interactively to **nfdiag** when run
- details about the client machine
- details about any environment variables
- output from the following command-line utilities:
  - **enquiry**
  - **stattree**
  - **ncversions**
  - **nfkminfo**
- the contents of the following log files (if they are available):
  - **hardserver.log**
  - **keysafe.log**
  - **cmdadp.log**
  - **ncsnmpd.log**
- any attached diagnostic files
- AIX only: AIX partitioning information

Because the contents of the output file are plain text, they are human readable. You can choose to view the output file to ensure no sensitive information has been included.

**Note** The **nfdiag** utility does not capture any pass phrases in the output file.

## nfkminfo: information utility

The **nfkminfo** utility displays information about the Security World and the keys and card sets associated with it.

### Usage

---

```
nfkminfo -w|--world-info [-r|--repeat] [-p|--preload-client-id]
```

---

```
nfkminfo -k|--key-list [APPNAME [IDENT]]
```

---

```
nfkminfo -l|--name-list [APPNAME [APPNAME...]]
```

---

```
nfkminfo [-c|--cardset-list]|[-s|--softcard-list] [TOKENHASH]
```

---

```
nfkminfo --cardset-list [TOKENHASH] --key-list [APPNAME[APPNAME]]|--name-list APPNAME[IDENT...]
```

---

### Security World options

#### **-w|--world-info**

This option specifies that you want to display general information about the Security World. These options are the default and need not be included explicitly.

#### **-r|--repeat**

This option displays the information repeatedly. There is a pause at the end of each set of information. The information is displayed again when you press **Enter**.

#### **-p|--preload-client-id**

This option displays the preloaded client ID value, if any.

### Key, card set, and softcard options

#### **-k|--key-list [APPNAME[APPNAME]]**

This option lists keys without key names. If **APPNAME** is specified, only keys for these applications are listed.

**-l|--name-list [APPNAME[IDENT]]**

This option lists keys with their names. If **APPNAME** is specified, only keys for these applications are listed. If **IDENT** is listed, only the keys with the specified identifier are listed.

**-c|--cardset-list [TOKENHASH]**

If **TOKENHASH** is not specified, this option lists the card sets associated with the Security World.

The output is similar to this:

---

```
Cardset list - 1 cardsets: (P)ersistent/(N)ot, (R)emoteable/(L)ocal-only
Operator logical token hash k/n timeout name hash 1/1 none-PL name
```

---

If **TOKENHASH** is specified, these options list the details of the card identified by **hash**.

The output is similar to this:

---

```
Cardset
 name "name"
 k-out-of-n 1/1
 flags Persistent PINRecoveryForbidden(disabled) !RemoteEnabled
 timeout none
 card names ""
 hkltu hash
 gentime 2005-10-14 10:56:54
Keys protected by cardset hash:
 AppName app Ident keyident
 AppName app Ident keyident

```

---

**-s|--softcard-list [TOKENHASH]**

This option works like the **-c|--cardset-list** option, except it lists softcards instead of card sets. If **TOKENHASH** is not specified, this option lists the softcards associated with the Security World.

## Security World output info

If you run **nfkminfo** with the **-w|--world-info** option, it displays information similar to that shown in these examples:

---

```

generation 1
state 0x70000 Initialised Usable Recovery !PINRecovery
!ExistingClient !RTC !NVRAM !FTO !SEEDebug
n_modules 1
hknso hash_knso
hkm hash_km
hkmwk hash_knwk
hkcre hash_kre
hkra hash_kra
ex.client none

...

...
Module #1
generation 1
state 0x1 Useable
flags 0x10000 ShareTarget
n_slots 2
esn 34F3-9CB4-753B
hkml hash_kml
Module #1 Slot #0 IC 11
generation 1
phystype SmartCard
slotlistflags 0x2
state 0x4 Operator
flags 0x20000 RemoteEnabled
shareno 2
shares
error OK
Cardset
name "fred"
k-out-of-n 1/2
flags NotPersistent
timeout none
card names "" ""
hkltu hash_kt
Module #1 Slot #1 IC 0
generation 1
phystype SmartCard
slotlistflags 0x2 SupportsAuthentication
state 0x4 Admin
flags 0x10000 Pass phrase
shareno 1
shares LTNSO(PIN) LTM(PIN) LTR(PIN) LTNV(PIN) LTRTC(PIN) LTDSEE(PIN)
LTFTO(PIN)
error OK
No Cardset

No Pre-Loaded Objects

```

---

World

**nfkminfo** reports the following information about the Security World:

**generation**

This indicates the internal number.

**state**

This indicates the status of the current world:

**Initialised**

This indicates that the Security World has been initialized.

**Usable**

This indicates that there is at least one usable module in this Security World on this host.

**!Usable**

This indicates that there are no usable modules in this Security World on this host.

**Recovery**

This indicates that the Security World has the OCS and softcard replacement and the key recovery features enabled.

**!Recovery**

This indicates that the Security World has the OCS and softcard replacement and the key recovery features disabled.

**StrictFIPS140**

This indicates that the Security World is FIPS 140-2 level 3 compliant. If this is not shown, the Security World is level 2 compliant only.

**Note** This indicates that your Security World is compliant with the roles and services of the FIPS 140-2 level 3 standard. It is included for those customers who have a regulatory requirement for compliance.

**ExistingClient**

This indicates that there is a Client ID set, for example, by **preload**. This Client ID is given in the **ex.client** output if the **--preload-client-id** flag was supplied.

**!ExistingClient**

This indicates that no Client ID is set. The **ex.client** output will be empty.

**AlwaysUseStrongPrimes**

This indicates that the Security World always generates RSA keys in a manner compliant with FIPS 186-3.

**!AlwaysUseStrongPrimes**

This indicates that the Security World leaves the choice of RSA key generation algorithm to individual clients.

**SEEDebug**

This indicates that the Security World has an SEE Debugging delegation key.

**!SEEDebug**

This indicates the Security World has no SEE Debugging delegation key.

**SEEDebugForAll**

This indicates no authorization is required for SEE Debugging.

**PINRecovery**

This indicates that the Security World has the pass phrase replacement feature enabled.

**!PINRecovery**

This indicates that the Security World has the pass phrase replacement feature disabled.

**FTO**

This indicates that the Security World has an FTO delegation key.

**!FTO**

This indicates that the Security World has no FTO delegation key.

**NVRAM**

This indicates that the Security World has an NVRAM delegation key.

**!NVRAM**

This indicates that the Security World has no NVRAM delegation key.

**RTC**

This indicates that the Security World has an RTC delegation key.

**!RTC**

This indicates that the Security World has no RTC delegation key.



**n\_modules**

This indicates the number of nShield modules connected to this computer.

**hknso**

This indicates the SHA-1 hash of the Security Officer's key.

**hkm**

This indicates the SHA-1 hash of the Security World key.

**hkmwk**

This indicates the SHA-1 hash of a dummy key used to load the Administrator Card Set (the dummy key is the same on all modules that use Security Worlds and is not secret).

**hkcre**

This indicates the SHA-1 hash of the recovery key pair.

**hkra**

This indicates the SHA-1 hash of the recovery authorization key.

**ex.client**

This indicates the **ClientID** required to use any pre-loaded keys and tokens.

**k-out-of-n**

This indicates the values of **K** and **N** for this Security World.

**other quora**

This indicates the number (quora) of Administrator Cards (**K**) required to perform certain other functions as configured for this Security World.

**ciphersuite**

This indicates the name of the Cipher suite that the Security World uses.

**Module**

For each module in the Security World, **nfkminfo** reports:

**generation**

This indicates the version of the module data.

**state**

This indicates one of the following:

**PreInitMode**

This indicates that the module is in the pre-initialization state.

**InitMode**

This indicates that the module is in the initialization state.

**Unknown**

This indicates that the module's state could not be determined.

**Usable**

This indicates that the module is programmed in the current Security World and can be used.

**Uninitialized**

This indicates that the module does not have the Security Officer's key set and that the module must be initialized before use.

**Factory**

This indicates that the module has module key zero only and that the Security Officer's key is set to the factory default.

**Foreign**

This indicates that the module is from an unknown Security World.

**AccelOnly**

This indicates that the module is acceleration only.

**Unchecked**

This indicates that, although the module appears to be in the current Security World, **nfkminfo** could not find a module initialization certificate (a **module\_ESN** file) for this module.

**Failed**

This indicates that the module has failed.

**MaintMode**

This indicates that the module is in the maintenance state.

#### **flags**

This displays ShareTarget if the module has been initialized to allow reading of remote card sets.

#### **n\_slots**

This indicates the number of slots on the module (there is one slot for each physical smart card reader, plus one slot for each soft token and for any remote slots available).

#### **esn**

This indicates the electronic serial number of the module (if the module is not in the **Usable** state, the electronic serial number may not be available).

#### **hkml**

This indicates the hash of the module signing key (if the module is not in the **Usable** state, this value may not be available).

#### **Slot**

For each slot on the module, **nfkminfo** reports:

#### **IC**

This indicates the insertion count for this slot (which is 0 if there is no card in the slot).

#### **generation**

This indicates the version of the **slotinfo** structure.

#### **phystype**

This indicates the type of slot, which can be one of:

- **SmartCard**
- **SoftToken.**

**slotlistflags**

These are flags describing the capabilities of the slot:

**0x1**

This indicates that the slot uses a protected PIN path.

**0x2**

This indicates that the slot supports challenge-response authentication.

**state**

This can be one or more of the following flags:

**Blank**

This indicates that the smart card in the reader is unformatted.

**Admin**

This indicates that the smart card in the reader is part of the Administrator Card Set.

**Empty**

This indicates that there is no smart card in the reader.

**Error**

This indicates that the smart card in the reader could not be read (the card may be from a different Security World).

**Operator**

This indicates that the smart card in the reader is an Operator Card.

**flags**

This displays **Passphrase** if the smart card requires a pass phrase.

**shareno**

This indicates the number of the card within the card set.

**shares**

If the card in the slot is an Operator Card, no values are displayed for **shares**.

If the card in the slot is an Administrator Card, values are displayed indicating what key shares are stored on the card. Each share is prefixed with the letters **LT** (Logical Token), and the remaining letters identify the key (for example, the value **LTNSO** indicates that a share of **K<sub>NSO</sub>**, the Security Officer's key, is stored on the card).

**error**

This indicates the error status encountered if the smart card could not be read:

**OK**

This indicates that there were no errors.

**TokenAuthFailed**

This indicates that the smart card in the reader failed challenge response authentication (the card may come from a different Security World).

**PhysTokenNotPresent**

This indicates that there is no card in the reader.

If you purchased a developer kit, you can refer to the relevant developer documentation for a full list of error codes.

**Card set**

If there is an Operator Card in the reader, **nfkminfo** reports:

**name**

This indicates the name given to this card set.

**k-out-of-n**

This indicates the values of **K** and **N** for this card.

**flags**

This displays one or more of each of the following pairs of flags:

**NotPersistent**

This indicates that the Operator Card is not persistent.

**Persistent**

This indicates that the Operator Card is persistent.

**NotRemoteEnabled**

This indicates that the card in the slot is not from a Remote Operator Card Set.

**RemoteEnabled**

This indicates that the card in the slot is from a Remote Operator Card Set.

**PINRecoveryForbidden(disabled)**

This indicates that the card in the slot does not have pass phrase replacement enabled. This is always true if pass phrase replacement is disabled for the Security World.

**PINRecoveryRequired(enabled)**

This indicates that the card in the slot does have pass phrase replacement enabled.

**timeout**

the period of time in seconds after which the module automatically removes the Operator Card Set. If timeout is set to **none**, the Operator Card Set does not time out.

**card**

lists the names of the cards in the set, not all software can give names to individual cards in a set.

**hkltu**

the SHA-1 hash of the secret on the card.

## perfcheck: performance measurement checking tool

Use the **perfcheck** command-line utility to run various tests measuring the cryptographic performance of a Thales module.

**Note** Run **perfcheck** with the standard **-h|--help** option to display information about the options and parameters that control the program's behavior.

The available tests are grouped into suites:

- **kx** (key exchange)
- **keygen** (key generation)
- **signing** (signing)
- **verify** (verification)
- **enc** (encryption)
- **dec** (decryption)
- **misc** (miscellaneous).

To see the list of tests available in a particular suite, run a command of the form:

---

```
perfcheck --list suite
```

---

For example, to list all the **signing** tests, run the command:

---

```
perfcheck --list signing
>>> Suite `signing' -- Signing (222 tests)
>>> 1 - DSA using RIPEMD160 with 512-bit p and 160-bit q.
>>> 2 - DSA using RIPEMD160 with 1024-bit p and 160-bit q.
>>> 3 - DSA using RIPEMD160 with 2048-bit p and 160-bit q.
>>> 4 - DSA using RIPEMD160 with 3072-bit p and 160-bit q.
>>> ...
```

---

In the output, each listed test in the suite is identified with a number. You must supply the number of the desired test, along with its suite, in a command of the form:

---

```
perfcheck suite:test_number
```

---

For example, to use test 16 of the **signing** suite, run the command:

---

```
perfcheck signing:16
```

---

This command first produces some output containing information about operating environment. Then it produces additional output of the following form:

---

```
>>> Results: Signing
>>> Test Reps Total time (s) Mean time (s) Std dev (s) Latency (ms) Rate
>>> (signatures/s)
>>> 16 DSA using SHA224 50 22.36 0.4472 7.3 11843.7858 2.2362
>>> with 3072-bit p
>>> and 224-bit q.
>>>
>>> ...
```

---

Output from test suites, such as that in the example, includes the following information:

| Value             | Description                                                                                                                                                                                                                                                                                                                                                                                             |
|-------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Reps</b>       | This value shows the number of repetitions of the test that have been run. Generally, a larger number of repetitions produces more accurate data. You can specify the number of repetitions by setting the <b>--repetitions=REPS</b> option.                                                                                                                                                            |
| <b>Total time</b> | This value shows the total time in seconds taken to run all repetitions of the test.                                                                                                                                                                                                                                                                                                                    |
| <b>Mean time</b>  | This value shows the average time in seconds taken to run for one repetition of the test. (This value is calculated by dividing the value of <b>Total time</b> by the value of <b>Reps</b> ).                                                                                                                                                                                                           |
| <b>Std dev</b>    | This value shows the standard deviation in the <b>Mean time</b> . If this value is high, it indicates that a large number of repetitions are necessary in order for the throughput figure to be meaningful. You can suppress calculation of the standard deviation by running <b>perfcheck</b> with the <b>--no-std-dev</b> option; in some cases, setting <b>--no-std-dev</b> can increase throughput. |
| <b>Latency</b>    | This value is a measure of how long in milliseconds a single command takes to travel from the host computer to the module and back to the host computer again.                                                                                                                                                                                                                                          |
| <b>Rate</b>       | This value is a measure of throughput. It is calculated by dividing the value of <b>Reps</b> by the value of <b>Total time</b> , and displayed in terms of a unit appropriate to the type of test (for example, for <b>signing</b> tests, the number of signatures performed each second).                                                                                                              |



## How perfcheck calculates statistics

When an nCore command is submitted to a module by a client application, it is processed as follows:

- 1 The command is passed to the hardserver.
- 2 The client hardserver encrypts the command.
- 3 When the module is free, the command is submitted from the hardserver queue.
- 4 The command is executed by the module, and the reply is given to the hardserver.
- 5 The hardserver queues the reply.
- 6 When the client application is ready, the queued reply is returned to it.

Because a module can execute several commands at once, throughput is maximized by ensuring there is always at least one command in the hardserver queue (so that there are always commands available to give to the module).

The **perfcheck** utility sends multiple simultaneous nCore commands to keep the module busy. It can send more commands if a required number of repetitions has not yet been reached.

After sending some initial commands, **perfcheck** begins marking commands with the time at which are submitted; when a command comes back with a timestamp, **perfcheck** checks the amount of time needed to complete the command and updates the values for **Std dev** and **Latency**. The value of **Total time** is the amount of time from sending the first job to receiving the final one.

## stattree: information utility

The **stattree** utility returns the statistics gathered by the hardserver and modules.

### Usage

---

```
stattree [<node> [<node> [...]]]
```

---

### Output

Running the **stattree** utility displays a snapshot of statistics currently available on the host machine. Statistics are gathered both by the hardserver (relating to the server itself, and its current clients) and by each attached module.

Times are listed in seconds. Other numbers are integers, which are either real numbers, IP addresses, or counters. For example, a result **-CmdCount 74897** means that there have been 74,897 commands submitted.

A typical fragment of output from **stattree** looks like this:

---

```
+PerModule:
 +#1:
 +ModuleObjStats:
 -ObjectCount 5
 -ObjectsCreated 5
 -ObjectsDestroyed 0
 +ModuleEnvStats:
 -MemTotal 15327232
 -MemAllocKernel 126976
 -MemAllocUser 0
 +ModuleJobStats:
 -CmdCount 169780
 -ReplyCount 169778
 -CmdBytes 3538812
 -ReplyBytes 4492764
 -HostWriteCount 169772
 -HostWriteErrors 0
 -HostReadCount 437472
 -HostReadErrors 0
 -HostReadEmpty 100128
 -HostReadDeferred 167578
 -HostReadTerminated 0
 -PFNIssued 102578
 -PFNRejected 1
```

```

-PFNCompleted 102577
-ANIssued 1
-CPULoadPercent 0
+ModuleSerialStats:
 -HostReadCount 437476
 -HostReadDeferred 167580
 -HostReadReconnect 167579
 -HostReadErrors 0
 -HostWriteCount 169774
 -HostWriteErrors 0

```

---

**PerModule**, **ModuleObjStats**, and **ModuleEnvStats** are node tags that identify classes of statistics. **#1** identifies an instance node.

**ObjectCount**, **MemTotal**, and the remaining items at the same level are statistics **IDs**. Each has a corresponding value.

If **node** is provided, **stattree** uses the value given as the starting point of the tree and displays only information at or below that node in the tree. Values for **node** can be numeric or textual. For example, to view the object counts for local module number 3:

---

```

$ stattree PerModule 3 ModuleObjStats
+##PerModule:
 +##3:
 +##ModuleObjStats:
 -ObjectCount 6
 -ObjectsCreated 334
 -ObjectsDestroyed 328

```

---

The value of **node** must be a node tag; it must identify a node in the tree and not an individual statistic. Thus, the following command does not work:

---

```

$ stattree PerModule 3 ModuleObjStats ObjectCount
+##PerModule:
 +##3:
 +##ModuleObjStats:
Unable to convert 'ObjectCount' to number or tag name.

```

---

## Node tags

These hold statistics for each module:

| Category               | Contains                                                                                            |
|------------------------|-----------------------------------------------------------------------------------------------------|
| <b>ModuleJobStats</b>  | This tag holds statistics for the Security World Software commands [jobs] processed by this module. |
| <b>ModulePCISStats</b> | This tag holds statistics for the PCI connection between the module and the host computer.          |

| Category                 | Contains                                                                                                                                                                                                                                                                                                                                                                                                         |
|--------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>ModuleSerialStats</b> | This tag is for nShield Edge modules only. It holds statistics for the serial connection between the module and the host computer.                                                                                                                                                                                                                                                                               |
| <b>ServerGlobals</b>     | Aggregate statistics for all commands processed by the hardserver since it started.<br>The standard statistics (as described below) apply to the commands sent from the hardserver to modules. Commands processed internally by the server are not included here. The Uptime statistic gives the total running time of the server so far.                                                                        |
| <b>Connections</b>       | Statistics for connections between clients and the hardserver. There is one node for each currently active connection. Each node has an instance number that matches the log message generated by the server when that client connected. For example, when the hardserver message is <b>Information: New client #24 connected</b> , the client's statistics appear under node #24 in the <b>stattree</b> output. |
| <b>PerModule</b>         | Statistics kept by the modules. There is one instance node for each module, numbered using the standard module numbering. The statistics provided by each module depend on the module type and firmware version.                                                                                                                                                                                                 |
| <b>ModuleObjStats</b>    | Statistics for the module's Object Store, which contains keys and other resources. These statistics may be useful in debugging applications that leak key handles, for example.                                                                                                                                                                                                                                  |
| <b>ModuleEnvStats</b>    | General statistics for the module's operating environment.                                                                                                                                                                                                                                                                                                                                                       |

## Statistics IDs

| ID                      | Value                                                                                                                                                                                                                                                                                                                                                             |
|-------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Uptime</b>           | The length of time (in seconds) since a module was last reset, the hardserver was started, or a client connection was made.                                                                                                                                                                                                                                       |
| <b>CmdCount</b>         | The total number of commands sent for processing from a client to the server, or from the server to a module. Contains the number of commands currently being processed.                                                                                                                                                                                          |
| <b>ReplyCount</b>       | The total number of replies returned from server to client, or from module to server.                                                                                                                                                                                                                                                                             |
| <b>CmdBytes</b>         | The total length of all the command blocks sent for processing.                                                                                                                                                                                                                                                                                                   |
| <b>ReplyBytes</b>       | The total length of all the reply blocks received after completion.                                                                                                                                                                                                                                                                                               |
| <b>CmdMarshalErrors</b> | The number of times a command block was not understood when it was received. A nonzero value indicates either that the parties at each end of a connection have mismatched version numbers (for example, a more recent hardserver has sent a command to a less recent module that the module does not understand), or that the data transfer mechanism is faulty. |

| ID                        | Value                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>ReplyMarshalErrors</b> | The number of times a reply was not understood when it was received. A nonzero value indicates either that the parties at each end of a connection have mismatched version numbers (for example, a more recent hardserver has sent a command to a less recent module that the module does not understand), or that the data transfer mechanism is faulty. |
| <b>ClientCount</b>        | The number of client connections currently made to the server. This appears in the hardserver statistics.                                                                                                                                                                                                                                                 |
| <b>MaxClients</b>         | The maximum number of client connections ever in use simultaneously to the hardserver. This gives an indication of the peak load experienced so far by the server.                                                                                                                                                                                        |
| <b>DeviceFails</b>        | The number of times the hardserver has declared a device to have failed. The hardserver provides a diagnostic message when this occurs.                                                                                                                                                                                                                   |
| <b>DeviceRestarts</b>     | The number of times the hardserver has attempted to restart a module after it has failed. The hardserver provides a Notice message when this occurs. The message does not indicate that the attempt was successful.                                                                                                                                       |
| <b>QOutstanding</b>       | The number of commands waiting for a module to become available on the specified client connection. When a module accepts a command from a client, this number decreases by 1 and <b>DevOutstanding</b> increases by 1. Commands that are processed purely by the server are never included in this count.                                                |
| <b>DevOutstanding</b>     | The number of commands sent by the specified client that are currently executing on one or more modules. When a module accepts a command from a client, this number decreases by 1 and <b>QOutstanding</b> increases by 1. Commands that are processed purely by the server are never included in this count.                                             |
| <b>LongOutstanding</b>    | The number of <b>LongJobs</b> sent by the specified client that are currently executing on one or more modules. When a module accepts a <b>LongJobs</b> command from a client, this number increases by 1 and <b>QOutstanding</b> decreases by 1. Commands that are processed purely by the server are never included in this count.                      |
| <b>RemoteIPAddress</b>    | The remote IP address of a client who has this connection. A local client has the address 0.0.0.0.                                                                                                                                                                                                                                                        |
| <b>HostWriteCount</b>     | The number of write operations (used to submit new commands) that have been received by the module from the host machine. One write operation may contain more than one command block. The operation is most efficient when this is the case.                                                                                                             |
| <b>HostWriteErrors</b>    | The number of times the module rejected the write data from the host. A nonzero value may indicate that data is being corrupted in transfer, or that the hardserver/device driver has got out of sync with the module's interface.                                                                                                                        |
| <b>HostWriteBadData</b>   | Not currently reported by the module. Attempts to write bad data to the module are reflected in <b>HostWriteErrors</b> .                                                                                                                                                                                                                                  |
| <b>HostWriteOverruns</b>  | Not currently reported by the module. Write overruns are reflected in <b>HostWriteErrors</b> .                                                                                                                                                                                                                                                            |
| <b>HostWriteNoMemory</b>  | Not currently reported by the module. Write failures due to a lack of memory are reflected in <b>HostWriteErrors</b> .                                                                                                                                                                                                                                    |

| ID                        | Value                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|---------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>HostReadCount</b>      | The number of times a read operation to the module was attempted. The module can defer a read if it has no replies at the time, but expects some to be available later. Typically the module reports <b>HostReadCount</b> in two places: the number under <b>ModuleJobStats</b> counts a deferred read twice, once when it is initially deferred, and once when it finally returns some data. The number under <b>ModulePCISStats</b> counts this as one operation. |
| <b>HostReadErrors</b>     | The number of times a read to a module failed because the parameters supplied with the read were incorrect. A nonzero value here typically indicates some problem with the host interface or device driver.                                                                                                                                                                                                                                                         |
| <b>HostReadEmpty</b>      | The number of times a read from the module returned no data because there were no commands waiting for completion. In general, this only happens infrequently during module startup or reset. It can also happen if <b>PauseForNotifications</b> is disabled.                                                                                                                                                                                                       |
| <b>HostReadUnderruns</b>  | Not currently reported by the module.                                                                                                                                                                                                                                                                                                                                                                                                                               |
| <b>HostReadDeferred</b>   | The number of times a read operation to the module was suspended because it was waiting for more replies to become available. When the module is working at full capacity, a sizeable proportion of the total reads are likely to be deferred.                                                                                                                                                                                                                      |
| <b>HostReadTerminated</b> | The number of times a module had to cancel a read operation which has been deferred. This normally happens only if the clear key is pressed while the module is executing commands. Otherwise it might indicate a device driver, interface, or firmware problem.                                                                                                                                                                                                    |
| <b>PFNIssued</b>          | The number of <b>PauseForNotifications</b> commands accepted by the module from the hardserver. This normally increases at a rate of roughly one every two seconds. If the hardserver has this facility disabled (or a very early version), this does not occur.                                                                                                                                                                                                    |
| <b>PFNRejected</b>        | The number of <b>PauseForNotifications</b> commands rejected by the module when received from the hardserver. This can happen during module startup or reset, but not in normal use. It indicates a hardserver bug or configuration problem.                                                                                                                                                                                                                        |
| <b>PFNCompleted</b>       | The number of <b>PauseForNotifications</b> commands that have been completed by the module. Normally, this is one less than the <b>PFNIssued</b> figure because there is normally one such command outstanding.                                                                                                                                                                                                                                                     |
| <b>ANIssued</b>           | The number of <b>Asynchronous Notification</b> messages issued by the module to the hardserver. These messages indicate such things as the clear key being pressed and the module being reset. In later firmware revisions inserting or removing the smartcard or changing the non-volatile memory also generate asynchronous notifications.                                                                                                                        |
| <b>ChanJobsIssued</b>     | The number of fast channel jobs issued to the module. The fast channel facility is unsupported on current modules. This number should always be <b>0</b> .                                                                                                                                                                                                                                                                                                          |
| <b>ChanJobsCompleted</b>  | The number of fast channel jobs completed by the module. The fast channel facility is unsupported on current modules. This number should always be <b>0</b> .                                                                                                                                                                                                                                                                                                       |

| ID                       | Value                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|--------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>CPUloadPercent</b>    | The current processing load on the module, represented as a number between 0 and 100. Because a module typically contains a number of different types of processing resources (for example, main CPU, and RSA acceleration), this figure is hard to interpret precisely. In general, modules report 100% CPU load when all RSA processing capacity is occupied; when performing non-RSA tasks the main CPU or another resource (such as the random number generator) can be saturated without this statistic reaching 100%. |
| <b>HostIRQs</b>          | On PCI modules, the total number of interrupts received from the host. On current modules, approximately equal to the total of <b>HostReadCount</b> and <b>HostWriteCount</b> .                                                                                                                                                                                                                                                                                                                                             |
| <b>ChanJobErrors</b>     | The number of low-level (principally data transport) errors encountered while processing fast channel jobs. Should always be 0 on current modules.                                                                                                                                                                                                                                                                                                                                                                          |
| <b>HostDebugIRQs</b>     | On PCI modules, the number of debug interrupts received. This is used only for driver testing, and should be 0 in any production environment.                                                                                                                                                                                                                                                                                                                                                                               |
| <b>HostUnhandledIRQs</b> | On PCI modules, the number of unidentified interrupts from the host. If this is nonzero, a driver or PCI bus problem is likely.                                                                                                                                                                                                                                                                                                                                                                                             |
| <b>HostReadReconnect</b> | On PCI modules, the number of deferred reads that have now completed. This should be the same as <b>HostReadDeferred</b> , or one less if a read is currently deferred.                                                                                                                                                                                                                                                                                                                                                     |
| <b>ObjectsCreated</b>    | The number of times a new object has been put into the object store. This appears under the module's <b>ModuleObjStats</b> node.                                                                                                                                                                                                                                                                                                                                                                                            |
| <b>ObjectsDestroyed</b>  | The number of items in the module's object store that have been deleted and their corresponding memory released.                                                                                                                                                                                                                                                                                                                                                                                                            |
| <b>ObjectCount</b>       | The current number of objects (keys, logical tokens, buffers, SEE Worlds) in the object store. This is equal to <b>ObjectsCreated</b> minus <b>ObjectsDestroyed</b> . An empty module contains a small number of objects that are always present.                                                                                                                                                                                                                                                                           |
| <b>CurrentTempC</b>      | The current temperature (in degrees Celsius) of the module main circuit board. First-generation modules do not have a temperature sensor and do not return temperature statistics.                                                                                                                                                                                                                                                                                                                                          |
| <b>MaxTempC</b>          | The maximum temperature recorded by the module's temperature sensor. This is stored in non-volatile memory, which is cleared only when the unit is initialized. First-generation modules do not have a temperature sensor and do not return temperature statistics.                                                                                                                                                                                                                                                         |
| <b>MinTempC</b>          | The minimum temperature recorded by the module's temperature sensor. This is stored in non-volatile memory, which is cleared only when the unit is initialized. First-generation modules do not have a temperature sensor and do not return temperature statistics.                                                                                                                                                                                                                                                         |
| <b>MemTotal</b>          | The total amount of RAM (both allocated and free) available to the module. This is the installed RAM size minus various fixed overheads.                                                                                                                                                                                                                                                                                                                                                                                    |

| ID                    | Value                                                                                                                                                                                                                                                                                                                                         |
|-----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>MemAllocKernel</b> | The total amount of RAM allocated for kernel (that is, non-SEE) use in a module. This is principally used for the object store (keys, logical tokens, and similar) and for big-number buffers.                                                                                                                                                |
| <b>MemAllocUser</b>   | The total amount of RAM allocated for user-mode processes in the module (0 for non-SEE use). This includes the size of the SEE Machine image, and the total heap space available to it. The module's kernel does not know (and does not report in the statistics) how much of the user-mode's heap is currently free, and how much is in use. |

## How data is affected when a module loses power and restarts

Thales modules use standard RAM to store many kinds of data, and data stored in such RAM is lost in the event that a module loses power (either intentionally, because you turned off power to it, or accidentally because of a power failure).

Therefore, after restoring power to a module, you must reload any keys that had been loaded onto it before it lost power. After reloading, the **KeyIDs** are different.

Likewise, after restoring power to a module, you must reload any cards that were loaded onto it before it lost power.

However, data stored in NVRAM is unaffected when a module loses power.

**Note** If you are using multiple Thales modules in the same Security World, and have the same key (or keys) loaded onto each module as part of a load-sharing configuration, loss of power to one module does not affect key availability (as long as at least one other module onto which the keys are loaded remains operational). However, in such a multiple-module system, after restoring power to a module, you must still reload any keys to that module before they can be available from that module.





## Appendix F: Hardserver configuration files

The hardserver configuration file has the following sections that you can update to configure the hardserver on an nShield module. If a section is not present, it is assumed to have no entries.

### Hardserver configuration files

Hardserver configuration files are text files. They must contain only characters with ASCII values between 32 and 127, and the tab, line break, and return characters.

Lines starting with a # character are comments and are ignored. Some comments that document the configuration options are generated by the configuration process. You can add your own comments, but in some cases they may later be overwritten.

A hardserver configuration file begins with a single line that specifies the version of the file syntax. This syntax-version line has the format:

---

```
syntax-version=n
```

---

In this syntax-version line example, *n* represents the version of the syntax in which the file is written. The system can process a file with a lower syntax version than the one it uses, but not one with a higher version.

After the syntax-version line, the rest of the configuration file consists of sections that can be edited to control different aspects of hardserver behavior. Each section begins with its name in square brackets, as in this example:

---

```
[slot_imports]
```

---

You can update the parameters defined in most of these sections to configure the way that the hardserver handles secure transactions between modules connected to the host computer and applications that run on the host computer.

**Note** Some sections are updated automatically and should not be edited manually. For more information, see the descriptions of individual sections.

In each section, the bracketed name is followed by a specified set of fields. Each field is on a separate line. Each field begins with its name, followed by an equals sign (=) and a value of the appropriate type. White space can be included at either end of the line (for example, in order to indent lines as an aid to clarity).

Some types of field are grouped into entries. An entry is a set of fields of different types that define an instance of an object (for example, a particular client as distinct from other clients). Entries in the same section are separated by a line that contains one or more hyphens (-). Blank lines and comments are allowed between the fields in an entry.

Strings are case sensitive in the section names and field names.

If a particular section is not present in the configuration file, it is assumed to have no entries.

## General hardserver configuration settings

### server\_settings

The **server\_settings** section defines the settings for the client hardserver you can modify while the hardserver is running.

**Note** These flags are used by the **NFLOG\_DETAIL** environment variable [see [Environment variables to control logging](#) on page 249].

The section contains the following fields:

| Field            | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>loglevel</b>  | <p>This field specifies the level of logging performed by the hardserver. It takes a value that is one of the following:</p> <ul style="list-style-type: none"> <li>• <b>info</b></li> <li>• <b>notice</b></li> <li>• <b>client</b></li> <li>• <b>remoteserver</b></li> <li>• <b>error</b></li> <li>• <b>serious</b></li> <li>• <b>internal</b></li> <li>• <b>startup</b></li> <li>• <b>fatal</b></li> <li>• <b>fatalinternal</b></li> </ul> <p>The default is <b>info</b>. For more information, see Appendix E: <a href="#">Logging, debugging, and diagnostics</a>.</p> <p>If the <b>NFAST_SERVERLOGLEVEL</b> environment variable is set, it overrides any <b>loglevel</b> value set in the configuration file.</p> |
| <b>logdetail</b> | <p>This field specifies the level of detail logged by the hardserver. You can supply one or more flags in a space-separated list. For more information about the flags, see the table below.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |

| Field                        | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>connect_retry</b>         | This field specifies the number of seconds to wait before retrying a remote connection to a client hardserver. The default is 10.                                                                                                                                                                                                                                                                                                                                 |
| <b>connect_broken</b>        | This field specifies the number of seconds of inactivity allowed before a connection to a client hardserver is declared broken. The default is 90.                                                                                                                                                                                                                                                                                                                |
| <b>connect_keepalive</b>     | This field specifies the number of seconds between <b>keepalive</b> packets for remote connections to a client hardserver. The default is 10.                                                                                                                                                                                                                                                                                                                     |
| <b>connect_command_block</b> | When the module has failed, this field specifies the number of seconds the hardserver should wait before failing commands directed to that module with a <b>NetworkError</b> message. For commands to have a chance of succeeding after the module has failed this value should be greater than that of <b>connect_retry</b> . If it is set to 0, commands to a module are failed with <b>NetworkError</b> immediately, as soon as the module. The default is 35. |
| <b>max_pci_if_vers</b>       | This field specifies the maximum PCI interface version number. If <b>max_pci_if_vers</b> is set to 0 (the default), there is no limit.                                                                                                                                                                                                                                                                                                                            |

These flags are those used by the **NFLOG\_DETAIL** environment variable (see [Environment variables to control logging](#) on page 249).

You can supply a number of flags with the **logdetail** field, which specifies the level of detail logged by the hardserver (see the table above). Supply the flags in a space separated list:

| Flag                   | Description                                                                                                                                                                                                        |
|------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>external_time</b>   | This flag specifies the external time (that is, the time according to your machine's local clock) with the log entry.                                                                                              |
| <b>external_date</b>   | This flag specifies the external date (that is, the date according to your machine's local clock) with the log entry.                                                                                              |
| <b>external_tid</b>    | This flag specifies the external thread ID with the log entry.                                                                                                                                                     |
| <b>external_time_t</b> | This flag specifies the external <b>time_t</b> (that is, the time in machine clock ticks rather than local time) with the log entry.                                                                               |
| <b>stack_backtrace</b> | This flag specifies the stack backtrace with the log entry.                                                                                                                                                        |
| <b>stack_file</b>      | This flag specifies the stack file with the log entry.                                                                                                                                                             |
| <b>stack_line</b>      | This flag specifies the message line number in the original library. This flag is likely to be most useful in conjunction with example code we have supplied that has been written to take advantage of this flag. |
| <b>msg_severity</b>    | This flag specifies the message severity (a severity level as used by the <b>NFLOG_SEVERITY</b> environment variable) with the log entry.                                                                          |
| <b>msg_categories</b>  | This flag specifies the message category (a category as used by the <b>NFLOG_CATEGORIES</b> environment variable) with the log entry.                                                                              |
| <b>msg_writeable</b>   | This flag specifies message writeables, and extra information that can be written to the log entry, if any such exist.                                                                                             |
| <b>msg_file</b>        | This flag specifies the message file in the original library. This flag is likely to be most useful in conjunction with example code we have supplied that has been written to take advantage of this flag.        |

| Flag               | Description                                                                                                                                                                                                        |
|--------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>msg_line</b>    | This flag specifies the message line number in the original library. This flag is likely to be most useful in conjunction with example code we have supplied that has been written to take advantage of this flag. |
| <b>options_utc</b> | This flag showing the date and time in UTC (Coordinated Universal Time) instead of local time.                                                                                                                     |

## module\_settings

The **module\_settings** section defines the settings for the module that can be changed while the hardserver is running. The section contains the following fields:

| Field           | Description                                                                                                                                       |
|-----------------|---------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>esn</b>      | This field specifies the electronic serial number of the module.                                                                                  |
| <b>priority</b> | This field specifies the priority of the module. The value for this field can be an integer from 1 (highest) to 100 (lowest). The default is 100. |

## server\_remotecomms

The **server\_remotecomms** section defines the remote communication settings for the client hardserver. These are read only at hardserver start-up. This section contains the following fields:

| Field              | Description                                                                                                                                                                                                                |
|--------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>impath_port</b> | This field specifies the port on which the hardserver listens for incoming impath connections. The default is 9004. Setting this field to <b>0</b> specifies that the hardserver does not listen for incoming connections. |

## server\_startup

The **server\_startup** section defines the settings for the hardserver that are loaded at start-up. Any changes you make to the settings in this section do not take effect until after you restart the hardserver. For more information, see [Stopping and restarting the hardserver](#) on page 70.

The section contains the following fields:

| Field                       | Description                                                                                                                                                                                                                                                                                                                                                                                                        |
|-----------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>unix_socket_name</b>     | This field specifies the name of the UNIX socket to use for non-privileged connections on UNIX. The default is <code>/dev/nfast/nserver</code> .<br>If the <b>NFAST_SERVER</b> environment variable is set, it overrides any value set for <b>unix_socket_name</b> in the hardserver configuration file. For more information about environment variables, see Appendix D: <a href="#">Environment variables</a> . |
| <b>unix_privsocket_name</b> | This field specifies the name of the UNIX socket to use for privileged connections on UNIX. The default is <code>/dev/nfast/privnserver</code> .<br>If the <b>NFAST_PRIVSERVER</b> environment variable is set, it overrides any value set for <b>unix_privsocket_name</b> in the hardserver configuration file.                                                                                                   |
| <b>nt_pipe_name</b>         | This field is not used on Unix systems.                                                                                                                                                                                                                                                                                                                                                                            |
| <b>nt_pipe_users</b>        | This field is not used on Unix systems.                                                                                                                                                                                                                                                                                                                                                                            |
| <b>nt_privpipe_name</b>     | This field is not used on Unix systems.                                                                                                                                                                                                                                                                                                                                                                            |
| <b>nt_privpipe_users</b>    | This field is not used on Unix systems.                                                                                                                                                                                                                                                                                                                                                                            |
| <b>nonpriv_port</b>         | This field specifies the port on which the hardserver listens for local non-privileged TCP connections. The value <b>0</b> (which is the default) specifies none. Java clients default to connecting to port 9000.<br>If the <b>NFAST_SERVER_PORT</b> environment variable is set, it overrides any value set for <b>nonpriv_port</b> in the hardserver configuration file.                                        |
| <b>priv_port</b>            | This field specifies the port on which the hardserver listens for local privileged TCP connections. The value <b>0</b> (which is the default) specifies none. Java clients default to connecting to port 9001.<br>If the <b>NFAST_SERVER_PRIVPORT</b> environment variable is set, it overrides any value set for <b>priv_port</b> in the hardserver configuration file.                                           |

## load\_seemachine

The **load\_seemachine** section of the hardserver configuration file defines SEE machines that the module should load and, if required, start for use by other clients. Each SEE machine is defined by the following fields:

| Field                  | Description                                                                                                                                                                                                                                                        |
|------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>module</b>          | This field specifies the module on to which to load the SEE machine. The value must be an integer. A module with this ID must be configured on the client computer.                                                                                                |
| <b>machine_file</b>    | This field specifies the file name of the SEE machine.                                                                                                                                                                                                             |
| <b>userdata</b>        | This field specifies the <b>userdata</b> file name to pass to the SEE machine on start-up. If this field is blank (" "), the SEE machine is loaded but not started. By default, this field is blank.                                                               |
| <b>worldid_pubname</b> | This field specifies the <b>PublishedObject</b> name to use for publishing the <b>KeyID</b> of the started SEE machine. If this field is blank (" "), the <b>KeyID</b> is not published. This field is ignored if the value of the <b>userdata</b> field is blank. |

| Field                | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>postload_prog</b> | This field specifies the program to run after loading the SEE machine in order to perform any initialization required by the SEE machine or its clients. The specified program must accept an argument of the form - <b>m module#</b> .<br>To run <b>see-sock-serv</b> directly on the nShield Connect, set this field to <b>sockserv</b> .                                                                                                                                                                                                                                                           |
| <b>postload_args</b> | This field specifies arguments to pass to the program specified by the <b>postload_prog</b> field. The argument - <b>m module#</b> is automatically passed as the first argument. The <b>postload_args</b> field is ignored if <b>postload_prog</b> is not specified or is blank.<br>To run <b>see-sock-serv</b> directly on the nShield Connect, set this field to - <b>p pubname</b> .                                                                                                                                                                                                              |
| <b>pull_rfs</b>      | This field specifies whether the SEE machine name and userdata should be pulled from the RFS. The default is <b>0</b> ; set to <b>1</b> to pull the SEE machine and userdata from the RFS before loading on the remote module.<br>This field will be ignored if set on client machine configurations.<br><br><div> <p>Note</p> <p>This field will not be added to existing configuration files if you are upgrading an image. If you require the new functionality enabled by this field, you can add the field to the <b>load_seemachine</b> section of your existing configuration file.</p> </div> |

## slot\_imports

The **slot\_imports** section defines slots from remote modules that will be available to the local computer. Each slot is defined by the following fields:

| Field                | Description                                                                                                                                      |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>local_esn</b>     | This field specifies the ESN of the local module importing the slot.                                                                             |
| <b>local_slotid</b>  | This field specifies the <b>SlotID</b> to use to refer to the slot when it is imported on the local module. The default is 2.                    |
| <b>remote_ip</b>     | This field specifies the IP address of the machine that hosts the slot to import.                                                                |
| <b>remote_port</b>   | This field specifies the port number on which the remote hardserver is listening.                                                                |
| <b>remote_esn</b>    | This field specifies the ESN of the remote module from which to import the slot.                                                                 |
| <b>remote_slotid</b> | This field specifies the <b>SlotID</b> of the slot to import on the remote module. The value of this field must be an integer. The default is 0. |

## slot\_exports

The **slot\_exports** section defines the slots on local modules that the local hardserver should allow network modules to import. Each local slot has an entry for each remote module that can import it, consisting of the following fields:

| Field               | Description                                                                                                                |
|---------------------|----------------------------------------------------------------------------------------------------------------------------|
| <b>local_esn</b>    | This field specifies the ESN of the local module whose slot can be imported by a network module.                           |
| <b>local_slotid</b> | This field specifies the <b>SlotID</b> of the slot that is to be imported. The value must be an integer. The default is 0. |
| <b>remote_ip</b>    | This field specifies the IP address of the module that is allowed to import the slot.                                      |
| <b>remote_esn</b>   | This field specifies the ESN of the module allowed to import the slot.                                                     |

## Sections only in client configuration files

### nethsm\_imports

The **nethsm\_imports** section defines the network modules that the client imports. It can also be set up by the **nethsmenroll** utility. Each module is defined by the following fields:

| Field                           | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|---------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>local_module</b>             | This field specifies the ModuleID to assign to the imported module. The value must be an integer. A module with this ID must not be already configured on the client computer.                                                                                                                                                                                                                                                                                                                                                                                                              |
| <b>remote_ip</b>                | This field specifies the IP address of the module to import.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| <b>remote_port</b>              | This field specifies the IP address of the module to import.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| <b>remote_esn</b>               | This field specifies the ESN of the imported module.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| <b>keyhash</b>                  | This field specifies the hash of the key that the module should use to authenticate itself.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| <b>timelimit</b>                | This field specifies the time interval in seconds between renegotiations of the session key. The value <b>0</b> means no limit. The default is <b>60*60*24</b> seconds (that is, one day).                                                                                                                                                                                                                                                                                                                                                                                                  |
| <b>datalimit</b>                | This field specifies the amount of data in bytes to encrypt with a given session key before renegotiating a new one. The value <b>0</b> means no limit. The default is <b>1024*1024*8</b> bits (that is, 8 megabytes).                                                                                                                                                                                                                                                                                                                                                                      |
| <b>privileged</b>               | The value in this field specifies whether the client can make a privileged connection to the module. The default is <b>0</b> , which specifies no privileged connections. Any other value specifies privileged connections.                                                                                                                                                                                                                                                                                                                                                                 |
| <b>privileged_use_high_port</b> | <p>The value in this field specifies whether the client uses a higher port number for privileged connections. The value <b>1</b> specifies that high-numbered ports are used for privileged connections. The default is <b>0</b>; valid values are <b>0</b> or <b>1</b>. Only used when <b>privileged=1</b>.</p> <p><b>Note</b> This field will not be added to existing configuration files if you are upgrading an image. If you require the new functionality enabled by this field, you can add the field to the <b>nethsm_imports</b> section of your existing configuration file.</p> |
| <b>ntoken_esn</b>               | This field specifies the ESN of this client's nToken module, if an nToken is installed.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |



The default value for **remote\_keyhash** (40 zeros) specifies that no authentication should occur. We recommend that you set a specific key hash in place of this default.



## rfs\_sync\_client

This section defines which remote file system the client should use to synchronize its key management data:

| Field              | Description                                                                                                                 |
|--------------------|-----------------------------------------------------------------------------------------------------------------------------|
| <b>remote_ip</b>   | The IP address of the remote server against which to synchronize.                                                           |
| <b>remote_port</b> | Which port to connect to on the RFS server. The default is 9004.                                                            |
| <b>use_kneti</b>   | Setting this option to <b>yes</b> requires the client to use an authenticated channel to communicate with the RFS server.   |
| <b>local_esn</b>   | This is only required if <b>use_kneti</b> is set to <b>yes</b> . It is the ESN of the local module used for authentication. |

## remote\_file\_system



This section is updated automatically when the **rfs-setup** utility is run. Do not edit it manually.

The **remote\_file\_system** section defines a remote file system on the client by listing the modules allowed to access the file system on this client. Each module is defined by an entry consisting of the following fields:

| Field              | Description                                                                                                                                                                         |
|--------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>remote_ip</b>   | This field specifies the IP address of the remote module that is allowed to access the file system on this client.                                                                  |
| <b>remote_esn</b>  | This field specifies the ESN of the remote module allowed to access the file system on this client.                                                                                 |
| <b>keyhash</b>     | This field specifies the hash of the key with which the client must authenticate itself to the module. The default is 40 zeros, which means that no key authentication is required. |
| <b>native_path</b> | This field specifies the local file name for the volume to which this entry corresponds.                                                                                            |
| <b>volume</b>      | This field specifies the volume that the remote host would access to use this entry.                                                                                                |
| <b>allow_read</b>  | If this field is set to <b>yes</b> , it means that a remote server is allowed to read the contents of the file. The default is <b>no</b> .                                          |
| <b>allow_write</b> | If this field is set to <b>yes</b> , it means that a remote server is allowed to write to the file. The default is <b>no</b> .                                                      |
| <b>allow_list</b>  | If this field is set to <b>yes</b> , it means that a remote server is allowed to list the contents of the file. The default is <b>no</b> .                                          |

| Field               | Description                                                                                                                        |
|---------------------|------------------------------------------------------------------------------------------------------------------------------------|
| <b>is_directory</b> | If this field is set to <b>yes</b> , it means that this entry represents a directory. The default is <b>no</b> .                   |
| <b>is_text</b>      | If this field is set to <b>yes</b> , it means that line endings should be converted to and from the Unix convention for transfers. |



## Appendix G: Cryptographic algorithms

### Symmetric algorithms

In the following tables, the column labelled **nfmverify** documents the implemented behavior of the **nfmverify** command-line utility (and its underlying library). An entry **N** in this column means that **nfmverify** always rejects the algorithm, while **Y** means that **nfmverify** always accepts the algorithm. A constraint on the number of bits means that **nfmverify** only accepts keys of at least that many bits.

| Symmetric Algorithms |               |                    |               |                    |            |       |           |
|----------------------|---------------|--------------------|---------------|--------------------|------------|-------|-----------|
| Algorithm            | FIPS approval | Key length in bits | Use for ...   |                    |            |       | nfmverify |
|                      |               |                    | Authorization | Digital Signatures | Encryption | Other |           |
| AES                  | Y             | 128, 192, 256      | Y             |                    | Y          |       | Y         |
| Arcfour              |               | Any                |               |                    | Y          |       | N         |
| ARIA                 | N             | 128, 192, 256      |               |                    | Y          |       |           |
| Camellia             | N             | 128, 192, 256      |               |                    | Y          |       |           |
| CAST 6               |               | up to 256          | Y             |                    | Y          |       | N         |
| DES                  |               | 56                 | Y             |                    | Y          |       | N         |
| MD5 HMAC             |               | 8 to 2048          | Y             |                    |            |       | N         |
| RIPEMD160 HMAC       |               | 8 to 2048          | Y             |                    |            |       | ≥80 bits  |
| SEED                 |               | 128                | Y             |                    | Y          |       | N         |
| SHA-1 HMAC           | Y             | 8 to 2048          | Y             |                    |            |       | ≥80 bits  |
| SHA-224 HMAC         | Y             | 8 to 2048          | Y             |                    |            |       | ≥80 bits  |
| SHA-256 HMAC         | Y             | 8 to 2048          | Y             |                    |            |       | ≥80 bits  |
| SHA-384 HMAC         | Y             | 8 to 2048          | Y             |                    |            |       | ≥80 bits  |
| SHA-512 HMAC         | Y             | 8 to 2048          | Y             |                    |            |       | ≥80 bits  |
| Tiger HMAC           |               | 8 to 2048          | Y             |                    |            |       | ≥80 bits  |
| Triple DES           | Y             | 112, 168           | Y             |                    | Y          |       | Y         |

## Asymmetric algorithms

| Asymmetric Algorithms |               |                    |               |                    |            |              |                                                        |
|-----------------------|---------------|--------------------|---------------|--------------------|------------|--------------|--------------------------------------------------------|
| Algorithm             | FIPS approval | Key length in bits | Use for ...   |                    |            |              | nfmverify                                              |
|                       |               |                    | Authorization | Digital Signatures | Encryption | Other        |                                                        |
| RSA                   | Y             | ≥512               | Y             | Y                  | Y          |              | ≥1024 bits                                             |
| Diffie-Hellman        | Y             | ≥1024              |               |                    |            | Key exchange | ≥1024 bits                                             |
| DSA                   | Y             | 512 to 3072        | Y             | Y                  |            |              | ≥1024 bits                                             |
| El-Gamal              |               | ≥1024              |               |                    | Y          |              |                                                        |
| KCDSA                 |               | ≥1024              | Y             | Y                  |            |              | ≥1024 bits                                             |
| ECDSA                 | Y             | ≥160               | Y             | Y                  |            |              | all named/custom curves with cofactor ≤4 and order 160 |
| ECDH                  | Y             | ≥160               |               |                    |            | Key exchange | all named/custom curves with cofactor ≤4 and order 160 |

Note The RSA algorithm is based on the factorization of integers.

Note The Diffie-Hellman, DSA, El-Gamal, and KCDSA algorithms are all based on discrete logarithms in a multiplicative group of a finite field.

## FIPS information

The latest guidance from the National Institute of Standards and Technology (NIST) is that

- a module is only operating in FIPS mode when it uses NIST-approved algorithms

- a module operating at FIPS 140-2 level 3 must not only indicate whether it is in FIPS mode but also actively prevent use of algorithms not approved by NIST in FIPS-approved mode.

Thus, with the current release of firmware, when a module is initialized into FIPS 140-2 level 3 mode, you are only offered NIST-approved algorithms. If you have a Security World created to comply with FIPS 140-2 level 3 and have any protocols that use algorithms not approved by NIST, you must either migrate to a FIPS 140-2 level 2 Security World or change your protocols. If you have a Security World created to comply with FIPS 140-2 level 3 and have existing long-term keys for unapproved algorithms, then these keys cannot be used with the current firmware. In such a case, we recommend that you either migrate your Security World to a FIPS 140-2 level 2 Security World or replace these keys with approved keys before upgrading to the current firmware.

These changes do not affect Security Worlds that were created to comply with FIPS 140-2 level 2, nor do they affect systems that use the nShield module to protect long-term keys but perform encryption with session keys on the host (as is the case with the nCipher MSCAPI, CHIL, and PKCS #11 libraries).

The NIST-approved algorithms that nShield modules offer are:

- DSA
- ECDSA
- RSA
- Diffie-Hellman
- Triple DES
- AES
- SHA-1
- SHA-224
- SHA-256
- SHA-384
- SHA-512
- TLS key derivation
- HMAC (SHA-1, SHA-224, SHA-256, SHA-384, SHA-512)

The algorithms *not* approved by NIST that nShield modules offer are:

- KCDSA
- ECDH
- ARIA
- Camellia
- Arc Four
- CAST 6
- DES
- SEED
- SHA-160
- MD2
- MD5
- RIPEMD 160
- HMAC (MD5, RIPEMD160)



## Appendix H: Key generation options and parameters

This appendix describes the various options and parameters that you can set when running the **generatekey** utility to control the application type and other properties of a key being generated.

**Note** For information about generating keys with the **generatekey** utility, see [Generating keys on the command line](#) on page 201 in Chapter 9: [Working with keys](#).

### Key application type (*APPNAME*)

The *APPNAME* parameter specifies the name of the application for which **generatekey** can generate keys. Specifying an application can restrict your choice of key type. A value for *APPNAME* must follow any *OPTIONS* and must precede any parameters specified for the key:

| Parameter     | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|---------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>simple</b> | Specifying the <b>simple</b> application type generates an nShield-native key. No special action is taken after the key is generated.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| <b>custom</b> | Specifying the <b>custom</b> application type generates a key for custom applications that require the key blob to be saved in a separate file. Specifying <b>custom</b> also causes the generation of a certificate request and self-signed certificate. However, we recommend that you specify the <b>simple</b> [instead of <b>custom</b> ] application type whenever possible.                                                                                                                                                                                                                                                                                                                  |
| <b>pkcs11</b> | <p>Specifying the <b>pkcs11</b> application type generates keys that are formatted for use with PKCS #11 applications and are given a suitable identifier. The set of possible supported key types is currently limited to:</p> <ul style="list-style-type: none"><li>• <b>DES3</b></li><li>• <b>DH</b></li><li>• <b>DSA</b></li><li>• <b>ECDH</b></li><li>• <b>ECDSA</b></li><li>• <b>HMACSHA1</b></li><li>• <b>RSA</b></li><li>• <b>Rijndael (AES)</b></li></ul> <p>Some key types are only available if the features that support them have been enabled for the module, if the Security World is not compliant with FIPS 140-2 level 3, or if you do not set the <b>--no-verify</b> option.</p> |

| Parameter       | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|-----------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>embed</b>    | <p>Specifying the <b>embed</b> application type generates a key for use with CHIL applications that:</p> <ul style="list-style-type: none"> <li>do not support <b>hwcrhk</b> key storage</li> <li>have a key importation facility capable of reading PEM-format RSA key files.</li> </ul> <p>Specify the <b>hwcrhk</b> application type for CHIL applications that support <b>hwcrhk</b> key storage.</p> <p>You can use a key of the <b>embed</b> application type like a PEM-format RSA/DSA key file, even though it is really a specially encoded reference to a key stored in <b>opt/nfast/kmdata/local</b>. This allows you to use an <b>embed</b> key when integrating with applications that normally require software RSA keys. For example, you can supply an <b>embed</b> key to the patched version of OpenSSL we have provided so that it uses the module to access the key rather than using its own built-in RSA operations.</p> |
| <b>hwcrhk</b>   | <p>Specifying the <b>hwcrhk</b> application type generates a key for Cryptographic Hardware Interface Library (CHIL) applications that do not require <b>embed</b> keys. Only <b>RSA</b>, <b>DSA</b>, and <b>DH</b> key types are supported</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| <b>kpm</b>      | <p>Specifying the <b>kpm</b> application type generates a key for delivery by an nForce Ultra key server. The <b>generatekey</b> utility automatically creates a special ACL entry that permits a <b>kpm</b> to be delivered to an nForce Ultra's enrolled internal hardware security module.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| <b>seeinteg</b> | <p>Specifying the <b>seeinteg</b> application type generates an SEE integrity key. The DSA, RSA, ECDSA and KCDSA algorithms are supported. SEE integrity keys are always protected by an OCS and cannot be imported. You cannot retarget an existing key as an SEE integrity key.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| <b>seeconf</b>  | <p>Specifying the <b>seeconf</b> application type generates an SEE confidentiality key. Both the Triple DES and AES algorithms are supported for this key type. SEE confidentiality keys are module-protected by default and cannot be imported. You cannot retarget an existing key as an SEE confidentiality key.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |

## Key properties (*NAME=VALUE*)

The *NAME=VALUE* syntax is used to specify the properties of the key being generated.

**Note** If a parameter's argument contains spaces, you must enclose the argument within quotation marks (" ").



You can supply an appropriate **VALUE** for the following **NAME** options:

| Option                  | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|-------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>alias</b>            | The <b>VALUE</b> for <b>alias</b> specifies an alias to assign to the key.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| <b>blobsavefile</b>     | When using the <b>custom</b> application type, the <b>VALUE</b> for <b>blobsavefile</b> specifies a file name of the form <b>FILENAME.EXT</b> to which the key blob is saved. Additionally, a text file containing information about the key is saved to a file whose name has the form <b>ROOT_inf.txt</b> ; for asymmetric key types, the public key blob is also saved to a file whose name has the form <b>ROOT_pub.EXT</b> .                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| <b>cardset</b>          | The <b>VALUE</b> for <b>cardset</b> specifies an OCS that is to protect the key (if <b>protect</b> is set to <b>token</b> ). In interactive mode, if you do not specify an OCS, you are prompted to select one at card-loading time. The default is the OCS to which the card currently inserted in the slot belongs (or the first one returned by <b>nfkminfo</b> ).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| <b>certreq</b>          | Setting <b>certreq</b> enables you to generate a certificate request when generating a PKCS #11 key (RSA keys only). The default behavior is to not generate a certificate request.<br>To generate a certificate request you must set the <b>VALUE</b> for <b>certreq</b> to <b>yes</b> , which makes <b>generatekey</b> prompt you to fill in the extra fields required to generate a key with a certificate request. The resultant certificate request is saved to the current working directory with a file name of the form <b>FILENAME_req.ext</b> (where <b>FILENAME</b> is a name of your choice). An extra file with a name of the form <b>FILENAME.ext</b> is also generated for use as a pseudo-key-header. This file can be removed after the certificate request has been generated. You can use <b>certreq</b> with the <b>--retarget</b> option to generate a self-signed certificate for an existing key. |
| <b>checks</b>           | For RSA key generation only, this specifies the number of checks to be performed. Normally, you should leave <b>VALUE</b> empty to let the module pick an appropriate default.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| <b>curve</b>            | For ECDH and ECDSA key generation only, the <b>VALUE</b> for <b>curve</b> specifies which curves from the supported range to use. Supported curves are: NISTP192, NISTP224, NISTP256, NISTP384, NISTP521, NISTB163, NISTB233, NISTB283, NISTB409, NISTB571, NISTK163, NISTK233, NISTK283, NISTK409, NISTK571, ANSIB163v1, ANSIB191v1, and SECP160r1.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <b>embedconvfile</b>    | The <b>VALUE</b> for <b>embedconvfile</b> specifies the name of the PEM file that contains the RSA key to be converted.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| <b>embedsavefile</b>    | When using the <b>embed</b> application type, the <b>VALUE</b> for <b>embedsavefile</b> specifies the name for the file where the fake RSA private key is to be saved. The file has the same syntax as an RSA private key file, but actually contains the key identifier rather than the key itself, which remains protected.<br>A certificate request and a self-signed certificate are also written. If the filename is <b>ROOT.EXT</b> then the request is saved to <b>ROOT_req.EXT</b> and the self-signed certificate is saved to <b>ROOT_selfcert.EXT</b> .                                                                                                                                                                                                                                                                                                                                                        |
| <b>from-application</b> | When retargeting a key, the <b>VALUE</b> for <b>from-application</b> specifies the application name of the key to be retargeted. Only applications for which at least one key exists are acceptable.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |

| Option                | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|-----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>from-ident</b>     | When retargeting a key, the <b>VALUE</b> for <b>from-ident</b> specifies the identifier of the key to be retargeted (as displayed by the <b>nfkminfo</b> command-line utility).                                                                                                                                                                                                                                                                                               |
| <b>hexdata</b>        | The <b>VALUE</b> for <b>hexdata</b> specifies the hex value of DES or Triple DES key to import. The hex digits are echoed to the screen and can appear in process listings if this parameter is specified on the command line.                                                                                                                                                                                                                                                |
| <b>ident</b>          | The <b>VALUE</b> for <b>ident</b> specifies a unique identifier for the key in the Security World. For applications of types <b>simple</b> or <b>hwcrhk</b> , this is the key identifier to use (the exact identifier for <b>simple</b> , for <b>hwcrhk</b> the key type is implicitly included). For other application types, keys are assigned an automatically generated identifier and accessed by means of some application-specific name.                               |
| <b>keystore</b>       | The <b>VALUE</b> for <b>keystore</b> specifies the file name of the key store to use. This must be an nShield key store.                                                                                                                                                                                                                                                                                                                                                      |
| <b>keystorepass</b>   | The <b>VALUE</b> for <b>keystorepass</b> specifies the password to the key store to use.                                                                                                                                                                                                                                                                                                                                                                                      |
| <b>module</b>         | <p>The <b>VALUE</b> for <b>module</b> specifies a module to use when generating the key. If there is more than one usable module, you are prompted to supply a value for one of them. The default is the first usable module (one in the current Security World and in the operational state).</p> <p>Note You can also specify a module by setting the <b>--module</b> option.</p>                                                                                           |
| <b>paramsreadfile</b> | The <b>VALUE</b> for <b>paramsreadfile</b> specifies the name of the group parameters file that contains the discrete log group parameters for Diffie-Hellman keys only. This should be a PEM-formatted PKCS#3 file. If a <b>VALUE</b> for <b>paramsreadfile</b> is not specified, the module uses a default file.                                                                                                                                                            |
| <b>pemreadfile</b>    | The <b>VALUE</b> for <b>pemreadfile</b> specifies the name of the PEM file that contains the key to be imported. When importing an RSA key, this is the name of the PEM-encoded PKCS #1 file to read it from. Password-protected PEM files are not supported.                                                                                                                                                                                                                 |
| <b>plainname</b>      | The <b>VALUE</b> for <b>plainname</b> specifies the key name within the Security World. For some applications, the key identifier is derived from the name, but for others the name is just recorded in <b>kmdata</b> and not used otherwise.                                                                                                                                                                                                                                 |
| <b>protect</b>        | The <b>VALUE</b> for <b>protect</b> specifies the protection method, which can be <b>module</b> for security-world protection, <b>softcard</b> for softcard protection or <b>token</b> for Operator Card Set protection. The default is <b>token</b> , except for <b>seeconf</b> keys, where the default is <b>module</b> . <b>seeinteg</b> keys are always token-protected. The <b>softcard</b> option is only available when your system has at least one softcard present. |
| <b>pubexp</b>         | For RSA key generation only, the <b>VALUE</b> for <b>pubexp</b> specifies (in hexadecimal format) the public exponent to use when generating RSA keys. We recommend leaving this parameter blank unless advised to supply a particular value by Support.                                                                                                                                                                                                                      |

| Option               | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>recovery</b>      | The <b>VALUE</b> for <b>recovery</b> enables recovery for this key and is only available for card-set protected keys in a recovery-enabled world. If set to <b>yes</b> , the key is recoverable. If set to <b>no</b> , key is not recoverable. The default is <b>yes</b> . Non-recoverable module-protected keys are not supported.                                                                                                                                                                                                                                                                                |
| <b>seeintegname</b>  | If present, the <b>VALUE</b> for <b>seeintegname</b> identifies a <b>seeinteg</b> key. The ACL of the newly generated private key is modified to require a certificate from the <b>seeinteg</b> key for its main operational permissions, such <b>Decrypt</b> and <b>Sign</b> ( <b>DuplicateHandle</b> , <b>ReduceACL</b> , and <b>GetACL</b> are still permitted without certification.)                                                                                                                                                                                                                          |
| <b>selfcert</b>      | The <b>VALUE</b> for <b>selfcert</b> enables you to generate a self-signed certificate when generating a PKCS #11 key (RSA keys only). To generate a self-signed certificate request you must set <b>selfcert</b> to <b>yes</b> , which makes <b>generatekey</b> prompt you to fill in the extra fields required to generate a key with a self-signed certificate. The resultant certificate is saved to the current working directory with a file name of the form <b>FILENAME.ext</b> . You can use this parameter with the <b>--retarget</b> option to generated a self-signed certificate for an existing key. |
| <b>size</b>          | For key types with variable-sized keys, the <b>VALUE</b> for <b>size</b> specifies the key size in bits. The range of allowable sizes depends on the key type and whether the <b>--no-verify</b> option is used. The default depends on the key type; for information on available key types and sizes, see Appendix G: <a href="#">Cryptographic algorithms</a> . This parameter does not exist for fixed-size keys, nor for ECDH and ECDSA keys which are specified using <b>curve</b> .                                                                                                                         |
| <b>strict</b>        | For DSA key generation only, setting the <b>VALUE</b> for <b>strict</b> to <b>yes</b> enables strict verification, which also limits the size to exactly 1024 bits. The default is <b>no</b> .                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <b>type</b>          | The <b>VALUE</b> for <b>type</b> specifies the type of key. You must usually specify the key type for generation and import (though some applications only support one key type, in which case you are not asked to choose). Sometimes the type must also be specified for retargeting; for information on available key types and sizes, see Appendix G: <a href="#">Cryptographic algorithms</a> . The <b>--verify</b> option limits the available key types.                                                                                                                                                    |
| <b>x509country</b>   | The <b>VALUE</b> for <b>x509country</b> specifies a country code, which must be a valid 2-letter code, for the certificate request.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| <b>x509dnscommon</b> | The <b>VALUE</b> for <b>x509dnscommon</b> specifies a site domain name, which can be any valid domain name, for the certificate request.                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| <b>x509email</b>     | The <b>VALUE</b> for <b>x509email</b> specifies an email address for the certificate request.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| <b>x509locality</b>  | The <b>VALUE</b> for <b>x509locality</b> specifies a city or locality for the certificate request.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| <b>x509org</b>       | The <b>VALUE</b> for <b>x509org</b> specifies an organization for the certificate request.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| <b>x509orgunit</b>   | The <b>VALUE</b> for <b>x509orgunit</b> specifies an organizational unit for the certificate request.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |

| Option              | Description                                                                                                                                                                                                  |
|---------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>x509province</b> | The <b>VALUE</b> for <b>x509province</b> specifies a province for the certificate request.                                                                                                                   |
| <b>xsize</b>        | The <b>VALUE</b> for <b>xsize</b> specifies the private key size in bits when generating Diffie-Hellman keys. The defaults are 256 bits for a key size of 1500 bits or more or 160 bits for other key sizes. |

## Available key properties by action/application

The following table shows which actions (generate, import, and retarget) and applications are applicable to the different **NAME** options:

| Property                | Action   |        |          | Application |       |        |       |         |          |        |        | kpm |
|-------------------------|----------|--------|----------|-------------|-------|--------|-------|---------|----------|--------|--------|-----|
|                         | generate | import | retarget | custom      | embed | hwcrhk | pks11 | seeconf | seeinteg | seessl | simple |     |
| <b>alias</b>            | X        | X      | X        |             |       |        |       |         |          |        |        |     |
| <b>blobsavefile</b>     | X        | X      | X        | X           |       |        |       |         |          |        |        |     |
| <b>cardset</b>          | X        | X      |          | X           | X     | X      | X     |         |          |        | X      | X   |
| <b>certreq</b>          |          |        |          |             |       |        | X     |         |          |        |        |     |
| <b>checks</b>           | X        |        |          | X           | X     | X      | X     |         |          |        | X      | X   |
| <b>curve</b>            | X        |        |          | X           | X     | X      | X     | X       | X        |        | X      |     |
| <b>embedconvfile</b>    |          | X      |          |             | X     |        |       |         |          |        |        |     |
| <b>embedsavefile</b>    | X        | X      | X        |             | X     |        | X     |         |          |        |        |     |
| <b>from-application</b> |          |        | X        | X           | X     | X      | X     |         |          |        | X      | X   |
| <b>from-ident</b>       |          |        | X        | X           | X     | X      | X     |         |          |        | X      | X   |
| <b>hexdata</b>          |          | X      |          | X           | X     | X      | X     |         |          |        | X      |     |
| <b>ident</b>            | X        | X      |          |             |       | X      |       |         |          |        | X      | X   |
| <b>keystore</b>         | X        | X      | X        |             |       |        |       |         |          |        |        |     |
| <b>keystorepass</b>     | X        | X      | X        |             |       |        |       |         |          |        |        |     |
| <b>module</b>           | X        | X      |          | X           | X     | X      | X     |         |          | X      | X      | X   |
| <b>nvrnm</b>            | X        | X      |          | X           | X     | X      | X     |         |          |        | X      |     |
| <b>paramsreadfile</b>   | X        |        |          | X           | X     | X      | X     | X       | X        |        | X      |     |
| <b>pemreadfile</b>      |          | X      |          | X           |       | X      |       |         |          |        | X      | X   |
| <b>plainname</b>        | X        | X      | X        | X           | X     |        | X     | X       | X        | X      | X      | X   |
| <b>protect</b>          | X        | X      |          | X           | X     | X      | X     | X       | X        | X      | X      | X   |
| <b>pubexp</b>           | X        |        |          | X           | X     | X      | X     |         |          |        | X      | X   |
| <b>qsize</b>            | X        |        |          | X           | X     | X      | X     |         |          |        | X      | X   |
| <b>recovery</b>         | X        | X      |          | X           | X     | X      | X     | X       | X        |        | X      | X   |
| <b>seeintegname</b>     |          |        |          | X           |       |        |       |         |          |        | X      |     |
| <b>selfcert</b>         |          |        |          |             |       |        | X     |         |          |        |        |     |

| Property      | Action   |        |          | Application |       |        |        |         |          |        |        | kpm |
|---------------|----------|--------|----------|-------------|-------|--------|--------|---------|----------|--------|--------|-----|
|               | generate | import | retarget | custom      | embed | hwcrhk | pkcs11 | seeconf | seeinteg | seessi | simple |     |
| size          | X        |        |          | X           | X     | X      | X      | X       | X        | X      | X      | X   |
| strict        | X        |        |          | X           | X     | X      | X      |         |          |        | X      |     |
| type          | X        |        |          | X           | X     | X      | X      | X       | X        | X      | X      | X   |
| x509country   | X        | X      | X        |             | X     |        |        |         |          |        |        | X   |
| x509dnscommon | X        | X      | X        |             | X     |        |        |         |          |        |        | X   |
| x509email     | X        | X      | X        |             | X     |        |        |         |          |        |        | X   |
| x509locality  | X        | X      | X        |             | X     |        |        |         |          |        |        | X   |
| x509org       | X        | X      | X        |             | X     |        |        |         |          |        |        | X   |
| x509orgunit   | X        | X      | X        |             | X     |        |        |         |          |        |        | X   |
| x509province  | X        | X      | X        |             | X     |        |        |         |          |        |        | X   |
| xsize         | X        |        |          | X           | X     | X      | X      |         |          |        | X      |     |



## Appendix I: Checking and changing module mode

This appendix explains how to check and change the module mode for different types of module. You must change the module mode in order to perform certain maintenance and configuration tasks.

### Checking and changing nShield Solo (PCI and PCIe) module mode

#### Mode switch

The Mode switch on the back panel controls the mode of nShield PCI and PCIe modules. To prevent accidental operation of the Mode switch, turn on the override switch on the HSM. If this override switch is on, the HSM ignores the position of the Mode switch.

If your nShield PCI or PCIe module does not enter initialization or maintenance mode:

- 1 Check that the override switch is off (see Figure 4).
- 2 Clear the module in either of two ways:
  - Run the command:

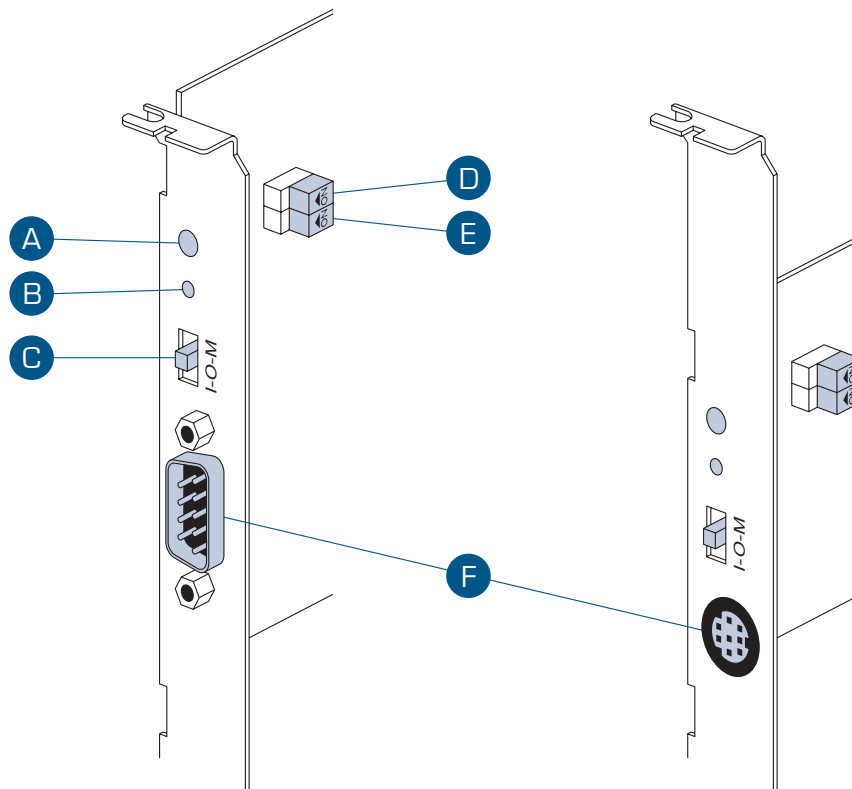
---

```
nopclearfail -clear --all
```

---

- Press the Clear switch.

Figure 4 Back panel and switches (nShield PCI module [left] and nShield PCIe module [right])



| Label | Description                                                                                                             |
|-------|-------------------------------------------------------------------------------------------------------------------------|
| A     | Status LED                                                                                                              |
| B     | Clear switch                                                                                                            |
| C     | Mode switch                                                                                                             |
| D     | Override jumper (shown in “off” position)                                                                               |
| E     | Unused jumper (shown in “off” position)                                                                                 |
| F     | Smart card connector (D-type connector on nShield PCI module [left], mini-DIN connector on nShield PCIe module [right]) |

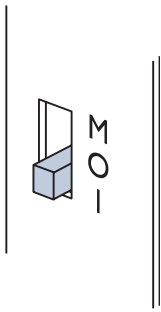
## Changing the module’s mode

### Putting a module in pre-initialization mode

To put a module into pre-initialization mode:

- 1 Switch the Mode switch on the module’s back panel to the initialization (I) position, as shown below:

Figure 5 Mode switch set to initialization



- 2 Reset the module by running the command **nopclearfail --clear --all** or by pressing the Clear switch.

The module performs self-tests, during which the Status LED is lit continuously.

When the self-tests are complete, the unit normally enters pre-initialization mode. In this mode, the Status LED flashes a series of single short pulses.

If the Status LED does not flash a series of single short pulses, refer to the following table:

| LED                                                                                                                                                                                         | State           | Reason and remedy                                                                                                                                                      |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Mainly on but regularly blinks off<br>(The exact timing depends on the nShield module. The longer the LED stays on the less the load. At 100% load the LED is off for as long as it is on.) | Operational     | The override switch is on. Refer to the installation instructions in the <i>Quick Start Guide</i> for information on accessing and setting the override switch to off. |
| Emits repeated single long flashes                                                                                                                                                          | Pre-maintenance | The Mode switch is in the maintenance position. Repeat steps 1 to 2, positioning the Mode switch in the initialization position.                                       |
| Flashes the Morse SOS pattern followed by a code                                                                                                                                            | Error           | The module has encountered an unrecoverable error. There is a list of these errors and the actions to take in the <i>Quick Start Guide</i> .                           |

**Note** If the Status LED remains continuously on for more than a minute, this indicates that the self tests have failed terminally. Contact Support.

You can use the **enquiry** command-line utility to check that the module is in the pre-initialization mode.

After the module has been put into pre-initialization mode, it is ready to be initialized. It enters *initialization mode* when it receives an **initialization** command (for example, when you run the **new-world** command-line utility).



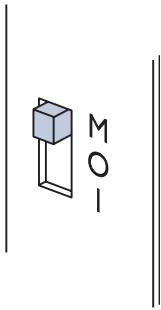
## Putting a module in pre-maintenance mode

This section describes how to put nShield modules into *pre-maintenance mode*. Only put a module into its pre-maintenance mode if you need to upgrade module firmware.

To put a module into pre-maintenance mode, follow these steps:

- 1 Set the Mode switch on the module's back panel to the maintenance (M) position, as shown below:

Figure 6 Mode switch set to maintenance



- 2 Reset the module by running the command **nopclearfail --clear --all** or by pressing the Clear switch.

The module performs self-tests, during which the Status LED is lit continuously.

When the self-tests are complete, the unit normally enters pre-maintenance mode. In this mode, the Status LED flashes a series of long pulses.

If the Status LED does not flash a series of long pulses, refer to the following table:

| LED                                                                                                                                                                                         | State              | Reason and remedy                                                                                                                                                         |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Mainly on but regularly blinks off<br>(The exact timing depends on the nShield module. The longer the LED stays on the less the load. At 100% load the LED is off for as long as it is on.) | Operational        | The override switch is on. Refer to the installation instructions in the <i>Quick Start Guide</i> for information on accessing the override switch and setting it to off. |
| Emits repeated single short flashes                                                                                                                                                         | Pre-initialization | The Mode switch is in the initialization position. Repeat steps 1 to 2, positioning the mode switch in the maintenance position.                                          |
| Flashes the Morse SOS pattern followed by a code                                                                                                                                            | Error              | The module has encountered an unrecoverable error. There is a list of these errors and the actions to take in the <i>Quick Start Guide</i> .                              |

**Note** If the Status LED remains continuously on for more than a minute, this indicates that the self tests have failed terminally. Contact Support.

You can use the **enquiry** command-line utility to check that the module is in the pre-maintenance mode.

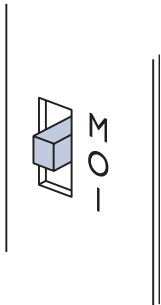
After the module has been put into pre-maintenance mode, it is ready for maintenance. It enters *maintenance mode* when it receives a **Maintenance** command (for example, when you run the **loadrom** command-line utility).

## Putting a module in operational mode

To put a module into operational mode:

- 1 Set the Mode switch on the module's back panel to the operational (O) position, as shown below:

Figure 7 Mode switch set to operational



- 2 Reset the module by running the command **nopclearfail --clear --all** or by pressing the Clear switch.

The module performs self-tests, during which the Status LED is lit continuously. When the self-tests are complete, the unit normally enters operational mode and ready to accept commands.

In operational mode, the Status LED is mainly on, but blinks off briefly at regular intervals. The frequency with which the Status LED blinks off is an indication of the load on the module. The more frequently the Status LED blinks off, the greater the load.

If, when you first return the module to operational mode, the Status LED is not mainly on while blinking off regularly, refer to the following table:

### Status indications

| <i><b>LED</b></i>                                | <i><b>Mode</b></i> | <i><b>Reason and remedy</b></i>                                                                                                                                                                                          |
|--------------------------------------------------|--------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Emits repeated single short flashes              | Pre-initialization | You have not removed the initialization link or have left the initialization switch on. Remove the link, or turn off the switch, and restart the module.                                                                 |
| Emits repeated single long flashes               | Pre-maintenance    | The maintenance link has been fitted instead of the initialization link or the maintenance switch was on instead of the initialization switch. Open up the case and fit the correct link, or turn the correct switch on. |
| Flashes the Morse SOS pattern followed by a code | Error              | The module has encountered an unrecoverable error. There is a list of these errors and the actions to take in the <i>Quick Start Guide</i> .                                                                             |



## Appendix J: Upgrading firmware

This appendix describes how to load updated firmware onto your nShield hardware security device. nShield firmware is supplied on your Security World Software DVD-ROM.

### Version Security Number (VSN)

All nShield firmware includes a *Version Security Number* (VSN). This number is increased whenever we improve the security of the firmware or add significant new functionality.

We supply several versions of the module firmware. You can always upgrade to firmware with an equal or higher VSN than that currently installed on your module.



You can never load firmware with a lower VSN than the currently installed firmware.

Ensuring you use firmware with the highest available VSN allows you to benefit from security improvements and enhanced functionality. It also prevents future downgrades of the firmware that could potentially weaken security. However, you may choose to install firmware that does not have the highest available VSN. For example, if you have a regulatory requirement to use FIPS-approved firmware, you should install the latest available FIPS-validated firmware, which may not have the highest VSN. Similarly, if you want to install a version with enhanced features without committing yourself to the upgrade, you can do so providing you upgrade only to firmware with a VSN equal to that currently installed on your module.

In general, we recommend using the latest firmware with the highest available VSN.

### Firmware on the DVD-ROM

Your Security World Software DVD-ROM contains several sets of firmware for each supplied product. These can include the latest available:

- FIPS-approved firmware with the base VSN
- FIPS-approved firmware with a higher VSN
- firmware awaiting FIPS approval with the base VSN

- firmware awaiting FIPS approval with a higher VSN.

You should ensure you are using the latest firmware, unless you have a regulatory requirement to use firmware that has been FIPS validated. In the latter case, you should ensure that you are using the latest available FIPS validated firmware.

## Recognising firmware files

The firmware and monitor files are stored in subdirectories within the **firmware** directory on the DVD-ROM. The subdirectories are named by version number.

Firmware and monitor files for hardware modules have a **.nff** filename suffix. Monitor filenames have an **ldb** prefix. (Files that have a **.ftv** suffix are used for checking similarly named firmware files. They are not firmware files.)

Files for use with nShield Solo modules have **ncx3p** in the filename. Files for use with nShield Edge modules have **ncx1z** in the filename.

The VSN of a firmware file is incorporated into its filename and is denoted by a dash followed by the digits of the VSN. For example, **-24** means the VSN is 24.

**Note** See the *Release Notes* for details of current firmware versions and types supplied with this release to help you select the correct files to use.

To display information about a firmware file on the DVD-ROM, enter the following command:

---

```
loadrom --view /disc-name/firmware/firmware_ver/firmware_file.nff
```

---

In this command, *disc-name* is the directory on which you mounted the DVD-ROM, *firmware\_ver* is the firmware version number, and *firmware\_file* is the file name.

## Using new firmware

To use the new firmware, you must:

- 1 Install the latest server software as described in Chapter 3: [Support software installation](#).
- 2 Install the latest firmware, as described below.



This chapter describes how to upgrade module firmware for nShield Solo modules only. If you have another type of module (for example, an nShield Connect module), refer to the corresponding chapter in the appropriate *User Guide*.

## Firmware installation overview

The process of installing or updating firmware on an nShield module depends on whether you need to upgrade the module's monitor.

Each module has a monitor, which allows you to load firmware onto the module. For a module to be able to use firmware 2.50.16 (VSN 25) or later, it must use the supplied monitor. Refer to the *Release Notes* for monitor firmware versions.

To check the version number of the monitor on the module:

- 1 Log in to the host as root or as a user in the group **nfast**.
- 2 Put the module in Maintenance mode and reset the module.

For information on doing this, see [Checking and changing module mode](#) on page 302.

- 3 Run the **enquiry** command-line utility and check that the module is in the pre-maintenance state.

The **Version** number shown is for the monitor.

If you need to upgrade both the monitor and firmware, you must use the **nfloadmon** utility; see [Upgrading both the monitor and firmware](#) on page 310.

If you need to upgrade the firmware only, you must use the **loadrom** utility; see [Upgrading firmware only](#) on page 312.



If you are upgrading a module which has SEE program data or NVRAM-stored keys in its nonvolatile memory, use the **nvr-am-backup** utility to backup your data first.

## Upgrading both the monitor and firmware



You must only use this procedure if you need to upgrade the monitor and firmware on an nShield module. If you only need to upgrade the firmware, see [Upgrading firmware only](#) on page 312.

Follow this procedure carefully. Do not interrupt power to the module during this upgrade process.

To upgrade the monitor and firmware on a module:

- 1 Log in to the host as root or as a user in the group **nfast**.

## 2 Run the command:

---

```
nffloadmon -m<module_number> /disc-name/firmware/monitor_ver/monitor_file.nff /disc-
name/firmware/firmware_ver/firmware_file.nff
```

---

In this command, *<module\_number>* is the module number (such as **-m2** for module 2), *disc-name* is the directory on which you mounted the DVD-ROM, *monitor\_ver* is the monitor version number, *monitor\_file* is the monitor file name, *firmware\_ver* is the firmware version number, and *firmware\_file* is the firmware file name.

For example:

---

```
nffloadmon -m2 /mnt/cdromname/firmware/2-50-16/ldb_ncx3p-24.nff /mnt/cdromname/firmware/2-50-16/ncx3p-25.nff
```

---

The firmware files are signed and encrypted; you can load only the correct version for your module.

- 3 Confirm the version of the monitor and firmware.
- 4 Put the module into the different modes when prompted to do so.

When you change the mode, do not use the **nopclearfail** command or Clear switch. The **nffloadmon** utility resets the module after each mode change.

For information on changing the mode, see [Changing the module's mode](#) on page 303.

**Note** If you are upgrading an nShield Solo module and it is not changing state, it means that the override switch is on. Refer to the installation instructions in the *Quick Start Guide* for information on accessing the override switch and switching it off.

- 5 When the **nffloadmon** utility has completed, put the module into Initialization mode (as prompted), and then initialize the module by running the command:

---

```
initunit
```

---

- 6 Put the module in Maintenance mode and reset the module.
- 7 Run the **enquiry** command to verify the module is in maintenance state and has the correct monitor version.

In Maintenance mode, the **enquiry** command shows the version number of the monitor.

- 8 Put the module in Operational mode and reset the module.

- 9 Run the **enquiry** command to verify the module is in operational state and has the correct firmware version.

In Operational mode, the **enquiry** command shows the version number of the firmware.

- 10 Log in to the host as normal.

## Upgrading firmware only

To upgrade the firmware on a module:

- 1 Log in to the host as root or as a user in the group **nfast**.
- 2 Put the module in Maintenance mode and reset the module.

For information on changing the mode, see [Changing the module's mode](#) on page 303.

- 3 Run the **enquiry** command-line utility to check that the module is in the pre-maintenance state.

**Note** If the nShield Solo module is still in the operational state, it means that the override switch is on. Refer to the installation instructions in the *Quick Start Guide* for information on accessing the override switch and switching it off.

- 4 Insert the Security World Software DVD-ROM and mount it on your file system.
- 5 Load the new firmware by running the command:

---

```
loadrom -m<module_number> /disc-name/firmware/firmware_ver/firmware_file.nff
```

---

In this command, **<module\_number>** is the module number (such as **-m2** for module 2), **disc-name** is the directory on which you mounted the DVD-ROM, **firmware\_ver** is the firmware version number, and **firmware\_file** is the firmware file name.

For example:

---

```
loadrom -m2 /mnt/cdromname/firmware/2-50-16/ncx3p-25.nff
```

---

The firmware files are signed and encrypted; you can load only the correct version for your module.

- 6 Put the module in Initialization mode and reset the module.



- 7 Initialize the module by running the command:

---

```
initunit
```

---

- 8 Put the module in Operational mode and reset the module.
- 9 Run the **enquiry** command to verify the module is in operational state and has the correct firmware version.

In Operational mode, the **enquiry** command shows the version number of the firmware.

- 10 Log in to the host as normal.

## After firmware installation

After you have installed new firmware and initialized the module, you can create a new Security World with the module or reinitialize the module into an existing Security World.

If you are initializing the module into a new Security World, see [Creating a Security World](#) on page 71.

If you are re-initializing the module into an existing Security World, see [Adding or restoring a module to the Security World](#) on page 94.



## Appendix K: nCipher SNMP monitoring agent

This appendix describes the nCipher SNMP monitoring agent. The SNMP monitoring agent provides you with components that you can add to your (third-party) SNMP manager application.

The Simple Network Management Protocol (SNMP) was developed in 1988 and revised in 1996. It is currently regarded as the standard method of network management. It is widely supported and offers greater interoperability than traditional network management tools (for example, **rsh** or **netstat**). This makes it ideal for use for the large array of platforms that we support and also avoids the overhead of remote login and execution, helping to reduce network congestion and improve performance.

SNMP defines a collection of network management functions allowing management stations to gather information from, and transmit commands to, remote machines on the network. Agents running on the remote machines can take information gathered from the system and relay this information to the manager application. Such information is either requested from the underlying operating system or gained by interrogating the hardware.

**Note** Every SNMP manager adds monitor components differently. Consult the documentation supplied with your SNMP Manager application for details on how to add the nCipher MIB files.

SNMP defines the following SNMP messages:

| Message     | Description                                                                            |
|-------------|----------------------------------------------------------------------------------------|
| <b>get</b>  | This message is sent by a manager to retrieve the value of an object at the agent.     |
| <b>set</b>  | This message is sent by a manager to set the value of an object at the agent.          |
| <b>trap</b> | This message is sent by an agent to notify a management station of significant events. |

In the development of the nCipher SNMP agent, we have used open-source tools that are part of the NET-SNMP project (formerly UCD-SNMP). More information on SNMP in general, and the data structures used to support SNMP installations, is available from the NET-SNMP project Web site: <http://net-snmp.sourceforge.net/>.

This site includes some support information and offers access to discussion e-mail lists. You can use the discussion lists to monitor subjects that might affect the operation or security of the nCipher SNMP agent or command-line utilities.



Discuss any enquiries arising from information on the NET-SNMP Web site with Support before posting potentially sensitive information to the NET-SNMP Web site.

## Installing and activating the nCipher SNMP agent

The nCipher SNMP Agent enables other computers on the network to connect to it and make requests for information. The nCipher agent is based on the NET-SNMP kit, which has been tested but not fully reviewed by Thales. We strongly recommend that the nCipher agent is deployed only on a private network, or protected from the global Internet by an appropriate firewall.

The nCipher SNMP agent is installed and activated separately. After installing the nCipher SNMP components, an activation command must be issued. This two-stage process is to avoid the inadvertent activation of SNMP capabilities that could supply management information from servers and nShield modules that are not explicitly protected, or which could potentially expose the host computer to attacks on the nCipher SNMP agent itself.

### Default installation settings

When installing Security World Software, you are prompted to select Security World Software components from a list. If you select **all** components, then the nCipher SNMP agent is installed as part of a full Security World Software installation. The default installation directory for the nCipher Management Information Base (MIB) and the SNMP configuration files (**snmp.conf** and **snmpd.conf**) is `/opt/nfast/etc/snmp/`.

### Do you already have an SNMP agent running?

If no existing SNMP agent is found, the nCipher SNMP agent runs on the default port 161. If an existing SNMP agent is detected, and no nCipher SNMP agent configuration files are found (implying a fresh installation), the installer automatically configures the nCipher SNMP agent to use the first unused port above 161 by creating a new **snmp.conf** configuration file with the appropriate directive. It then displays a message indicating the number of the port that it has selected.

If an existing SNMP agent is found and an existing nCipher SNMP agent installation exists, the installer checks the existing configuration files for an appropriate directive and warns you if one does not exist. If you need to edit these configuration files yourself, a port is assigned by editing

the **agentaddress** entry in **snmpd.conf** or editing the **defaultPort** entry in **snmp.conf**. If both files have been edited, the **agentaddress** entry in **snmpd.conf** takes priority for snmpd, and the **defaultPort** entry in **snmp.conf** is ignored.

## Activating the nCipher SNMP agent

The agent is installed with the installation of the Security World Software (see Chapter 3: [Support software installation](#)) and starts automatically.

When installing the Security World Software on Linux platforms, if you have chosen to install the nCipher SNMP agent, the following message is displayed during installation:

---

```
I need to create the ncsnmpd user, in group ncsnmpd, with home directory
/opt/nfast/etc/snmp.
I can either:
1) Try to do it by editing /etc/passwd and /etc/group myself;
2) Try to use \Qadduser ncsnmpd' (this may sort of work on some systems);
3) Try to use \Qadduser --group --system $us' a la Debian 1.3 and later;
4) Try to use \Quseradd -r ncsnmpd' a la at least Red Hat 5.0 and later;
5) Let you do it;
6) Abort the installation process.
Please type a number from 1 to 5:
```

---

Choose the appropriate option for your installation, and then press **Enter**.

Uninstalling the Security World Software (see [Uninstall existing Software](#) on page 35) also uninstalls the agent.

To stop, start, or restart (that is, stop and then immediately start again) the SNMP demon **ncsnmpd**, run a command of the form:

---

```
/etc/init.d/nc_ncsnmpd stop|start|restart
```

---

In a command of this form, specify:

- **stop** to stop the SNMP demon
- **start** to start the SNMP demon
- **restart** to restart the SNMP demon

## Further information

### Protecting the nCipher SNMP installation

The nCipher SNMP Agent allows other computers on the network to connect to it and make requests for information. The nCipher agent is based on the NET-SNMP code base, which has been tested but not fully reviewed by Thales. We strongly recommend that you deploy the nCipher SNMP agent only on a private network or a network protected from the global Internet by an appropriate firewall.

The default nCipher SNMP installation allows read-only access to the Management Information Base (MIB) with any community string. There is no default write access to any part of the MIB.

Every effort has been taken to ensure the confidentiality of cryptographic keys even when the SNMP agent is enabled. In particular, the nShield module is designed to prevent the theft of keys even if the security of the host system is compromised, provided that the Administrator Cards are used only with trusted hosts. Care must be used when changing the configuration of the nCipher SNMP agent.



We strongly advise that you set up suitable access controls in **snmpd.conf**, or a firewall, to protect the agent and the information it can return.

Care has also been taken to ensure that malicious attackers are unable to flood your module with requests by flooding your SNMP agent. Command results from administration or statistics commands are cached, and thus the maximum rate at which the agent sends commands to the module is throttled. For more information on setting the cache time-outs, see [The SNMP agent configuration file: snmpd.conf](#) on page 318.

### Configuring the nCipher SNMP agent

The Security World Software package uses various configuration files to configure its applications. This section describes the overall nature of the configuration files for the SNMP agent.

If you are installing the nCipher SNMP agent to a host that has an existing SNMP agent installation that is not an nCipher SNMP agent, you may need to edit the SNMP configuration files (**snmpd.conf** and **snmp.conf**) associated with the nCipher SNMP agent to change the port on which the agent listens for SNMP requests. For more information, see [Do you already have an SNMP agent running?](#) on page 315.

**Note** Make sure you protect the configuration files if you are storing sensitive information in them.

By default, the nCipher SNMP configuration file (**snmp.conf**) is located in the **/opt/nfast/etc/snmp/** directory. In this directory, the agent looks for files with the **.conf** extension.

Note You can override the default search path by setting the environment variable **SNMPCONFPATH** to a colon-separated [":"] list of directories for which to search.

## Re-reading snmpd.conf and snmpd.local.conf

The nCipher SNMP agent can be forced to re-read its configuration files with:

- an snmp set of integer(1) to **enterprises.nCipher.reloadConfig.0(.1.3.6.1.4.1.7682.999.0)**
- a kill **-HUP** signal sent to the **snmpd** agent process.

## The SNMP configuration file: snmp.conf

The **snmp.conf** configuration file contains directives that apply to all SNMP applications. These directives can be configured to apply to specific applications. The **snmp.conf** configuration file is not required for the agent to operate and report MIB entries.

## The SNMP agent configuration file: snmpd.conf

The **snmpd.conf** configuration file defines how the nCipher SNMP agent operates. It is required only if an agent is running. This file can contain any of the directives described in this section.

The **snmpd.conf** file can contain any of the directives available for use in the **snmp.conf** file.

The **snmpd.conf** file can also contain the following Security World Software-specific directives:

| Directive           | Description                                                                                                                                                                                 |
|---------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>stattimeout</b>  | This directive specifies the maximum length of time for which statistics commands are cached. The default is 60 seconds.                                                                    |
| <b>admintimeout</b> | This directive specifies the maximum length of time for which administrative commands are cached. The default is 5 seconds.                                                                 |
| <b>keytable</b>     | This directive sets the initial state of the key table to <b>none</b> , <b>all</b> , or <b>query</b> . See <b>listKeys</b> in <a href="#">Administration sub-tree overview</a> on page 321. |

## Using the nCipher SNMP agent with a manager application

**Note** The nCipher SNMP monitoring agent provides you with components that can be added to your (third-party) SNMP manager application. Every SNMP manager adds monitor components differently. Consult the documentation supplied with your SNMP Manager application for details on how to add the nCipher MIB files.

### Manager configuration

The manager application is the interface through which the user is able to perform network management functions. A manager communicates with agents using SNMP primitives (**get**, **set**, **trap**) and is unaware of how data is retrieved from, and sent to, managed devices. This form of encapsulation raises two main issues:

- the manager is hidden from all platform specific details
- the manager can communicate with agents running on any IP-addressable machine.

As a consequence, manager applications are generic and can be bought “off the shelf”. You may already be running SNMP managers, and if so, you can use them to query the nCipher agent.

**Note** The manager is initially unaware of the MIB tree structure at a particular node. Managed objects can be retrieved or modified, but only if their location in the tree is known.

It is more useful if the manager can *see* the MIB tree present at each managed node. The MIB module descriptions for a particular node must be parsed by a manager-specific MIB compiler and converted to configuration files. These files are read by the manager application at run time.

The nCipher SNMP agent is designed to monitor all current nShield modules. The SNMP agent can monitor e-commerce accelerator and key management modules, working with all supported versions of nShield firmware (contact Support for details of supported firmware).

### nCipher MIB module overview

A large proportion of the nCipher SNMP system is fully specified by the structure of the MIB; the behavior of the agent depends on relaying information according to the layout of the MIB.

The nCipher MIB module resides at a registered location in the MIB tree determined by the Internet Assigned Numbers Authority (IANA). The private enterprise number of 7682 designated by the IANA corresponds to the root of the nCipher branch, and by convention this (internal) node is the company name.

The MIB module groups logically related data together, organizing itself into a classification tree, with managed objects present at leaf nodes. The nC-series node (**enterprises.nCipher.nC-series**) is placed as a sub-tree of the nCipher root (**enterprises.nCipher**); this allows future product lines to be added as additional sub-trees. The structure of the tree underneath the registered location is vendor-defined, and this specification defines the structure chosen to represent Security World Software-specific data.

The MIB file is shipped as `/opt/nfast/etc/snmp/mibs/ncipher-mib.txt`.

## MIB functionality

The nCipher MIB module separates module information into the following categories:

- retrieval of status and information about installed nC-series modules
- retrieval of live statistics of performance of installed nC-series modules

These categories form the top-level nodes of the nCipher sub-tree; the functionality of the first category is in the administration sub-tree, and the second category is in the statistics sub-tree. The top-level tree also contains 3 items that it would be useful to check at-a-glance:

| Node name        | R/W | Type       | Remarks                                                                                                                                                   |
|------------------|-----|------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------|
| hardserverFailed | R   | TruthValue | <b>True</b> if the remote hardserver is not running. If the hardserver is not running, then most of the rest of the information is unreliable or missing. |
| modulesFailed    | R   | TruthValue | <b>True</b> if any modules have failed.                                                                                                                   |
| load             | R   | Unsigned32 | Percentage of total available capacity currently utilized.                                                                                                |

## Traps

The traps sub-tree (**enterprises.nCipher.nC-series.C-traps**) contains traps that the SNMP agent sends when certain events occur. To enable traps, add a **trap2sink** directive to the configuration file **snmpd.conf**.

**Note** Adding a **trapsink** directive, instead of **trap2sink**, does not enable traps.



The following table gives details of the individual traps:

| Node name         | Description                                                      |
|-------------------|------------------------------------------------------------------|
| hardserverAlert   | This trap is sent when the hardserver fails or is shut down.     |
| hardserverUnAlert | This trap is sent when the hardserver restarts.                  |
| moduleAlert       | This trap is sent when a module fails.                           |
| moduleUnAlert     | This trap is sent when a module is restarted after a failure.    |
| psuAlert          | This trap is sent when a PSU fails.                              |
| psuUnAlert        | This trap is sent when a previously-failed PSU is working again. |
| fanfailureAlert   | This trap is sent when a fan fails.                              |
| fanfailureUnAlert | This trap is sent when a previously-failed fan is working again. |

Note Some traps can take up to five minutes to be received.

## Administration sub-tree overview

The administration sub-tree (**enterprises.nCipher.nC-series.administration**) contains information about the permanent state of the hardserver and the connected modules. It is likely that most of the information in this branch rarely changes over time, unlike the **statistics** branch. The information given in the administration sub-tree is mostly acquired by the **NewEnquiry** command and is supplied both per-module and (where appropriate) aggregated over all modules.

The following table gives details of the individual nodes in the administration sub-tree:

| Node name                 | R/W | Type                                | Remarks                                                                                             |
|---------------------------|-----|-------------------------------------|-----------------------------------------------------------------------------------------------------|
| SecurityWorld             | R   | TruthValue                          | <b>True</b> if a Security World is installed and operational.                                       |
| hardserverRunning         | R   | Enum<br>1: Running<br>2: NotRunning | This variable reflects the current state of the hardserver ( <b>Running</b> or <b>NotRunning</b> ). |
| noOfModules               | R   | Unsigned32                          | Number of nC-series modules.                                                                        |
| hsVersion                 | R   | DisplayString                       | Hardserver version string.                                                                          |
| [global]speedIndex        | R   | Unsigned32                          | Number of 1024-bit signatures each second.                                                          |
| [global]minQ [global]maxQ | R   | Unsigned32                          | Minimum and maximum recommended queues.                                                             |
| swState                   | R   | DisplayString                       | Security World display flags, as reported by <b>nfkminfo</b> .                                      |

| Node name        | R/W | Type                                                                   | Remarks                                                                                                                                                                                 |
|------------------|-----|------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| listKeys         | R/W | Enum<br><br>1: none<br><br>2: all<br><br>3: query<br><br>4: resetquery | Controls the behavior of the key table (switch off, display all keys, enable individual attribute queries, clear the query fields). Displaying all keys can result in a very long list. |
| serverFlags      | R   | DisplayString                                                          | Supported hardserver facilities (the <b>NewEnquiry</b> level 4 flags).                                                                                                                  |
| remoteServerPort | R   | Gauge                                                                  | TCP port on which the hardserver is listening.                                                                                                                                          |
| swGenTime        | R   | DisplayString                                                          | Security World's generation time.                                                                                                                                                       |
| swGeneratingESN  | R   | DisplayString                                                          | ESN of the module that generated the Security World.                                                                                                                                    |

**listKeys** can be preset using the **keytable** config directive in **snmpd.conf** (see [The SNMP agent configuration file: snmpd.conf](#) on page 318).

## Security World hash sub-tree

The following table gives details of the nodes in the Security World hash sub-tree (**enterprises.nCipher.nC-series.administration.swHashes**):

| Node name | R/W | Type  | Remarks                                                   |
|-----------|-----|-------|-----------------------------------------------------------|
| hashKNSO  | R   | MHash | Hash of the Security Officer's key.                       |
| hashKM    | R   | MHash | Hash of the Security World key.                           |
| hashKRA   | R   | MHash | Hash of the recovery authorization key.                   |
| hashKRE   | R   | MHash | Hash of the recovery key pair.                            |
| hashKFIPS | R   | MHash | Hash of the FIPS authorization key.                       |
| hashKMC   | R   | MHash | Hash of the module certification key.                     |
| hashKP    | R   | MHash | Hash of the pass phrase replacement key.                  |
| hashKNV   | R   | MHash | Hash of the nonvolatile memory (NVRAM) authorization key. |
| hashKRTC  | R   | MHash | Hash of the Real Time Clock authorization key.            |
| hashKDSEE | R   | MHash | Hash of the SEE Debugging authorization key.              |
| hashKFTO  | R   | MHash | Hash of the Foreign Token Open authorization key.         |

## Security World quorums sub-tree

The following table gives details of the nodes in the Security World quorums sub-tree (**enterprises.nCipher.nC-series.administration.swQuorums**):

| Node name       | R/W | Type     | Remarks                                                              |
|-----------------|-----|----------|----------------------------------------------------------------------|
| adminQuorumK    | R   | Unsigned | The default quorum of Administrator cards.                           |
| adminQuorumN    | R   | Unsigned | The total number of cards in the ACS.                                |
| adminQuorumM    | R   | Unsigned | The quorum required for module reprogramming.                        |
| adminQuorumR    | R   | Unsigned | The quorum required to transfer keys for OCS replacement.            |
| adminQuorumP    | R   | Unsigned | The quorum required to recover the pass phrase for an Operator card. |
| adminQuorumNV   | R   | Unsigned | The quorum required to access nonvolatile memory (NVRAM).            |
| adminQuorumRTC  | R   | Unsigned | The quorum required to update the Real Time Clock.                   |
| adminQuorumDSEE | R   | Unsigned | The quorum required to view full SEE debug information.              |
| adminQuorumFTO  | R   | Unsigned | The quorum required to use a Foreign Token Open Delegate Key.        |

## Module administration table

The following table gives details of the nodes in the module administration table (**enterprises.nCipher.nC-series.administration.moduleAdminTable**):

| Node name           | R/W | Type                                                                                                                                                    | Remarks                                                                                                                                  |
|---------------------|-----|---------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------|
| moduleAdminIndex    | R   | Integer                                                                                                                                                 | Module number of this row in the table.                                                                                                  |
| mode                | R   | Enum<br><br>1: Operational<br><br>2: Pre-init<br><br>3: Init<br><br>4: Pre-maint<br><br>5: Maint<br><br>6: AccelOnly<br><br>7: Failed<br><br>8: Unknown | Current module state.                                                                                                                    |
| fwVersion           | R   | DisplayString                                                                                                                                           | Firmware version string.                                                                                                                 |
| serialNumber        | R   | DisplayString                                                                                                                                           | Module Electronic Serial Number (ESN).                                                                                                   |
| moduleType          | R   | DisplayString                                                                                                                                           | Module type code.                                                                                                                        |
| productName         | R   | DisplayString                                                                                                                                           |                                                                                                                                          |
| hwPosInfo           | R   | DisplayString                                                                                                                                           | Hardware bus/slot info (such as PCI slot number).                                                                                        |
| moduleSecurityWorld | R   | TruthValue                                                                                                                                              | Indicates whether or not the module is in the current SW.                                                                                |
| smartcardState      | R   | DisplayString                                                                                                                                           | Description of smart card in slot (empty, unknown card, admin/operator card from current SW, failed). N/A for acceleration only modules. |

| Node name        | R/W | Type                                                                                                                                                                                                                                              | Remarks                                                                                                |
|------------------|-----|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------|
| moduleSWState    | R   | Enum<br><br>1: unknown<br><br>2: usable<br><br>3: maintmode<br><br>4: uninitialized<br><br>5: factory<br><br>6: foreign<br><br>7: accelonly<br><br>8: failed<br><br>9: unchecked<br><br>10: initmode<br><br>11: preinitmode<br><br>12: outofrange | Current module and Security World state.                                                               |
| moduleSWFlags    | R   | DisplayString                                                                                                                                                                                                                                     | Security World flags for this module.                                                                  |
| hashKML          | R   | MHash                                                                                                                                                                                                                                             | Hash of the module's secret key.                                                                       |
| moduleFeatures   | R   | DisplayString                                                                                                                                                                                                                                     | Features enabled on this module.                                                                       |
| moduleFlags      | R   | DisplayString                                                                                                                                                                                                                                     | Like <b>serverFlags</b> , but for each module.                                                         |
| versionSerial    | R   | Gauge                                                                                                                                                                                                                                             | Firmware Version Security Number (VSN); see <a href="#">Version Security Number (VSN)</a> on page 308. |
| hashKNETI        | R   | MHash                                                                                                                                                                                                                                             | <b>K<sub>NETI</sub></b> hash, if present.                                                              |
| longQ            | R   | Gauge                                                                                                                                                                                                                                             | Max. rec. long queue.                                                                                  |
| connectionStatus | R   | DisplayString                                                                                                                                                                                                                                     | Connection status [for imported modules].                                                              |
| connectionInfo   | R   | DisplayString                                                                                                                                                                                                                                     | Connection information [for imported modules].                                                         |
| machineTypeSEE   | R   | DisplayString                                                                                                                                                                                                                                     | SEE machine type.                                                                                      |

## Slot administration table

The following table gives details of the nodes in the slot administration table (**enterprises.nCipher.nC-series.administration.slotAdminTable**):

| Node name            | R/W | Type                                                                                                                                                  | Remarks                                                               |
|----------------------|-----|-------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------|
| slotAdminModuleIndex | R   | Unsigned32                                                                                                                                            | Module number of the module containing the slot.                      |
| slotAdminSlotIndex   | R   | Unsigned32                                                                                                                                            | Slot number [1-based, unlike nCore which is 0-based].                 |
| slotType             | R   | Enum<br><br>1: Datakey<br>2: Smart card<br>3: Emulated<br>4: Soft token<br>5: Unconnected<br>6: Out of range<br>7: Unknown                            | Slot type.                                                            |
| slotFlags            | R   | DisplayString                                                                                                                                         | Flags referring to the contents of the slot [from <b>slotinfo</b> ].  |
| slotState            | R   | Enum<br><br>1: Unused<br>2: Empty<br>3: Blank<br>4: Administrator<br>5: Operator<br>6: Unidentified<br>7: Read error<br>8: Partial<br>9: Out of range | <b>Partial</b> refers to cards in a partially-created card set.       |
| slotListFlags        | R   | DisplayString                                                                                                                                         | Flags referring to attributes of the slot [from <b>getslotlist</b> ]. |

| Node name         | R/W | Type          | Remarks                                             |
|-------------------|-----|---------------|-----------------------------------------------------|
| slotShareNumber   | R   | Unsigned32    | Share number of card currently in slot, if present. |
| slotSharesPresent | R   | DisplayString | Names of shares present in card currently in slot.  |

## Card set administration table

The following table gives details of the nodes in the card set administration table (**enterprises.nCipher.nC-series.administration.cardsetAdminTable**):

| Node name      | R/W | Type          | Remarks                                          |
|----------------|-----|---------------|--------------------------------------------------|
| hashKLTU       | R   | MHash         | Hash of the token protected by the card set.     |
| cardsetName    | R   | DisplayString |                                                  |
| cardsetK       | R   | Unsigned32    | Required number of cards in the card set.        |
| cardsetN       | R   | Unsigned32    | Total number of cards in the card set.           |
| cardsetFlags   | R   | DisplayString | Other attributes of the card set.                |
| cardsetNames   | R   | DisplayString | Names of individual cards, if set.               |
| cardsetTimeout | R   | Unsigned32    | Token time-out period, in seconds, or 0 if none. |
| cardsetGenTime | R   | DisplayString | Generation time of card set.                     |

## Key administration table

The key administration table is visible as long as the **listKeys** node in the administration sub-tree is set to a value other than **none**.

The following table gives details of the nodes in the key administration table  
(**enterprises.nCipher.nC-series.administration.keyAdminTable**):

| Node name          | R/W | Type                                                                                   | Remarks                                                                                                                                                  |
|--------------------|-----|----------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------|
| keyAppname         | R   | DisplayString                                                                          | Application name.                                                                                                                                        |
| keyIdent           | R   | DisplayString                                                                          | Name of key, as generated by the application.                                                                                                            |
| keyHash            | R   | MHash                                                                                  | Required and total number of cards in the card set.                                                                                                      |
| keyRecovery        | R   | Enum<br>1: Enabled<br>2: Disabled<br>3: No key<br>4: Unknown<br>5: Invalid<br>6: Unset | The value <b>unset</b> is never returned by the key table. If you set the value <b>unset</b> , the keys are not filtered based on any of the attributes. |
| keyProtection      | R   | Enum<br>1: Module<br>2: Cardset<br>3: No key<br>4: Unknown<br>5: Invalid<br>6: Unset   | The value <b>unset</b> is never returned by the key table. If you set the value <b>unset</b> , the keys are not filtered based on any of the attributes. |
| keyCardsetHash     | R   | MHash                                                                                  | Hash of the card set protecting the key, if applicable.                                                                                                  |
| keyFlags           | R   | DisplayString                                                                          | Certificate and public key flags.                                                                                                                        |
| keyExtraEntities   | R   | Unsigned32                                                                             | Number of extra key attributes.                                                                                                                          |
| keySEInteg         | R   | DisplayString                                                                          | SEE integrity key, if present.                                                                                                                           |
| keyGeneratingESN   | R   | DisplayString                                                                          | ESN of the module that generated the key, if present.                                                                                                    |
| keyTimeLimit       | R   | Gauge                                                                                  | Time limit for the key, if set.                                                                                                                          |
| keyPerAuthUseLimit | R   | Gauge                                                                                  | Per-authentication use limit for the key.                                                                                                                |



## Key query sub-tree

The key query sub-tree is used if the **listKeys** node in the administration sub-tree is set to **query**.

If these values are set, they are taken as required attributes for filtering the list of available keys; if multiple attributes are set, the filters are combined (AND rather than OR).

The following table gives details of the nodes in the key query sub-tree (**enterprises.nCipher.nC-series.administration.keyQuery**):

| Node name               | R/W | Type                                                                                   | Remarks                                                                                                                                                  |
|-------------------------|-----|----------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------|
| keyQueryAppname         | R/W | DisplayString                                                                          | Application name.                                                                                                                                        |
| keyQueryIdent           | R/W | DisplayString                                                                          | Name of key, as generated by the application.                                                                                                            |
| keyQueryHash            | R/W | DisplayString                                                                          | Required and total number of cards in the card set.                                                                                                      |
| keyQueryRecovery        | R/W | Enum<br>1: Enabled<br>2: Disabled<br>3: No key<br>4: Unknown<br>5: Invalid<br>6: Unset | The value <b>unset</b> is never returned by the key table. If you set the value <b>unset</b> , the keys are not filtered based on any of the attributes. |
| keyQueryProtection      | R/W | Enum<br>1: Module<br>2: Cardset<br>3: No key<br>4: Unknown<br>5: Invalid<br>6: Unset   | The value <b>unset</b> is never returned by the key table. If you set the value <b>unset</b> , the keys are not filtered based on any of the attributes. |
| keyQueryCardsetHash     | R/W | DisplayString                                                                          | Hash of the card set protecting the key, if applicable.                                                                                                  |
| keyQueryFlags           | R/W | DisplayString                                                                          | Certificate and public key flags.                                                                                                                        |
| keyQueryExtraEntities   | R/W | Unsigned32                                                                             | Number of extra key attributes.                                                                                                                          |
| keyQuerySEInteg         | R/W | DisplayString                                                                          | SEE integrity key, if present.                                                                                                                           |
| keyQueryGeneratingESN   | R/W | DisplayString                                                                          | ESN of the module that generated the key, if present.                                                                                                    |
| keyQueryTimeLimit       | R/W | Gauge                                                                                  | Time limit for the key, if set (0 for no limit).                                                                                                         |
| keyQueryPerAuthUseLimit | R/W | Gauge                                                                                  | Per-authentication use limit for the key (0 for no limit).                                                                                               |

## Statistics sub-tree overview

The statistics sub-tree (**enterprises.nCipher.nC-series.statistics**) contains rapidly changing information about such topics as the state of the nShield modules, the work they are doing, and the commands being submitted.

**Note** Do not rely on information returned from the agent to change instantaneously on re-reading the value. To avoid loading the nShield module with multiple time-consuming statistics commands, the agent can choose to cache the values over a specified period. You can configure this period in the agent configuration file (see [The SNMP agent configuration file: `snmpd.conf`](#) on page 318).

## Statistics sub-tree and module statistics table

The following table gives details of the nodes in the statistics sub-tree, and the module statistics table (**enterprises.nCipher.nC-series.statistics.moduleStatsTable**):

| Node name        | R/W | Type          | Remarks                                                                                                                                                                                                                             |
|------------------|-----|---------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| moduleStatsIndex | R   | Integer       | Module number of this row (for <b>moduleStatsTable</b> ).                                                                                                                                                                           |
| [hs]uptime       | R   | Counter32     | Uptime of the hardserver or module.                                                                                                                                                                                                 |
| cmdCount[All]    | R   | Counter32     | Returned aggregated for all modules and all commands, and per-module for all commands.                                                                                                                                              |
| cmdBytes[All]    | R   | Counter32     | Returned as for <b>cmdCount</b> . When per-module total, aggregate all the different error states into one counter. When aggregated, returned value is combined module errors added to hardserver marshalling/unmarshalling errors. |
| cmdErrors[All]   | R   | Counter32     |                                                                                                                                                                                                                                     |
| replyCount[All]  | R   | Counter32     |                                                                                                                                                                                                                                     |
| replyBytes[All]  | R   | Counter32     |                                                                                                                                                                                                                                     |
| replyErrors[All] | R   | Counter32     | See notes above for <b>cmdErrors</b> .                                                                                                                                                                                              |
| clientCount      | R   | Counter32     |                                                                                                                                                                                                                                     |
| maxClients       | R   | Counter32     |                                                                                                                                                                                                                                     |
| deviceFails      | R   | Counter32     |                                                                                                                                                                                                                                     |
| deviceRestarts   | R   | Counter32     |                                                                                                                                                                                                                                     |
| load[All]        | R   | Counter32     |                                                                                                                                                                                                                                     |
| outstandingCmds  | R   | Counter32     | Total number of outstanding commands over all modules.                                                                                                                                                                              |
| objectCount      | R   | Counter32     |                                                                                                                                                                                                                                     |
| clock            | R   | DisplayString | Depending on the module settings, this can require <b>K<sub>NSO</sub></b> permissions to read (and therefore depend on the installation parameters of the agent).                                                                   |
| currentTemp      | R   | DisplayString | Character representation of the current temperature value (SNMP does not provide for a floating-point type).                                                                                                                        |
| maxTemp          | R   | DisplayString | Maximum temperature the module has ever reached.                                                                                                                                                                                    |

| Node name        | R/W | Type      | Remarks |
|------------------|-----|-----------|---------|
| nvRAMInUse       | R   | Counter32 |         |
| volatileRAMInUse | R   | Counter32 |         |

## Per connection statistics table

The following table gives details of the nodes in the per connection statistics table (**enterprises.nCipher.nC-series.statistics.connStatsTable**):

| Node name           | R/W | Type      | Remarks                                                           |
|---------------------|-----|-----------|-------------------------------------------------------------------|
| connStatsIndex      | R   | Integer   | Index of this entry.                                              |
| connNumber          | R   | Integer   | Hardserver connection number.                                     |
| connUptime          | R   | Counter32 | Uptime of the connection.                                         |
| connCmdCount        | R   | Counter32 | Number of commands submitted through this connection.             |
| connCmdBytes        | R   | Counter32 | Number of bytes submitted through this connection.                |
| connCmdErrors       | R   | Counter32 | Number of marshalling errors on commands through this connection. |
| connReplyCount      | R   | Counter32 | Number of replies received by this connection.                    |
| connReplyBytes      | R   | Counter32 | Number of bytes received by this connection.                      |
| connReplyErrors     | R   | Counter32 | Number of marshalling errors on replies through this connection.  |
| connDevOutstanding  | R   | Gauge32   | Number of commands outstanding on this connection.                |
| connQOutstanding    | R   | Gauge32   | Number of commands outstanding in the hardserver queue.           |
| connLongOutstanding | R   | Gauge32   | Number of long jobs outstanding for this connection.              |
| connRemoteIPAddress | R   | IpAddress | IP Address of connection client.                                  |
| connProcessID       | R   | Integer   | Process identifier reported by connection client.                 |
| connProcessName     | R   | Gauge     | Process name reported by connection client.                       |

## Fan table

The fan table provides the speeds of each fan on the remote module (HSM). The following table gives details of the nodes in the fan table

(**enterprises.nCipher.softwareVersions.netHSMFanTable**):

| Node name         | R/W | Type    | Remarks         |
|-------------------|-----|---------|-----------------|
| netHSMModuleIndex | R   | Integer | Module number   |
| netHSMFanIndex    | R   | Integer | Fan number      |
| netHSMFanSpeed    | R   | Gauge32 | Fan speed [RPM] |

## Software versions table

The following table gives details of the nodes in the software versions table

(**enterprises.nCipher.softwareVersions.softwareVersionsTable**):

| Node name        | R/W | Type                  | Remarks          |
|------------------|-----|-----------------------|------------------|
| compIndex        | R   | Integer               | Table index.     |
| compName         | R   | DisplayString         | Component name.  |
| compOutput       | R   | Component output name | Component name.  |
| compMajorVersion | R   | Gauge                 |                  |
| compMinorVersion | R   | Gauge                 |                  |
| compPatchVersion | R   | Gauge                 |                  |
| compRepository   | R   | DisplayString         | Repository name. |
| compBuildNumber  | R   | Gauge                 |                  |

# Useful nCipher SNMP agent command-line switches

## SNMP agent (snmpd) switches

The SNMP agent that binds to a port and awaits requests from SNMP management software is **snmpd**. Upon receiving a request, **snmpd** processes the request, collects the requested information and/or performs the requested operation(s) and returns the information to the sender.

The nCipher SNMP agent supports a limited subset of command line switches that can be specified when starting the agent.

## Usage

---

```
snmpd [-h] [-v] [-f] [-a] [-d] [-V] [-P PIDFILE):] [-q] [-D] [-p NUM] [-L] [-l LOGFILE] [-r]
```

---

This command can take the following options:

| Option                       | Description                                                                                                                        |
|------------------------------|------------------------------------------------------------------------------------------------------------------------------------|
| <b>-h</b>                    | This option displays a usage message.                                                                                              |
| <b>-H</b>                    | This option displays the configuration file directives that the agent understands.                                                 |
| <b>-v</b>                    | This option displays version information.                                                                                          |
| <b>-f</b>                    | This option specifies not forking from the calling shell.                                                                          |
| <b>-a</b>                    | This option specifies logging addresses.                                                                                           |
| <b>-A</b>                    | This option specifies that warnings and messages should be appended to the log file rather than truncating it.                     |
| <b>-d</b>                    | This option specifies the dumping of sent and received UDP SNMP packets.                                                           |
| <b>-V</b>                    | This option specifies verbose display.                                                                                             |
| <b>-P <i>PIDFILE</i></b>     | This option specifies the use of a file ( <i>PIDFILE</i> ) to store the process ID.                                                |
| <b>-q</b>                    | This option specifies that information be printed in a more easily parsed format (quick print).                                    |
| <b>-D</b>                    | This option turns on debugging output.                                                                                             |
| <b>-p <i>NUM</i></b>         | This option specifies running on port <i>NUM</i> instead of the default: 161.                                                      |
| <b>-c <i>CONFFILE</i></b>    | This option specifies reading <i>CONFFILE</i> as a configuration file.                                                             |
| <b>-C</b>                    | This option specifies that the default configuration files not be read.                                                            |
| <b>-L</b>                    | This option prints warnings and messages to <b>stdout</b> and <b>err</b> .                                                         |
| <b>-s</b>                    | This option logs warnings/messages to <b>syslog</b> .                                                                              |
| <b>-r</b>                    | This option specifies not exiting if root-only accessible files cannot be opened.                                                  |
| <b>-I[-]/<i>INITLIST</i></b> | This option specifies a list of MIB modules to initialize (or not). Run <b>snmpd</b> with the <b>-Dmib_init</b> option for a list. |
| <b>-l <i>LOGFILE</i></b>     | This option prints warnings/messages to a file <i>LOGFILE</i> (by default, <i>LOGFILE=log/snmpd.log</i> ).                         |

## Using the SNMP command-line utilities

As an alternative to using an SNMP manager application, we supply several command-line utilities to test your SNMP installation and enable you to obtain information about your nShield module from the nCipher SNMP agent. These utilities support the **-h** (display a usage message) as described in the table above.

| Utility              | Description                                                                                           |
|----------------------|-------------------------------------------------------------------------------------------------------|
| <b>snmptest</b>      | This utility monitors and manages SNMP information.                                                   |
| <b>snmpget</b>       | This utility runs a single <b>GET</b> request to query for SNMP information on a network entity.      |
| <b>snmpset</b>       | This utility runs a single <b>SET</b> request to set SNMP information on a network entity.            |
| <b>snmpgetnext</b>   | This utility runs a single <b>GET NEXT</b> request to query for SNMP information on a network entity. |
| <b>snmptable</b>     | This utility obtains and prints an SNMP table.                                                        |
| <b>snmptranslate</b> | This utility translates SNMP object specifications into human-readable descriptions.                  |
| <b>snmpwalk</b>      | This utility communicates with a network entity using repeated <b>GET NEXT</b> requests.              |
| <b>snmpbulkwalk</b>  | This utility communicates with a network entity using <b>BULK</b> requests.                           |

**Note** These tools are general purpose SNMP utilities and are configurable for use with other SNMP agents. For more information on configuring and using these tools, refer to the NET-SNMP project Web site: <http://net-snmp.sourceforge.net/>.





## Appendix L: Product returns

If you need to return your nShield product, contact Support for instructions:  
<http://iss.thalesgroup.com/en/Support.aspx>



## Glossary

### Access Control List (ACL)

An Access Control List is a set of information contained within a key that specifies what operations can be performed with the associated key object and what authorization is required to perform each of those operations.

### Administrator Card Set (ACS)

Part of the Security World architecture, an Administrator Card Set (ACS) is a set of smart cards used to control access to Security World configuration, as well as recovery and replacement operations.

See also [Security World](#). See also [Operator Card Set \(OCS\)](#).

### Advanced Encryption Standard (AES)

The Advanced Encryption Standard (AES) is a block cipher adopted as an encryption standard by the US government and officially documented as US FIPS PUB 197 (FIPS 197). Originally only used for non-classified data, AES was also approved for use with for classified data in June 2003. Like its predecessor, the Data Encryption Standard (DES), AES has been analyzed extensively and is now widely used around the world.

Although AES is often referred to as *Rijndael* (the cipher having been submitted to the AES selection process under that name by its developers, Joan Daemen and Vincent Rijmen), these are not precisely the same cipher. Technically, Rijndael supports a larger range of block and key sizes (at any multiple of 32 bits, to a minimum of 128 bits and a maximum of 256 bits); AES has a fixed block size of 128 bits and only supports key sizes of 128, 192, or 256 bits.

See also [Data Encryption Standard \(DES\)](#).

### attended module

An attended module is one whose slot is used to read a smart card and send its contents down a secure channel to an unattended module.

See also [hardware security module \(HSM\)](#).

## **CAST**

CAST is a symmetric encryption algorithm with a 64-bit block size and a key size of between 40 bits to 128 bits (but only in 8-bit increments).

## **Data Encryption Standard (DES)**

The Data Encryption Standard (DES) is a symmetric cipher approved by NIST for use with US Government messages that are Secure but not Classified. The implementation of DES used in the module has been validated by NIST. DES uses a 64-bit block and a 56-bit key. DES keys are padded to 64 bits with 8 parity bits.

See also [Triple DES](#). See also [Advanced Encryption Standard \(AES\)](#).

## **Diffie-Hellman**

The Diffie-Hellman algorithm was the first commercially published public key algorithm. The Diffie-Hellman algorithm can only be used for key exchange.

## **digital signature**

A digital signature is an algorithm that proves the authenticity and authorship of a message in the same way that a signature is used to authenticate a written document. Digital signatures can be implemented with either symmetric or public key algorithms. nShield modules implement digital signatures with the public key algorithms DSA and RSA and also with the symmetric key algorithms CAST, DES, and Triple DES.

## **Digital Signature Algorithm (DSA)**

Also known as the Digital Signature Standard (DSS), the Digital Signature Algorithm (DSA) is a digital signature mechanism approved by NIST for use with US Government messages that are Secure but not Classified. The implementation of the DSA used by nShield modules has been validated by NIST as complying with FIPS 186.

## **Digital Signature Standard (DSS)**

See [Digital Signature Algorithm \(DSA\)](#).

## dynamic feature

A dynamic feature is a feature that is enabled by means of a software switch in the module's volatile memory. In contrast to the behavior of static features, if the module is reinitialized, a dynamic feature must also be re-enabled.

## ElGamal

ElGamal is a public key encryption algorithm similar to the Diffie-Hellman key exchange algorithm.

See also [Diffie-Hellman](#).

## Federal Information Processing Standards (FIPS)

The Federal Information Processing Standards (FIPS) were developed by the United States federal government for use by non-military government agencies and government contractors. FIPS 140 is a series of publications intended to coordinate the requirements and standards for cryptographic security modules, including both their hardware and software components.

All Security Worlds are compliant with FIPS 140-2. By default, Security Worlds are created to comply with FIPS 140-2 at level 2, but those customers who have a regulatory requirement for compliance with FIPS 140-2 at level 3 can also choose to create a Security World that meets those requirements.

For more details about FIPS 140-2, see <http://csrc.nist.gov/publications/fips/fips140-2/fips1402.pdf>.

## hardserver

The hardserver software controls communication between applications and nShield modules, which may be installed locally or remotely. It runs as a daemon on the host computer. The behavior of the hardserver is controlled by the settings in the hardserver configuration file.

## hardware security module (HSM)

A hardware security module (commonly referred to simply as a *module*) is a hardware device used to hold cryptographic keys and software securely.

## key blob

A key blob is a key object with its ACL and application data encrypted by a module key, a logical token, or a recovery key. Key blobs are used for the long-term storage of keys. Blobs are cryptographically secure; they can be stored on the host computer's hard disk and are only readable by units that have access to the same module key.

See also [Access Control List \(ACL\)](#).

## module

See [hardware security module \(HSM\)](#).

## module key

A module key is a cryptographic key generated by each nShield module at the time of initialization and stored within the module. It is used to wrap key blobs and key fragments for tokens. Module keys can be shared across several modules to create a larger Security World.

See also [Security World](#). See also [hardware security module \(HSM\)](#).

## Operator Card Set (OCS)

Part of the Security World architecture, an Operator Card Set (OCS) is a set of smart cards that is used to control access to application keys within a Security World. OCSs are protected using the Security World key, and therefore they cannot be used outside the Security World.

See also [Security World](#). See also [Administrator Card Set \(ACS\)](#).

## Rijndael

See [Advanced Encryption Standard \(AES\)](#).

## Secure Execution Engine (SEE)

The Secure Execution Engine (SEE) is a feature available on some Thales hardware security modules that allows application code to be executed within the security boundary of the module.

## Security World

The Security World technology provides an infrastructure for secure lifecycle management of keys. A Security World consists of at least one hardware security module, some cryptographic key and certificate data encrypted by a Security World key and stored on at least one host computer, a set of Administrator Cards used to control access to Security World configuration, recovery and replacement operations, and optionally one or more sets of Operator Cards used to control access to application keys.

See also [Administrator Card Set \(ACS\)](#). See also [Operator Card Set \(OCS\)](#).

## SEE

See [Secure Execution Engine \(SEE\)](#).

## SEE machine

An SEE machine is executable binary code that runs within the security boundary of the module, cross-compiled to native code on the host. It communicates with the nCipher core using a defined set of software interrupts that, in conjunction with their wrapper functions, provide a run-time environment that includes memory and thread management, as well as an interface for accepting and returning jobs and calling nCore API commands.

See also [Secure Execution Engine \(SEE\)](#).

## SEE World

An SEE World is the result of loading an SEE machine into a module.

See also [SEE machine](#). See also [Secure Execution Engine \(SEE\)](#). See also [Security World](#).

## Triple DES

Triple DES is a highly secure variant of the Data Encryption Standard (DES) algorithm in which the message is encrypted three times.

See also [Data Encryption Standard \(DES\)](#). See also [Advanced Encryption Standard \(AES\)](#).

## userdata

A **userdata** file contains initialization data passed in to the SEE machine at run-time.

See also [Secure Execution Engine \(SEE\)](#).



## Addresses

### Americas

900 South Pine Island Road, Suite 710, Plantation, Florida 33324, USA

Tel: +1 888 744 4976 or + 1 954 888 6200

[sales@thalessec.com](mailto:sales@thalessec.com)

### Europe, Middle East, Africa

Meadow View House, Long Crendon, Aylesbury, Buckinghamshire HP18 9EQ, UK

Tel: + 44 (0)1844 201800

[emea.sales@thales-esecurity.com](mailto:emea.sales@thales-esecurity.com)

### Asia Pacific

Units 4101, 41/F. 248 Queen's Road East, Wanchai, Hong Kong, PRC

Tel: + 852 2815 8633

[asia.sales@thales-esecurity.com](mailto:asia.sales@thales-esecurity.com)

### Internet addresses

|                              |                                                                                                                                  |
|------------------------------|----------------------------------------------------------------------------------------------------------------------------------|
| Web site:                    | <a href="http://www.thales-esecurity.com">www.thales-esecurity.com</a>                                                           |
| Support:                     | <a href="http://www.thales-esecurity.com/en/Support.aspx">www.thales-esecurity.com/en/Support.aspx</a>                           |
| Online documentation:        | <a href="http://www.thales-esecurity.com/Resources.aspx">www.thales-esecurity.com/Resources.aspx</a>                             |
| International sales offices: | <a href="http://www.thales-esecurity.com/en/Company/Contact%20Us.aspx">www.thales-esecurity.com/en/Company/Contact%20Us.aspx</a> |