



ENTRUST

nShield Security World

nShield Connect v12.40 User Guide for Unix

04 March 2024

Contents

Chapter 1: Introduction	15
Read this guide if	15
Terminology	15
Conventions	15
Typographical conventions	15
CLI command conventions	16
Model numbers	16
Security World Software	17
Document version numbers	19
Further information	19
Security advisories	19
Contacting nCipher Support	19
Recycling and disposal information	19
Chapter 2: Security Worlds	20
Security	21
Smart cards	21
Remote Operator	22
Remote Administration	22
Security World and an nShield Connect	24
NIST SP800-131A	24
FIPS 140-2 compliance	25
Common Criteria compliance	26
Platform independence	26
Application independence	26
Flexibility	27
Using the Security World key: module-protected keys	27
Using Operator Card Sets: OCS-protected keys	28

Using pass phrases for extra security	30
Using softcard-protected keys	31
NVRAM key storage	31
Scalability	32
Load-sharing	32
Robustness	33
Backup and recovery	33
Replacing a hardware security module	33
Replacing the Administrator Card Set	34
Replacing an Operator Card Set or recovering keys to softcards	34
KeySafe and Security Worlds	35
Applications and Security Worlds	36
The nCipher PKCS #11 library and Security Worlds	37
Risks	37
Chapter 3: The nShield Connect user interface	38
Overview of the front panel controls	38
Front panel controls	38
Using the front panel controls	40
Start-up information	41
Administrative control of the unit	41
Using a keyboard to control the unit	43
Chapter 4: Physical security of the nShield Connect	44
Tamper event	44
nShield Connect lid is closed	44
nShield Connect lid is open	45
Physical security checks	46
Replacing the fan tray module and PSU	48
Replacing the fan tray module	49
Replacing the PSU	49
Battery life when storing the nShield Connect	50

Disabling tamper detection functionality	50
Reactivating disabled tamper functionality	51
Chapter 5: Software installation	52
After software installation	52
Chapter 6: Client configuration	54
About user privileges	54
About client configuration	54
Remote file system (RFS)	55
HSM configuration	55
Client configuration	56
Basic HSM and remote file system (RFS) configuration	56
Configuring the Ethernet interfaces	56
Optionally configure hardserver interfaces	56
Configuring the remote file system (RFS)	57
Configuring log file storage	59
Setting the time and date	60
Keyboard layout	61
Configuring the nShield Connect to use the client	61
Remote configuration of additional clients	64
Configuring client computers to use the nShield Connect	64
Enrolling the client with the client configuration file	65
Enrolling the client from the command line	65
Client configuration utilities	66
Configuring the nShield Connect with configuration files	68
Setting up client cooperation	69
Useful utilities	71
Setting environment variables	72
Logging and debugging	72
Configuring Java support for KeySafe	73
Routing	73

Testing routes	73
Tracing network routes	75
Configuring an nShield Connect for Remote Administration	76
Configuring Dynamic Slots	77
Using Remote Administration with applications requiring cards in slot 0	78
Adjusting card removal detection timers to account for network characteristics	78
Using nethsmadmin to reboot an nShield Connect	79
Enabling and disabling remote mode changes	79
Enabling and disabling remote upgrade	80
Authorized Card List	81
Enabling optional features on the unit	81
Available optional features	82
Ordering additional features	86
Enabling features	86
Using multiple modules	87
Stopping and restarting the client hardserver	89
Resetting and testing the nShield Connect	89
Default configuration	89
Factory state	89
Testing the installation	90
Chapter 7: Creating and managing a Security World	91
Creating a Security World	91
The creation process	91
Security World files	92
Security World options	94
Creating a Security World using the nShield Connect front panel	97
Creating a Security World using new-world or KeySafe	99
Creating a Security World using new-world	100
Creating a Security World using KeySafe	106
After you have created a Security World	109

Displaying information about your Security World	110
Displaying information about a Security World with nfkminfo	110
Adding or restoring an HSM to the Security World	111
Adding an HSM to a Security World using the nShield Connect front panel	111
Adding an HSM to a Security World with new-world	112
Transferring keys between Security Worlds	112
Security World migration	116
About the migration utility	116
Migrating keys to an SP800-131A Security World	117
Troubleshooting	120
Erasing a module from a Security World	122
Erasing a module from the unit front panel	123
Erasing a module with new-world	123
Erasing a module with KeySafe	123
Erasing a module with initunit	124
Replacing an existing Security World	124
Deleting a Security World	124
Deleting the Security World using the nShield Connect front panel	125
Chapter 8: Managing card sets and softcards	126
Creating Operator Card Sets (OCSs)	127
Persistent Operator Card Sets	127
Time-outs	127
FIPS 140-2 level 3-compliant security worlds	128
Creating an Operator Card Set using the nShield Connect front panel	128
Creating an Operator Card Set using the command line	128
Creating an Operator Card Set with KeySafe	130
Creating softcards	132
Creating a softcard with ppmk	132
Creating softcards with KeySafe	133
Erasing cards and softcards	134

FIPS 140-2 level 3-compliant Security Worlds	134
Erasing card sets using the nShield Connect front panel	134
Erasing cards with KeySafe	134
Erasing cards using the command line	135
Erasing softcards	136
Viewing cards and softcards	136
Viewing card sets using the nShield Connect front panel	137
Viewing card sets with KeySafe	137
Viewing card sets using the command line	138
Viewing softcards	138
Verifying the pass phrase of a card or softcard	140
Changing card and softcard pass phrase	141
Changing known pass phrase	141
Changing unknown or lost pass phrase	144
Replacing Operator Card Sets	145
Replacing OCSs from the unit front panel	146
Replacing OCSs with KeySafe	146
Replacing OCSs or softcards with rocs	147
Replacing the Administrator Card Set	154
Replacing an Administrator Card Set using the nShield Connect front panel	155
Replacing an ACS with KeySafe	155
Replacing an Administrator Card Set with racs	156
Chapter 9: Application interfaces	158
Cryptographic Hardware Interface Library (CHIL)	158
Using keys	159
Generating keys	159
nCipherKM/JCA/JCE CSP	159
Overview of the nCipherKM/JCA/JCE CSP	159
Installing the nCipherKM/JCA/JCE CSP	159
keytool	163

Using keys	164
System properties	164
Compatibility	167
nCipher PKCS #11 library	168
Choosing functions	168
PKCS #11 library with Security Assurance Mechanism	169
Using the nCipher PKCS #11 library	170
nCipher PKCS #11 library environment variables	172
Checking the installation of the nCipher PKCS #11 library	182
How the nCipher PKCS #11 library protects keys	183
nShield native and custom applications	184
CodeSafe applications	184
Remotely loading and updating SEE machines	185
Chapter 10: Remote Operator	187
About Remote Operator	187
Configuring Remote Operator	187
Overview of configuring Remote Operator	187
Configuring HSMs for Remote Operator	188
Configuring slot import and export	188
Creating OCSs and keys for Remote Operator	189
Creating OCSs for use with Remote Operator	190
Loading Remote Operator Card Sets	190
Generating keys for use with Remote Operator	190
Configuring the application	191
Managing Remote Operator slots using the unit front panel	191
Chapter 11: Working with keys	192
Generating keys	192
Generating keys using the command line	192
Generating keys with KeySafe	194
Generating NVRAM-stored keys	194

Importing keys	195
Importing keys from the command line	195
Importing keys with KeySafe	196
Listing supported applications with generatekey	197
Retargeting keys with generatekey	197
Viewing keys	198
Viewing information about keys on the unit front panel	198
Viewing keys with KeySafe	198
Viewing keys using the command line	199
Discarding keys	200
Restoring keys	200
Appendix A: Using KeySafe	201
Setting up KeySafe	201
Starting KeySafe	202
About the KeySafe window	202
Sidebar	202
Menus	203
Module Status tree	203
Main panel area	206
Errors	206
Appendix B: Supplied utilities	208
Utilities for general operations	208
Hardware utilities	210
Test analysis tools	210
Security World utilities	211
CodeSafe utilities	213
PKCS # 11	214
nShield Connect utilities	215
Developer-specific utilities	217
Appendix C: Environment variables	218

Appendix D: Logging, debugging, and diagnostics	222
Logging and debugging	222
Environment variables to control logging	222
Logging and debugging information for PKCS #11	225
Debugging information for Java	226
Diagnostics and system information	227
nfdiag: diagnostics utility	228
nfkminfo: information utility	230
perfcheck: performance measurement checking tool	240
stattree: information utility	243
How data is affected when a module loses power and restarts	248
Appendix E: nShield Connect and client configuration files	250
Location of client configuration files	250
Location of module configuration files	250
Structure of configuration files	250
Sections only in module configuration files	252
nethsm_settings	252
nethsm_eth	252
nethsm_gateway	253
nethsm_route	253
nethsm_eth1_enable	253
nethsm_enable	253
cosmod	254
hs_clients	254
rfs_client	255
sys_log	255
config_op	255
ui_lockout	256
Sections in both module and client configuration files	256
server_settings	256

module_settings	259
server_remotecomms	260
server_startup	260
load_seemachine	261
slot_imports	262
slot_exports	262
dynamic_slots	263
slot_mapping	263
dynamic_slot_timeouts	263
Sections only in client configuration files	264
server_performance	264
nethsm_imports	264
rfs_sync_client	265
remote_file_system	265
remote_administration_slot_server_startup	266
Appendix F: Cryptographic algorithms	267
Symmetric algorithms	267
Asymmetric algorithms	268
FIPS information	268
Appendix G: Key generation options and parameters	270
Key application type (APPNAME)	270
Key properties (NAME=VALUE)	271
Available key properties by action/application	275
Appendix H: Checking and changing the mode on an nShield Connect	277
nShield Connect front panel controls	277
Available modes	277
Identifying the current mode	277
Checking the mode at the nShield Connect	277
Checking the mode using enquiry	278
Checking the mode by using KeySafe	279

Changing the mode	279
Changing the mode using the front panel controls of an nShield Connect	279
Changing the mode using remote mode and nopclearfail	280
Appendix I: Upgrading the nShield Connect image file and associated firmware	281
Version Security Number (VSN)	281
Key data	281
Upgrading the nShield Connect image file and firmware using the front panel	282
Remotely enabling dynamic feature certificates including nShield Connect client licenses	282
Upgrading the nShield Connect image file and firmware from a privileged client	284
Enabling and disabling remote upgrade	285
After firmware installation	285
Appendix J: SNMP monitoring agent	286
Installing the SNMP agent	287
Default installation settings	287
Do you already have an SNMP agent running?	287
Starting the SNMP agent	287
Basic configuration	288
Protecting the SNMP installation	288
Configuring the SNMP agent	288
The SNMP agent persistent configuration file	289
Agent Behaviour	290
agentaddress directive	290
agentgroup and agentuser directives	290
System information	291
USM users	291
Traditional access control	292
VACM configuration	295
Trap Configuration	298
SNMPv1 and SNMPv2 traps	298
SNMPv3 traps	299

Using the SNMP agent with a manager application	300
Manager configuration	300
MIB module overview	300
MIB functionality	300
Memory usage monitoring	302
Administration sub-tree overview	303
Statistics sub-tree overview	311
SNMP agent command-line	316
SNMP agent (snmpd) switches	316
Using the SNMP command-line utilities	317
Appendix K: Morse code error messages	319
Reading Morse code	319
Runtime library errors	320
Hardware driver errors	320
Maintenance mode errors	323
Operational mode errors	324
Appendix L: Application Performance Tuning	325
Job Count	325
Client Configuration	325
Highly Multi-threaded Client Applications	325
File Descriptor Limits	326
Appendix M: Product returns	327
Glossary	328

Chapter 1: Introduction

Read this guide if ...

Read this guide if you need to configure or manage:

- An nShield® Connect .
- An associated *Security World*. nCipher hardware security modules use the Security World paradigm to provide a secure environment for all your HSM and key management operations.

The nShield Connect is connected to a network by an Ethernet connection. Each HSM is configured to communicate with one or more client computers on the network. You can also configure clients to make use of any other network-connected HSMs on the network, as well as locally connected HSMs.

All nCipher HSMs 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 Installation Guide.
- You have installed your nShield Connect.

Note: Throughout this guide, the term Installation Guide refers to the particular Installation Guide for your product.

Terminology

The nShield Connect is referred to as the *nShield Connect*.

The internal security module of the nShield Connect is referred to as the *internal security module*.

When nCipher hardware security products are referred to in general, the term *hardware security module* or *HSM* is used. When specific products are referred to they are referred to by name.

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.

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

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

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 *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 nCipher hardware security devices.

Model number	Used for
NH2047	nShield Connect 6000
NH2040	nShield Connect 1500
NH2033	nShield Connect 500
NH2068	nShield Connect 6000+
NH2061	nShield Connect 1500+
NH2054	nShield Connect 500+
NH2075-B	nShield Connect XC Base
NH2075-M	nShield Connect XC Medium
NH2075-H	nShield Connect XC High
NH2082	nShield Connect XC SCAP

Security World Software

The Security World for nShield is a collection of programs and utilities, including the hardserver, supplied by nCipher to install and maintain your nCipher security system. For more information about the supplied utilities, see [Supplied utilities on page 208](#).

nCipher provides the firmware that runs on the nShield Connect, and software to run on each client computer. The nShield Connect is supplied with the latest version of the HSM firmware installed. For more information about:

- Upgrading the firmware, see [Upgrading the nShield Connect image file and associated firmware on page 281](#)
- Installing and configuring the software on each client computer, see the Installation Guide and [Client configuration on page 54](#).

Software architecture

nCipher supply the software to control communication between the unit and applications on the network.

Hardserver

The *hardserver* software controls communication between the internal security module and applications on the network.

Separate instances of the hardserver run on the unit and each client that is configured to work with the unit. There is a secure channel, known as the *impath*, between the two software instances, which forms a single secure entity for transferring data between the unit and the clients. See also [Compatibility issues on page 25](#)

The unit's hardserver is configured using the front panel on the unit, or by means of uploaded configuration data. Configuration data is stored on the unit and in files in a specially configured file system on each client computer. For more information about using:

- The front panel to configure the unit, see [The nShield Connect user interface on page 38](#)
- The specially configured file system to configure the unit and the client, see [Client configuration on page 54](#).

Remote file system (RFS)

Each unit uses a *remote file system* (RFS). You can configure the RFS on any computer, but it is normally located on the first client that is configured. The RFS contains:

- The master configuration information for the unit
- The Security World files
- The key data.

Do not copy the master configuration to file systems on other clients. You can copy Security World files and key data to other clients to allow you to manage the unit from more than one client. To make it available to the unit, copy to the RFS the data for Security Worlds, cards or keys that you create on a client that does not contain the RFS.

Security World Software

The Security World Software is a collection of programs and utilities that you install on the client and use to maintain the nCipher security system. The Security World Software includes the following:

- The appropriate installer for the client platform
- The client hardserver
- A set of utilities for configuring the nShield Connect
- A set of utilities and the KeySafe application for performing key management tasks on nCipher HSMs.

The nShield Connect is referenced and used by a utility or application in the same way as a local (PCI-connected) nCipher HSM.

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/hsm-ESN/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.

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 nCipher.

If you have installed the CodeSafe or CipherTools developer products, the Java Generic Stub classes, nCipherKM 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 installation media.

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

Security advisories

If nCipher Security becomes aware of a security issue affecting nShield HSMs, nCipher Security will publish a security advisory to customers. The security advisory will describe the issue and provide recommended actions. In some circumstances the advisory may recommend you upgrade the nShield firmware and or nShield Connect image file. In this situation you will need to re-present a quorum of administrator smart cards to the HSM to reload a Security World. As such, deployment and maintenance of your HSMs should consider the procedures and actions required to upgrade devices in the field.

Note: The Remote Administration feature supports remote firmware upgrade of nShield Solo and nShield Connects and remote ACS card presentation.

We recommend that you monitor the "nShield Announcements & Security Notices" section of the nCipher Security Support Portal, where any announcement of nShield Security Advisories will be made.

Contacting nCipher Support

To obtain support for your product, visit: <https://help.ncipher.com>.

Recycling and disposal information

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

The program enables you to ship an obsolete or excess nShield product line hardware security device to nCipher, 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, contact nCipher Support.

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

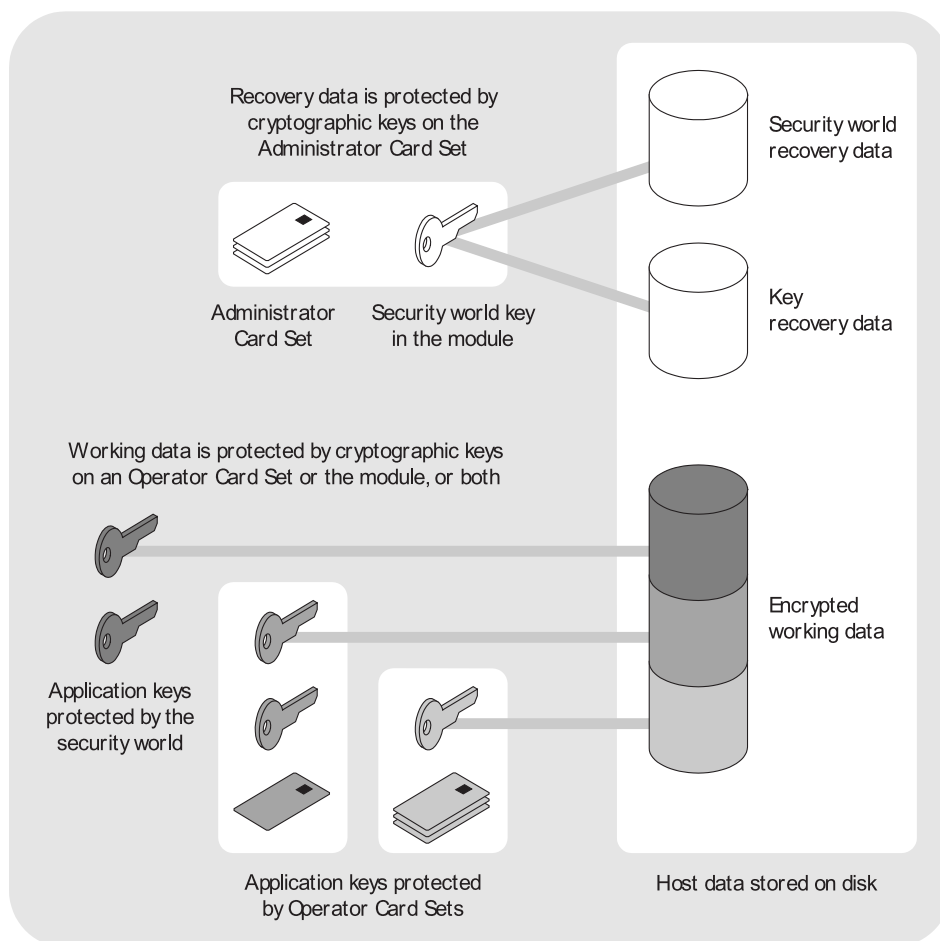
A Security World comprises:

- One or more nCipher nShield HSMs (such as the nShield Connect)
- An *Administrator Card Set* (ACS)
A set of Administrator smart cards used to control access to the 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 modules 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 nCipher key-management modules, keys are only ever available in plain text on secure hardware.



All Security Worlds rely on you using the security features of your operating system to control the users who can access the Security World and, for example, write data to the host.

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 a card from 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 another OCS.

Operator Cards employ the Security World key to perform a challenge-response protocol with the hardware security module. This means that Operator Cards are only useable by an HSM 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 from a persistent OCS
See [Using persistent Operator Card Sets on page 29](#) for more about persistent cards.
- The attended and unattended modules are in the same Security World

To achieve secure communication channels between the attended and unattended modules, the hardserver 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 25](#).

Remote Administration

The Remote Administration feature enables:

- Card holders to present smart cards to an HSM that is in a different location
For example, the card holder may be in an office, while the HSM is in a data center.
- All smart card operations to be carried out (apart from loading feature certificates)
Unlike the Remote Operator feature, Remote Administration supports the ACS and non-persistent OCS cards.
- Security World programs and utilities to be run and authorized remotely, without accessing an HSM card slot, when used in combination with a standard remote access solution

Once the software has been installed and the hardware security modules have been configured, Remote Administration enables full remote administration of Security Worlds and their HSMs.

Security World programs and utilities

As has always been the case, it is possible to run the Security World programs and utilities remotely using your preferred remote access solution, e.g. SSH or Remote Desktop. This means you can now run a utility like creatocs from a remote location and present the OCS to be created using a Remote Administration Client. The Remote Administration feature also adds the ability to change the mode of HSMs remotely using the nopclearfail utility. This means it is now possible to create a Security World remotely and perform future firmware upgrades.

HSM configuration

To support Remote Administration, HSMs have to be configured to support between zero (default) and 16 Dynamic Slots. These are virtual card slots that can be associated with a card reader connected to a remote computer. Dynamic Slots are in addition to the local slot of an HSM and any soft card slot that may be available. See [Configuring Dynamic Slots on page 77](#) for more information.

Note: To use Remote Administration, you may need to:

- Upgrade your HSM firmware to version 2.61.2 or later

The firmware is available on the installation media. See [Appendix I: Upgrading the nShield Connect image file and associated firmware on page 281](#).

Note: Dynamic Slots cannot be imported or exported as Remote Operator slots.

Remote Administration software

The following software is needed to allow remote card readers to be associated with an HSM:

- nShield Remote Administration Client software
Must be installed on the computer that has the card reader attached. See the nShield Remote Administration Client *User Guide* for more information.
- nShield Remote Administration Service software
Must be installed where it can access the appropriate HSM to provide communications between the card in the card reader and the HSM. See the *nShield Connect Installation Guide* for more about where to install the Remote Administration Service software.

When a card is inserted in a reader that is associated with an HSM, the nShield Remote Administration Client and the Remote Administration Service convey messages between the card and the HSM, allowing a secure channel of communications to be established.

nShield Remote Administration Cards

You must use nShield Remote Administration Cards with Remote Administration. These are smart cards that are capable of negotiating cryptographically secure connections with an HSM, using warrants as the root of trust. nShield Remote Administration Cards can also be used in the local slot of an HSM if required.

The use of nShield Remote Administration Cards is controlled by an Authorized Card List. If a card does not appear in the list, it cannot be used. See [Authorized Card List on page 81](#) for more information.

Card readers

nCipher supplies and recommends the use of the nShield Trusted Verification Device. This card reader allows the card holder to securely confirm the Electronic Serial Number (ESN) of the HSM to which they want to connect, using the nShield Trusted Verification Device display. As an alternative, and if it permitted by your security policy, card holders may use a standard smart card reader for ISO/IEC 7816 compliant smart cards.

Security World and an nShield Connect

To enable the secure transmission of data between an nShield Connect and each client, the hardserver uses an impath (see [Remote Operator on page 22](#)). Any nShield Connect on the network can load a Security World securely over the network, and access its keys, wherever the HSM is installed.

Security World and key data is stored on the file system of the nShield Connect, where it is updated whenever card or key operations are performed on the HSM. The data is also automatically transferred to the *remote file system* (RFS). If required, you can also share the data with client computers that use the Security World. If the Security World is updated on the nShield Connect, all the copies of the Security World data are also updated. For more information, see:

- [Remote file system \(RFS\) on page 55](#)
- [Configuring the remote file system \(RFS\) on page 57](#).

You can update the Security World on the host using:

- The nShield Connect front panel controls
- The command-line utilities
- KeySafe

You can also use these tools to create keys or cards. If you perform such tasks on a client other than the computer on which the RFS is installed, you must transfer the updated files to the RFS before they are available to the HSM.

NIST SP800-131A

You can create a Security World compliant with NIST (National Institute of Standards and Technology) SP800-131A (Special Publication 800-131A Revision 1, November 2015); see [Cipher suite on page 94](#). For details about NIST SP800-131A, see <http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-131Ar1.pdf>.

There is a migration tool available for transferring existing Security World data into an SP800-131A Security World; see [Migrating keys to an SP800-131A Security World on page 117](#).

Note: The `--disablepkcs1pad` option will only work on SP800-131A Security Worlds.

Compatibility issues

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

In most cases where versions of Security World 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 communication cannot be established between the HSM and clients or hosts:

- v11.50 or higher clients communicating with a v11.40 or lower nShield Connect, where the HSM client uses an nToken.
- v11.50 or higher nShield Connect communicating with a Remote File System (RFS) using v11.40 or lower.

Release version	Image versions nShield Connect	Security World Software ¹ v11.40	Security World Software ¹ v11.50 and later
Up to 11.40	Up to image version 0.3.5.	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.	RFS and client software upgrade required.	Supported.

¹ Previously known as nCipher 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.

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 nShield HSM initializes in that mode, conforming with the roles and services, key management, and self-test sections of the FIPS 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.

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

Common Criteria compliance

nShield Solo, Solo+, Connect and Connect+ are certified to Common Criteria v3.1 to EAL4+ AVA_VAN.5. To configure and operate the module in its evaluated configuration, the separate Common Criteria guides should be followed. Please contact nCipher Support for these guides, contact nCipher Support.

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

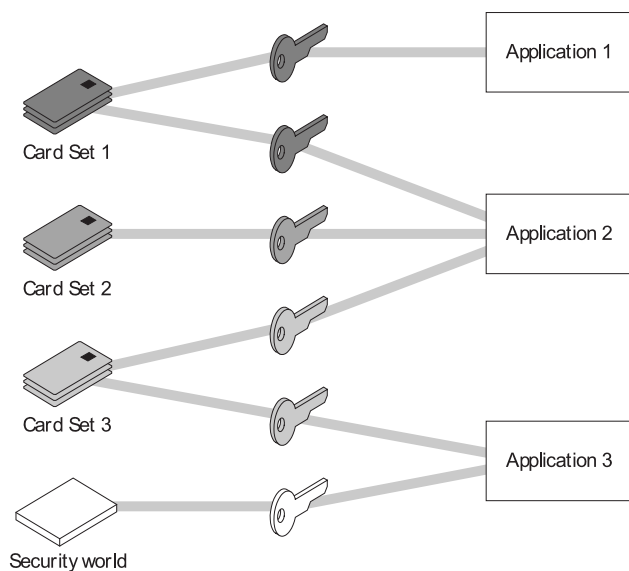
A Security World can protect keys for any applications correctly integrated with the Security World Software. Each key belongs to a specific application and is only ever used by that application. Keys are stored along with any additional data that is required by the application.

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 the OCSs in the Security World. This *Security World* key can be an AES (Advanced Encryption Standard) key which is the default and recommended option, or a Triple DES (Data Encryption Standard) key which is no longer recommended. The options are as follows:

Security World key / Cipher suite	Uses
Triple DES - not recommended	1024-bit DSA key with SHA-1 hash
AES 256 - original	1024-bit DSA key with SHA-1 hash
AES 256 (SP800-131A compliant) - default	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.

Note: The `--disablepkcs1pad` option will only work on SP800-131A Security Worlds.

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 module 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 module within the Security World to which the OCS belongs can read or erase 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 HSMs at the same time.

If you have a non-persistent OCS, you must leave one of the cards in an appropriate card slot of each HSM. This should only be done if it is in accordance with the security policies of your organization.

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

- K to 1
- N at least equal to the number of the HSMs you want to use.

You can then insert single cards from the OCS into the appropriate card slot of each HSM 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 modules. For

example, with a 3/5 OCS, you can load keys onto 3 hardware security modules 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 95](#).

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 34](#)).

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 95](#).

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 card reader. The keys protected by this card are removed from the memory of the hardware security module as soon as the card is removed from the card reader, which provides added security.

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.

This enables:

- The use of the same smart card in several hardware security modules at the same time
- Several users to load keys onto the same hardware security module 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 module:

- When the application that loaded the OCS closes the connection to the hardware security module
- After a time limit that is specified when the card set is created
- When an application chooses to remove a key
- When the HSM is cleared. See [Manually removing keys from an HSM on page 30](#) for more information
- If there is a power loss to the module, for example, due to power outage.

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 modules for such applications is by having multiple smart cards in an OCS.

Although the hardware security module 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.

Manually removing keys from an HSM

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

- Run the command `nopclearfail --clear --all`
- Press the Clear button of the hardware security module
- Turn off power to the hardware security module

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

Persistence is a permanent 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 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 module that belongs to the Security World to which the card belongs. For more information, see [Changing card and softcard pass phrase on page 141](#).

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

Maximum pass phrase length

Note: The maximum pass phrase length limitation is not applicable to software versions before Security World Software v11.72.

Pass phrases are limited to a maximum length of 254 characters, when using the following commands:

- `new-world`
- `createocs`
- `cardpp`
- `ppmk`
- `racs`

Other commands are unaffected.

You can still use and edit existing pass phrases that are longer than 254 characters.

Prior to Security World Software v11.72, 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.

nCipher recommends that your password only contains 7-bit ASCII characters:

A-Z, a-z, 0-9, ! @ # \$% ^ & * - _ + = [] { } | \ : ' , . ? / ` ~ " < > () ;

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 `/opt/nfast/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.

NVRAM key storage

Application keys protected by an nShield Connect are stored in an encrypted format and copied automatically to the remote file system. A hardware security module, such as an nShield Connect, and/or an OCS protects the keys, as described in the preceding sections. You can also store application keys within the nonvolatile memory of a suitable hardware security module.

NVRAM-stored keys are encrypted in exactly the same way as application keys that are protected by the unit. The encrypted application key on the unit is replaced by a file containing the name of the NVRAM file that contains the application key. This file allows applications to use NVRAM-stored keys in the same way as keys stored in the remote file system. You can protect an NVRAM-stored key with either the Security World or an OCS.

Note: NVRAM-stored keys differ from standard application keys only in their storage location. They still require protection by the unit or an OCS.

Use of an NVRAM-stored key is the same as for any other key protected by an nShield Security World.

NVRAM key storage:

- Offers no additional security benefits above those offered by the standard Security World Software mechanisms
- Is available for only a limited number of keys
- Reduces a Security World's ability to offer load-balancing and recovery

- Requires backup and recovery procedures in addition to any that you implement for data stored on the client computer.



Do not store keys in NVRAM unless you must do so to satisfy regulatory requirements.

NVRAM key storage was introduced only for those users who must store keys within the physical boundary of a hardware security module (such as an nShield Connect) 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.

Scalability

A Security World is scalable. You can add multiple hardware security modules 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 clients:

- Ensure each client has at least one hardware security module configured
- Copy the Security World data to each client
- load the Security World onto the hardware security modules for each client.

If you create cards or keys in a Security World from a client rather than on the hardware security module (using the command line or KeySafe), you must transfer the files from the client to the remote file system, unless the client is already on the same computer as a remote file system.

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

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).

If you want to make cards or keys which are normally created from the client available from the front panel of the hardware security module, we recommend that you use client cooperation to automate the copying of files to the device.

Load-sharing

If you have more than one hardware security module configured with a client, 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. Two approaches are supported:

- API specific load-sharing modes
- HSM Pool mode: a more generic load-sharing approach for module protected keys introduced with module firmware version 2.65.2.

Note: Some applications may not be able to make use of these features.

HSM Pool mode is supported on all major APIs except Java (i.e. nCipherKM JCA/JCE CSP). When HSM Pool mode is enabled for an API, the application sees the HSMs in the Security World as a single resource pool. A significant benefit is that when a failed HSM is restored to the Security World or a new HSM is added to the Security World, it is automatically added to the resource pool making it available for cryptographic operations without restarting the application (i.e. failback support). The pool of HSMs can be viewed as a single resource using the command `enquiry --pool`.

Note: `Module #1: Not Present` indicates that there are no HSMs in the pool.

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 file system and remote file system of the hardware security module 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 module, 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 modules to a Security World. At all other times, you must store these cards in a secure environment.

Note: In 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 module

If you have a problem with a hardware security module, you can replace it with another hardware security module of the same type by loading the Security World data on the remote file system onto

the replacement device. Alternatively, you may be able to erase the Security World from the device that has the problem, return the device to its default state and then reload the Security World on the same device.

If you have more than one hardware security module configured with a client and you use one of the load-sharing modes identified above, then your client application is resilient to the failure of individual hardware security modules. If you use HSM Pool mode, then an nShield Connect can be replaced and returned to the HSM pool without restarting the client application.

For information about replacing a hardware security module, see [Adding or restoring an HSM to the Security World on page 111](#).

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 front panel controls of the nShield Connect
- The KeySafe **Replace Administrator Card Set** option
- `racs` utility (see [Replacing the Administrator Card Set on page 154](#)).

Note: You should also use the front panel controls of the nShield Connect, `racs`, or the KeySafe **Replace Administrator Card Set** option to migrate the ACS from standard nShield cards to nShield Remote Administration Cards. Authorization needs to take place using the local slot of an HSM.

A hardware security module does not store recovery data for the ACS. Provided that K is less than N for the ACS, and you have at least K cards available, a hardware security module can re-create all the keys stored on the device even if the information from other 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 95](#)). 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 an OCS, you can use either:

- The front panel controls of the nShield Connect
- The `rocs` command-line utility.

Note: It is not possible to recover PKCS #11 keys using the front panel controls of the nShield Connect. You must 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 [Using KeySafe on page 201](#).

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.

When you use KeySafe to create cards or keys, the data is written to the Key Management Data directory on the computer on which you run KeySafe. An nShield Connect can only use this data when it is transferred to the remote file system (if it is on a different computer), from where it is loaded automatically onto the unit. For this reason, you may find it most convenient to run KeySafe on the same computer as the remote file system.

Although you may 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 hardware security modules, 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 computers.

KeySafe enables you to:

- Create a Security World and its ACS, as 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 device to a Security World
- Remove a hardware security module 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 module 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 *Supplied utilities on page 208*. Current versions of these tools can be used interchangeably with the current version of KeySafe.

You can also perform all the tasks available from KeySafe using the front panel controls of the unit, except for adding, importing and deleting keys.

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 <https://www.ncipher.com>.

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 nCipher hardware security modules and Security Worlds.

For more information about the nCipher range of Integration Guides:

- Visit <https://www.ncipher.com>
- 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.

The nCipher PKCS #11 library treats a smart card from an OCS in the current Security World as a PKCS #11 token. The current PKCS #11 standard only supports tokens that are part of a 1-of-N card set, however the nCipher PKCS #11 library has vendor specific extensions that support K/N card sets, see *nCipher PKCS #11 library with the preload utility on page 172*.

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 nCipher or directly from the smart card manufacturer.
 - Note:** nShield Remote Administration Cards can only be supplied by nCipher.
- Never insert a smart card used with nCipher key management products into a smart card reader you do not trust.
- Never insert a smart card reader you do not trust into your hardware security module.
- Never tell anyone your pass phrase.
- Never write down your pass phrase.
- Never use a pass phrase that is easy to guess.

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.

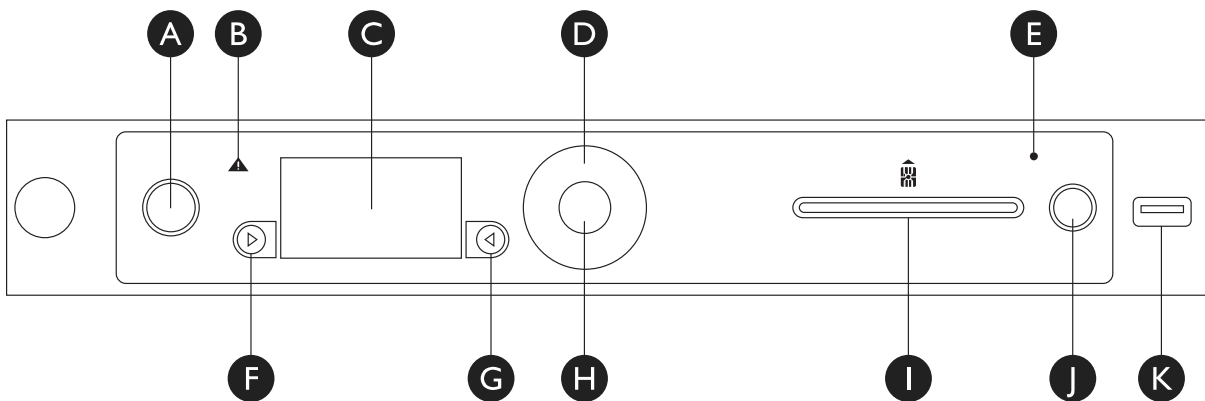
Chapter 3: The nShield Connect user interface

This chapter describes the nShield Connect user interface, including the front panel controls. You are also shown how to use a keyboard to control the unit.

Overview of the front panel controls

Front panel controls

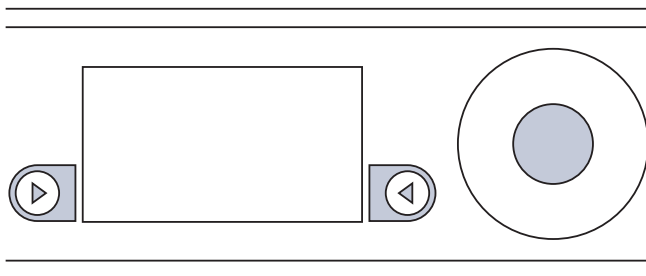
Figure 3. nShield Connect front panel controls



Label	Description
A	Power button
B	Warning LED (orange)
C	Display screen
D	Touch wheel
E	Status indicator LED (blue)
F	Display navigation button (left)
G	Display navigation button (right)
H	Select button
I	Slot for smart cards
J	Clear button
K	USB connector—nShield Connect only

Display screen and controls

Figure 4. nShield Connect: display screen and controls



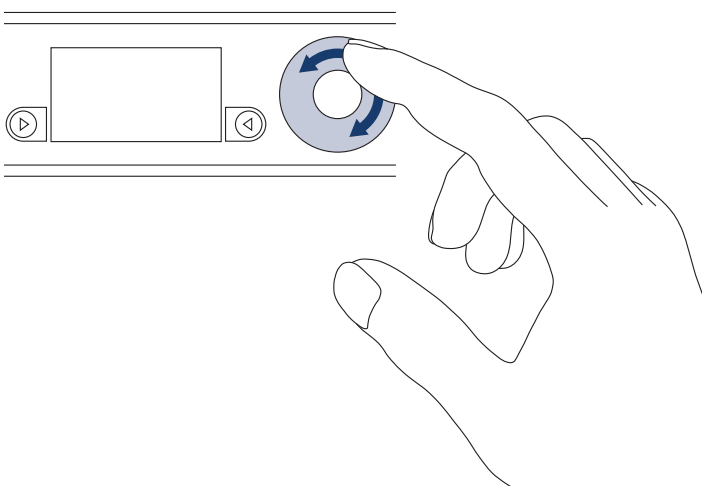
When the unit is powered, the display screen displays a menu or a dialog.

Each menu or dialog includes onscreen navigation labels that appear at the bottom of the display screen, on either side next to the display navigation buttons. Press the button next to the label to perform the action specified by the label.

To go back to the previous dialog or menu screen, use the navigation button to the left of the screen. To confirm a dialog value or select a menu option, either:

- Press the navigation button to the right of the screen.
- Touch the Select button.

Figure 5. nShield Connect touch wheel



Use the touch wheel to changes values or move the cursor on the display screen. To confirm a value, touch the Select button.

Menu screens

You can access menus from the display screen.

Menus are displayed as a list of selectable options. An onscreen arrow points to the currently selectable option. If the menu has more than four options, an arrow indicates the direction in which more options are available.

To select a menu option:

1. Move the indicator arrow up or down with the touch wheel.
2. When the indicator arrow points to the option you want to select, either:
 - Press the navigation button to the right of the screen (labeled onscreen as **SELECT**).
 - Touch the Select button.

At the top right of the display screen, a number sequence indicates the path to the current option. The last digit of the sequence shows the location of the menu you are currently viewing. The top level menu has no numbers, but when you select the System menu, the number **1** is shown.

The preceding digits in the sequence show the position of each option in turn that was selected in previous menu screens to reach the current menu. For example, the sequence **1-2** shows that the indicator is on the second option of the menu that was reached by selecting the first option on the top-level menu.

For a map of the menu screens, see the *Installation Guide*.

Dialogs

For some tasks, a dialog is displayed onscreen. When the dialog opens, the cursor is in the first field. To change and then enter values:

1. Use the touch wheel to change the displayed value of the fields.
2. Touch the Select button to enter the displayed value and move to the next field in the dialog.

Repeat the procedure to enter all necessary values in the dialog.

Information display

When you use a dialog to request information (for example, a log or details of a key), there is often too much information to display onscreen. In such cases, only the first part of the information is displayed.

To view the rest of the information:

- Use the touch wheel to scroll the displayed information in the direction indicated by the onscreen arrows.
- When an **Options** label is displayed, press the right-hand navigation button to see a menu of navigation options. You can normally choose to go to the top, to the bottom, or to a specified line in the display.

The numbers of the lines currently being displayed onscreen are shown at the left of the screen. They are followed in parentheses (()) by the total number of lines available for display.

Using the front panel controls

You can use the front panel controls to configure the unit and to perform other tasks described in this guide. When the unit is working over the network with another computer (a client computer), you can program and control the unit as if it were part of the client computer.

Note: If the unit is powered down while you are logged in, you are logged out automatically.

Start-up information

When you turn on power to the unit and it has completed its initialization, the lower part of the display screen shows basic start-up information about the unit.

There is a series of start-up information topics available. By default, the first displayed topic is the current **System time**. Use the touch wheel to view the other start-up information topics.

Administrative control of the unit

You can view and control the status of the unit by using the front panel controls and menu options.

Tasks	Action
Understand and control the power status of the unit	<p>Use the Power button to power up the unit.</p> <p>If the Power button is not illuminated, the unit is not powered. The Power button flashes intermittently as the unit powers up. It also flashes when the unit is in standby mode. For more information about the Power button, see the <i>Installation Guide</i>.</p>
Control access to the unit	<p>You can control access to the menus on the unit and the Power button on the front panel by using System > System configuration > Login settings.</p> <p>When UI Lockout with OCS has been enabled, you must log in with an authorized Operator Card before you can access the menus. You can still view information about the unit on the start-up screen. When you are logged in, you can log out and leave the unit locked.</p> <p>When UI Lockout without OCS has been enabled, you cannot access the menus, but you can still view information about the nShield Connect on the start-up screen. The only way to disable this setting (apart from returning the HSM to factory state) is to push an updated configuration file to the nShield Connect. See Configuring the nShield Connect with configuration files on page 68 and ui_lockout on page 256 for more information.</p> <p>Power button lockout can be enabled and disabled independently when UI Lockout allows access to the menus.</p>
Unlock the unit	<p>When UI Lockout with OCS has been enabled and you have logged out, the display screen displays the label Login next to the right-hand navigation button. Press the right-hand navigation button, then insert an Operator Card that has been authorized for login, and follow the onscreen instructions.</p>
Log out of the unit	<p>Select Logout.</p> <p>This option is not available if UI Lockout with OCS has not been enabled.</p>
Put the unit in standby mode	<p>Press the Power button or select System > Shutdown/Reboot > Shutdown.</p>

Tasks	Action
Restore the unit to its original configuration	Select System > System configuration > Default config .
Restore the unit to its factory state	Select System > Factory state .
Clear the memory of the internal hardware security module	Use the Clear button or select HSM > HSM reset .
View information about the current state of the internal hardware security module	Select HSM > HSM information .
View information about the current state of the system	See the next section.
Set the Real-Time Clock on the unit	Select Security World mgmt > Admin operations > Set secure RTC . This option requires Administrator Card authorization.
Change the mode of the unit	Select HSM > Set HSM mode . <ul style="list-style-type: none"> • Select Operational mode to run the unit normally. • Select Initialisation mode to configure the unit with software utilities rather than the front panel.

Viewing the current status of the unit

To view information about the current state of the system, from the main menu select **System > System information**. Select an option to view the associated information as follows:

Option	Description
View system log	Displays the system log.
View hardserver log	Displays the module hardserver log.
Display tasks	Displays the tasks that the system is currently performing.
Component versions	Displays the version numbers of the various system software components.
View h/w diagnostics	Displays the following environmental information about the module: <ul style="list-style-type: none"> • The current temperature at the left and right sensors • The minimum and maximum previous temperature at each sensor • The voltage on each power rail • The speed of each fan.
View tamper log	Displays the tamper log.
View unit id	Displays the ID of the unit.

Viewing the mode of the unit

See *Identifying the current mode on page 277* for more about viewing the mode of the unit.

Using a keyboard to control the unit

You can connect a keyboard to the USB connector on the front panel. You can connect either a US or a UK keyboard. To configure the unit for your keyboard type, select **System > System configuration > Keyboard layout** and then choose the keyboard type you require.

When you have connected a keyboard and configured the unit for its use, you can enter numbers and characters directly into the display. You can also control the unit by using the following keystrokes:

Keystroke	Use
F1	Same as pressing the left-hand navigation button on the front panel.
F2	Same as pressing the right-hand navigation button on the front panel.
F3	Same as touching the Select button.
Esc	Same as pressing the left-hand navigation button on the front panel.
Enter	Where the Select button is active, same as pressing Select: where Button B is active, same as pressing Button B.
Up arrow	Moves the indicator upwards in a menu.
Down arrow	Moves the indicator downwards in a menu.
Tab	Moves the cursor to the next field in a dialog.
Shift-Tab	Moves the cursor to the previous field in a dialog.
PgUp	Displays the previous screen.
PgDn	Displays the next screen.

Chapter 4: Physical security of the nShield Connect

This chapter provides a brief overview of the physical security measures that have been implemented to protect your nShield Connect. You are also shown how to:

- Check the physical security of your nShield Connect
- Disable and re-enable tamper detection functionality on your nShield Connect.

The tamper detection functionality on the nShield Connect provides additional physical security, over and above that provided by the holographic security seal, and alerts you to tampering in an operational environment. There is a removable lid on top of the nShield Connect, protected by the security seal and tamper switches. To prevent the insertion of objects into the nShield Connect, baffles are placed behind vents.

To optimize their effectiveness, use the physical security measures implemented on the nShield Connect in association with your security policies and procedures. For more information about creating and managing security policies, see the *Security Policy Guide* on the NIST CMVP website.

Note: Currently, the FIPS 140-2 Level 3 boundary is at the internal module. Future software releases may move the FIPS boundary so that it includes the entire nShield Connect chassis.

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

Tamper event

The nShield Connect offers several layers of tamper protection. The outer boundary of the box is tamper-responsive. When tampered, the unit ceases to provide cryptographic functionality, alerts the operator of the event, and ultimately forces the operator to reset the unit to factory defaults. Movements/vibrations, or replacing the fan tray module or a PSU, does not activate the tamper detection functionality. Tamper-responsiveness can be disabled, if desired. If a tamper event does occur, you can use the Security World data stored on the RFS and the Administrator Card Set to recover the keys and cryptographic data.

nShield Connect lid is closed

If the nShield Connect is powered, a tamper event has occurred, and the lid is closed, the following message is displayed onscreen:

```
** TAMPER DETECTED **
Consult User Guide.
Check physical seals.
Reset to factory state
or disable tamper
detection first?
RESET          DISABLE
```

When you see this message, examine your unit for physical signs of tampering (see [Physical security checks on page 46](#)).

If you discover signs of tampering do *not* attempt to put the unit back into operation. The date and time of the tamper event are recorded in the log (see [Logging, debugging, and diagnostics on page 222](#)).

Note: The tamper-responsiveness circuitry has a Real Time Clock that runs independently from the main nShield Connect clock. The times associated with events in the tamper log may have slight offsets to times recorded in other log files.

If there are signs of tampering, and the tamper event occurred:

- During transit from nCipher, contact Support.
- After installation, refer to your security policies and procedures. For more information about creating and managing security policies, see the *Security Policy Guide*.

If there are no signs of tampering, you can *either*:

- Reset the nShield Connect to a factory state

Or:

- Disable the tamper detection functionality and then reset the nShield Connect to a factory state.

Note: Do not disable tamper detection unless asked to do so by Support

You require a quorum of the Administrator Card Set (ACS) to restore the key data and reconnect the nShield Connect to the network. If you chose to disable the tamper protection circuitry when you reset the nShield Connect to a factory state, a cautionary warning is displayed prior to the request for card authorization. For information about re-enabling disabled tamper detection functionality, see [Disabling tamper detection functionality on page 50](#).

nShield Connect lid is open

If the nShield Connect is powered, a tamper event has occurred, and the lid is open, the following message is displayed onscreen:

```
** TAMPER DETECTED **
Unit lid is open
Replace lid or disable
tamper detection.
                DISABLE
```

An open lid indicates that the physical security of the unit is compromised. You may want to examine your unit for other physical signs of tampering (see [Physical security checks on page 46](#)). Do not attempt to put the unit back into operation.

The date and time of the tamper event are recorded in the log files (see [Logging, debugging, and diagnostics on page 222](#)). If the tamper event occurred:

- During transit from nCipher, contact Support.
- After installation, refer to your security policies and procedures. For more information about creating and managing security policies, see the *Security Policy Guide* on the NIST CMVP website.

If you choose to replace the lid of the nShield Connect, the onscreen message changes to the tamper alert message that is given when the lid is closed. Closing the lid provides you with the option to reset the unit to a factory state without disabling tamper detection functionality. If the lid remains open, all button presses other than those made using the right-hand navigation button are ignored. Pressing the right-hand navigation button disables tamper responsiveness and resets the nShield Connect to a factory state.

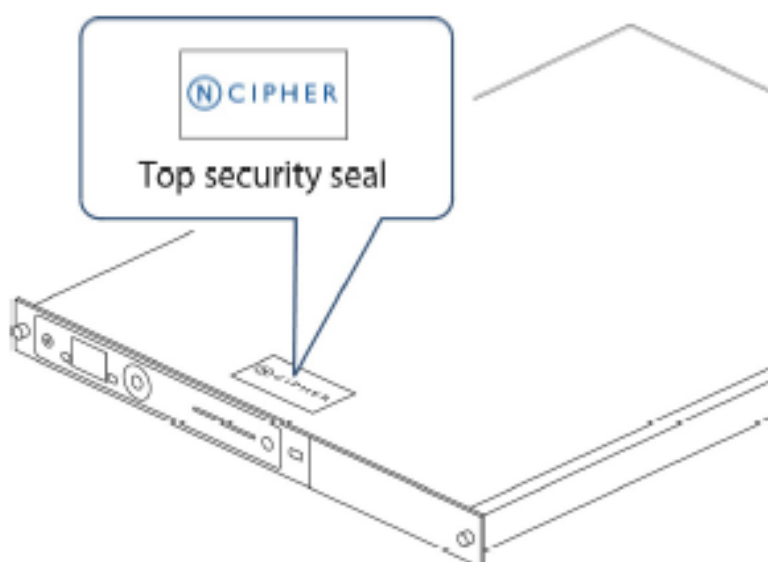
Physical security checks

Check the physical security of your nShield Connect before installation and at regular intervals afterwards. For an alternative presentation of the physical security checks described here, see the *Physical Security Checklist*. For more information about tamper events, and what actions to take if you discover signs of tampering, see [Tamper event on page 44](#).

To determine if the security of the nShield Connect is compromised:

1. Check that the physical security seal is authentic and intact. Look for the holographic foil bearing the nCipher logo. Look for cuts, tears and voiding of the seal. The seal is located on the top of the nShield Connect chassis.

Figure 6. Location of security seal



For information about the appearance of intact and damaged security seals, see the *Physical Security Checklist*.

2. Check that the metal lid remains flush with the nShield Connect chassis.

Figure 7. Metal lid in correct position

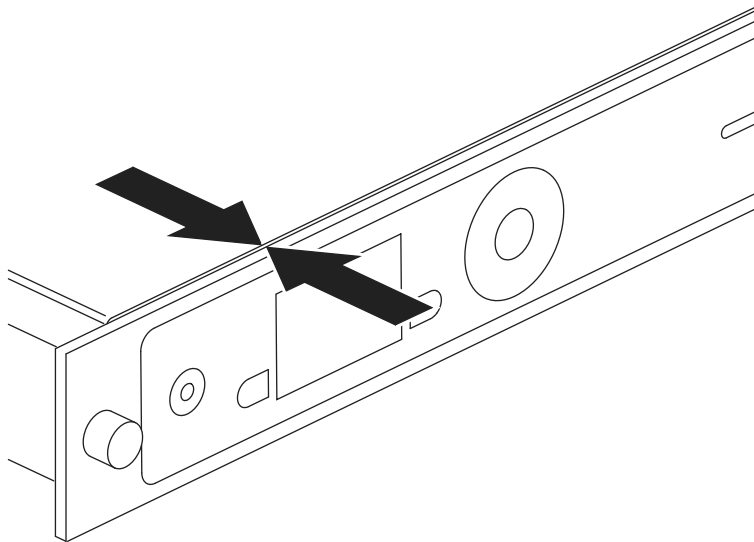
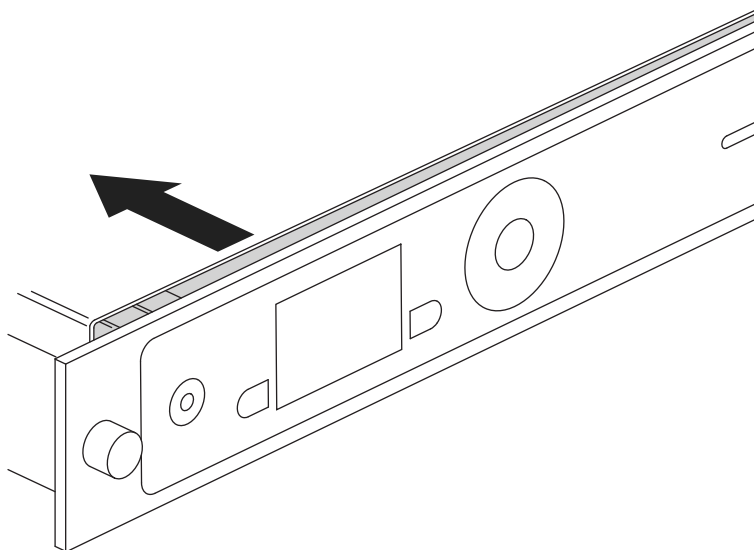


Figure 8. Metal lid in an incorrect position (pulled back)



3. Check all surfaces — the top, bottom and sides of the nShield Connect — for signs of physical damage.
4. Check that there are no signs of physical damage to the vents, including attempts to insert objects into the vents.

Figure 9. Front vent (to the right of the front panel controls)

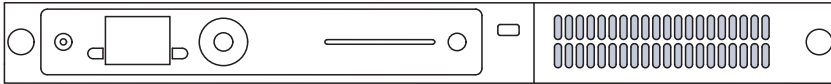
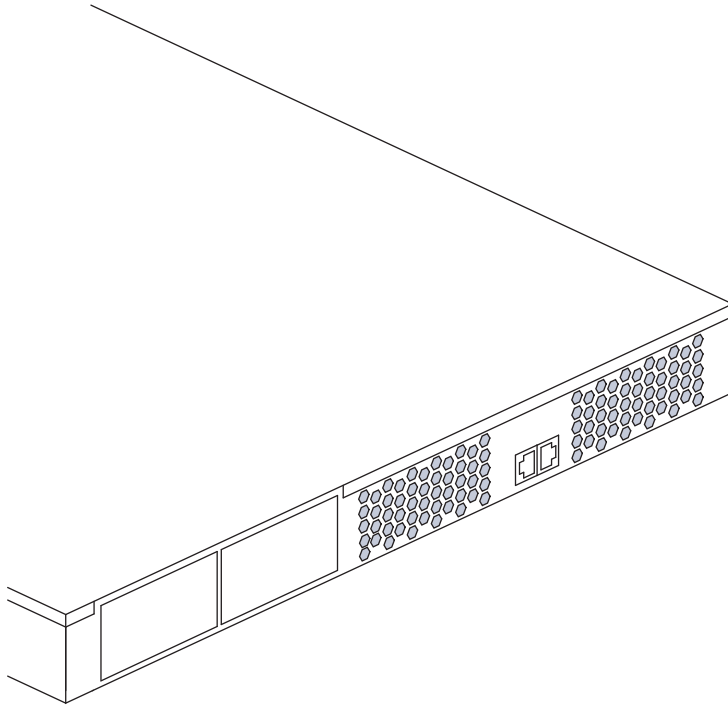


Figure 10. Rear vent (to the right of the power supplies)



Replacing the fan tray module and PSU

You can replace the fan tray module or a power supply unit (PSU) **without** activating a tamper event as both are outside the security boundary. You can access:

- The PSU(s) from the rear of the nShield Connect.
- The fan tray module through the removable front vent.

Should a problem occur with the fan tray module or a PSU, contact Support **before** taking further action. For more information about replacing the fan tray module or a PSU, see the *Fan Tray Module Installation Sheet* or the *Power Supply Unit Installation Sheet*.

Note: The fan tray module contains back-up batteries providing reserve capacity (a guaranteed minimum of 3 years) for tamper detection functionality even when the nShield Connect is in an unpowered state.

The tamper protection circuitry remains fully operational if the nShield Connect is placed on standby while a replacement operation is performed (whether you are replacing the fan tray module or one of the two PSUs, in the case of dual PSU units).

Note: Provided that the nShield Connect is connected to the mains power supply, the nShield Connect displays an onscreen error message when back-up battery power is low. The Status LED also displays a low power warning. For more information, see the *Installation Guide*.

Replacing the fan tray module

It is not necessary to remove mains power to replace a fan tray module (we recommend that you power down the unit into standby state using the front panel power button). However, if mains power is removed then a replacement fan tray module **must be installed within an hour** to ensure that a tamper event is not activated. If put in standby state the time required to change fan tray module is unlimited. For more information about replacing the fan tray module, see the *Fan Tray Module Installation Sheet*.

Fan tray module error messages

If you receive any of the following error messages on the nShield Connect display, accompanied by the orange warning LED, follow the related action in the table below:

Error message	Action
Single fan fail	Contact Support
Many fans fail	Replace fan tray
Battery power low	Consider replacing fan tray during the next scheduled service/maintenance period.
System Shutdown	Replace fan tray
Both fans in a pair had failed	

If the error message is **Single fan fail**, the nShield Connect can continue operating under the specified operating environment. Although you are advised to contact Support, the limited nature of such a failure means you can replace the fan tray module at your convenience.

If the error message is **Many fans fail**, you must replace the fan tray module immediately.

If the error message is **Battery Power low**, this indicates that one or both of the backup batteries located on the fan tray module (required only when the nShield Connect is removed from mains power) is running low.

The **Battery Power low** indication has no detrimental affect on the nShield Connect performance whilst the unit remains powered. nCipher recommend customers should consider replacing the fan tray module during the next service/maintenance.

If two fans fail from a redundant pair, the nShield Connect will display the error message **Many fans have failed** for a few seconds and it will then shutdown. On reboot, the nShield Connect will then display the error messages **System Shutdown** and **Both fans in a pair had failed**. In this situation the fan tray module must be replaced immediately.

Replacing the PSU

If you have a dual PSU nShield Connect, you do not have to remove power to the functioning PSU while replacing a faulty PSU. Tamper detection functionality will operate normally throughout the PSU replacement process. If you decide to remove power from both PSUs, tamper detection functionality will continue to operate normally for at least 3 years, as the fan tray module provides back-up capacity

for this circuitry. For more information about replacing the PSU, see the *Power Supply Unit Installation Sheet*.

PSU error messages

If a PSU fails, an orange warning LED comes on and an error message is displayed on the nShield Connect display. Although you are advised to contact Support, the unit can continue to operate normally and you can replace the failed PSU at your convenience. There is no need to power down the unit when you replace the failed PSU.

In addition to the orange warning LED, an audible warning is given when a PSU fails on an nShield Connect. The audible warning is turned off when you navigate to the Critical errors screen.

Battery life when storing the nShield Connect

If a nShield Connect has been in storage for an extended period of time the fan tray module may need replacement.

nCipher guarantees a *minimum* battery life of three years, even if the nShield Connect remains in storage and is not connected to the mains power supply during this time.

Disabling tamper detection functionality

We do *not* recommend disabling tamper detection functionality. However, there may be circumstances in which disabling this functionality is required. For example, you may want to:

- Bring the management of the nShield Connect into line with your existing operational procedures
- Standardize the management task, if you are managing a mixture of older and newer units.

To disable tamper responsiveness on your nShield Connect:

1. From the main menu select **System > System configuration > Tamper config**. You are shown the following dialog:

Figure 11. Tamper responsiveness enabled dialog

```
Tamper responsiveness
is currently enabled.

This matches nCipher's
recommendation.
Do you want to disable
tamper responsiveness?
```

2. Press the right-hand navigation button to disable tamper responsiveness.
3. You are asked to confirm the change. Press the right-hand navigation button again.

Tamper responsiveness is disabled and the unit is reset to a factory state. To restore the key data and reconnect the nShield Connect to the network you must present a quorum of the ACS.

Reactivating disabled tamper functionality

If your circumstances change, for example, your operational procedures are updated, you may want to re-enable tamper responsiveness.

To re-enable tamper responsiveness on your nShield Connect:

1. From the main menu select **System > System configuration > Tamper config**.
2. You are shown the following dialog:

Figure 12. Tamper responsiveness disabled dialog

```
Tamper responsiveness  
is currently DISABLED.
```

```
nCipher recommends  
that tamper should be  
enabled. Re-enable  
tamper responsiveness?
```

3. Press the right-hand navigation button to enable tamper responsiveness.
4. You are asked to confirm the change. Press the right-hand navigation button again.

Tamper responsiveness is enabled and the unit is reset to a factory state. To restore the key data and reconnect the nShield Connect to the network you must present a quorum of the ACS. Tamper events are logged and never erased.

Chapter 5: Software installation

See the appropriate Installation Guide for your nShield module for more about installing the Security World software.

After you have installed the software, you must complete further Security World creation, configuration and setup tasks before you can use your nShield environment to protect and manage your keys.

After software installation

Note: The Installation Guide provides brief explanations of how to perform all the post-installation tasks listed in this section. If this is the first time you are installing a unit and the Security World Software, or you are unfamiliar with the process, we recommend following the steps outlined in the Installation Guide.

After you have successfully installed the Security World Software, as described in the Installation Guide), complete the following steps to finish preparing your HSM for use:

1. Ensure that your public firewall is set up correctly. See the Installation Guide for your HSM for more information about firewall settings.
2. Perform the necessary basic HSM-client configuration tasks, as described in *Basic HSM and remote file system (RFS) configuration on page 56*.
3. Create and configure a Security World, as described in *Creating a Security World on page 91*.
4. Create an OCS, as described in *Creating Operator Card Sets (OCSs) on page 127*.
5. Complete additional necessary HSM-client configuration tasks:
 - a. To configure the unit so that it works with the client machine, see *Configuring the nShield Connect to use the client on page 61*.
 - b. To configure client computers so that they work with the unit, see *Configuring client computers to use the nShield Connect on page 64*.

Note: For this release, you must generate a new client configuration file to take advantage of new functionality. To generate a new client configuration file, back up your existing configuration file and run the command `cfg-mkdefault`. This generates a template for the configuration file into which you can copy the settings from your old configuration file.

- d. To enable the TCP sockets for Java applications (including KeySafe), run the command:

```
config-serverstartup -sp
```

For more information, see *Client configuration utilities on page 66*.

When all additional HSM configuration tasks are completed, you can:

1. Stop and then restart the hardserver, as described in *Stopping and restarting the client hardserver on page 89*.

2. Test the installation and configuration. See the Installation Guide for your HSM for more information.

Chapter 6: Client configuration

This chapter describes how to configure the internal security module of the nShield Connect and the client to communicate with each other, after you have installed the HSM and the Security World Software.

For more information about installing the HSM, see the *Installation Guide*. If you are configuring an HSM and client for the first time, or you want to complete a basic installation quickly, see the *Installation Guide*.

Note: The nShield Connect provides significant performance improvements, and can be deployed successfully with existing nShield products. Customers wishing to take advantage of these performance improvements **must** update their client machines with the latest Security World Software.

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.

There are three levels of user:

- Superuser
- nfast group user
- normal

Typically, normal users can carry out operations involving Security Worlds, cardsets and keys, but not create Security Worlds, keys and cardsets. nfast group users have enhanced access, enabling them to create Security Worlds, cardsets and keys. For example, encrypted copies of keys are held in `kmdata` (`/opt/nfast/kmdata`). Normal users only have read access to the files, whereas nfast group users have read and write access, enabling them to create and use keys. nfast group users can also change the mode of an HSM remotely.

Superuser access (e.g., root) is required for such tasks as software installation, starting and stopping the hardserver and SNMP

About client configuration

Note: You can add more HSMs to a client and more clients to an HSM at any time.

The HSM and a client communicate by means of the hardserver, which handles secure transactions between the HSM and applications that run on the client. You must configure:

- Each client hardserver to communicate with the HSM that the client needs to use
- The HSM to communicate with clients that are allowed to use the HSM.

Information about the current configuration of the HSM or a client is stored in configuration files that are stored in specified file systems on the clients and on the HSM. For more information about the contents of configuration files, see *nShield Connect and client configuration files on page 250*.

For information about configuring the HSM by importing an edited configuration file, see *Configuring the nShield Connect with configuration files on page 68*.

Remote file system (RFS)

Each HSM must have a remote file system (RFS) configured. The RFS contains master copies of all the files that the HSM needs:

- The HSM configuration file
- Feature-enabling certificates
- The encrypted Security World and key data for Security Worlds created on the HSM.

The RFS normally resides on a client computer, but it can be located on any computer that is accessible on the network.

For more information about setting up the remote file system, see *Configuring the remote file system (RFS) on page 57*.

HSM configuration

The data that defines the configuration of the HSM hardserver is stored in a file on the HSM. This file is automatically:

- Updated when the HSM is configured from the front panel
- Exported to the remote file system (RFS) directory.

Each HSM has separate configuration files on the RFS, stored in the directories with names of the form `/opt/nfast/kmdata/hsm-ESN/config`, where **ESN** represents the electronic serial number of the HSM from which the files were exported. These directories can contain the following files:

Option	Description
<code>config</code>	The master configuration file. This contains the current configuration for the HSM. It is always present in the directory.
<code>config-name</code>	An alternative configuration file saved by the system.
<code>config.new</code>	A hand-edited configuration file that can be read by the HSM.

You normally configure the HSM using the front panel controls. However, in some cases (for example, if you need to configure an HSM remotely, or if you are importing a number of clients), you may prefer to edit the exported configuration file and then re-import the file into the HSM. For more information, see:

- *Configuring the nShield Connect with configuration files on page 68*
- *nShield Connect and client configuration files on page 250*.

Client configuration

The data that defines the configuration of the client hardserver is stored in a file on the client's file system.

You must load the configuration file for the configuration to take effect. For information about loading a client configuration remotely, see *Remote configuration of additional clients on page 64*.

You can configure a client to use multiple HSMs. All the HSMs configured for use by a client can fail over if the application that uses them is set up appropriately.

For more information about the contents of the client configuration file, see *nShield Connect and client configuration files on page 250*.

Note: You can also configure the client's hardserver by setting environment variables, as described in *Setting environment variables on page 72*. Environment variable settings override settings in the client configuration files.

Basic HSM and remote file system (RFS) configuration

After installing the HSM hardware and software, there are several HSM and RFS configuration tasks you must perform. You perform these RFS tasks before:

- Creating the Security World and an Operator Card Set (OCS)
- Completing the process of configuring the HSM and client to work together.

Configuring the Ethernet interfaces

The HSM communicates with one or more clients. Each client is an Ethernet connected computer that has the Security World Software installed and configured. You must supply Internet Protocol (IP) addresses for the HSM and the client. Contact your system administrator for this information if necessary.

To configure the Ethernet interfaces (IPv4 and IPv6), see the *nShield Connect Installation Guide*.

Optionally configure hardserver interfaces

By default, the hardserver listens on all interfaces. However, you can alter the hardserver settings. Altering the hardserver settings would prove necessary, for example, if you wanted to connect one of the Ethernet interfaces to external hosts.

Ensure that you have configured the Ethernet interfaces on the HSM before attempting to configure the hardserver. See the Installation Guide for more information about configuring the Ethernet interfaces.

You can configure the following options to specify network interfaces on which the hardserver listens:

Option	Description
All interfaces	This option (which is the default) specifies that the hardserver listens on all interfaces.

Option	Description
<i>IP address of interface #1</i>	This option specifies that the hardserver listens only on interface 1. This option only appears if interface 1 has been configured.
<i>IP address of interface #2</i>	This option specifies that the hardserver listens only on interface 2. This option only appears if interface 2 has been configured.
Will not listen	This option specifies that the hardserver does not listen on any interfaces.

To define the interface and port on which the hardserver listens:

1. From the main menu, select **System > System configuration > Hardserver config**. The following screen appears:

```

Hardserver config
Select network I/F
hardserver listens on:
  All interfaces
Select TCP port: 9004
      CANCEL          FINISH

```

2. Select the network interfaces on which the hardserver is to listen.



For security reasons, do not allow the hardserver to listen on any interface that is to connect to the public Internet.

3. Press the Select button to move to the TCP port field, and set the port on which the hardserver is to listen. The default is 9004.

Note: Ensure that your firewall settings are consistent with your port settings. See the Installation Guide for more about firewall settings.

4. When the network interface and port are correct, press the right-hand navigation button.
5. Press the right-hand navigation button again to continue.
6. You are asked if you wish to reboot the system now or later. Press the right-hand navigation button to reboot now.

Configuring the remote file system (RFS)

The remote file system (RFS) contains the master copy of the HSM Security World data for backup purposes. The RFS can be located on either a client or another network-accessible computer where the Security World Software is installed. If the RFS is on a client, the same file structure also contains the configuration files for that client.



We recommend that you regularly back up the entire contents of the RFS. The `kmdata` directory is required to restore the HSM, or a replacement, to its current state, in case of failure.

You can specify a new remote file system, and modify or delete an existing remote file system configuration. To create or modify a remote file system configuration, specify the IP address of the computer on which the file system resides.

Note: You must have created an RFS on the client computer before you specify the IP address of the client.

For more information about the RFS and its contents, see:

- [Remote file system \(RFS\) on page 55](#)
- [Security World files on page 92](#).

The HSM must be able to connect to TCP port 9004 of the RFS. If necessary, modify the firewall configuration to allow this connection on either the RFS itself or on a router between the RFS and the HSM.

Obtain the following information about the HSM before you set up an RFS for the first time:

- The IP address
- The electronic serial number (ESN)
- The hash of the κ_{NETI} key (HK_{NETI}). The κ_{NETI} key authenticates the HSM to clients. It is generated when the HSM is first initialized from factory state.

If your network is secure and you know the IP address of the HSM, you can use the `anonkneti` utility to obtain the ESN and hash of the κ_{NETI} key by giving the following command on the client computer:

```
anonkneti mmm.mmm.mmm.mmm
```

In this command, `mmm.mmm.mmm.mmm` is the IP address of the HSM. The command returns output in the following form:

```
A285-4F5A-7500 2418ec85c86027eb2d5959fef35edc5e1b3b698f
```

In this example output, `A285-4F5A-7500` is the ESN and `2418ec85c86027eb2d5959fef35edc5e1b3b698f` is the hash of the κ_{NETI} key.

Alternatively, you can find this information on the HSM startup screen. Use the touch wheel to scroll to the appropriate information.

When you have the necessary information, set up an RFS as follows:

1. Prepare the RFS on the client computer (or another appropriate computer) by running the following command on that computer:

```
rfs-setup --force mmm.mmm.mmm.mmm EEEE-SSSS-NNNN keyhash
```

In this command:

- `mmm.mmm.mmm.mmm` is the IP address of the HSM
 - `EEEE-SSSS-NNNN` is the ESN of the HSM
 - `keyhash` is the hash of the K_{NET} key.
2. On the HSM display screen, use the right-hand navigation button to select **System > System configuration > Remote file system**, and enter the IP address of the client computer on which you set up the RFS.

Note: Leave the port number at the default setting of 9004.

To modify or delete an RFS at a later date, select **System configuration > Remote file system**, and then select the required action.

Note: You can allow non-HSM hardware security modules to access the remote file system and share Security World and key data that is stored in the `kmdata` directory in the same way as the HSM. Clients that access data in this way are described as *cooperating clients*. To configure client cooperation, you need to know the details of each client including IP address and ESN.

After you have defined the remote file system, the HSM configuration files are exported automatically to the remote file system. For more information about configuration files, see [nShield Connect and client configuration files on page 250](#).

Configuring auto push

You can allow a configuration to be pushed automatically from the RFS to the HSM, or you can fetch the updated configuration manually from the HSM.

The *auto push* feature allows future HSM configuration to be performed remotely (that is, without access to the front panel of the HSM). To enable auto push, use the right-hand navigation button to select **System > System configuration > Config file options > Allow auto push** and select **ON**.



Setting the **Allow auto push** option to **ON** allows the client computer on which the RFS resides to change the HSM configuration file and gives the client the power to make configuration changes that are normally only available through the HSM secure user interface.

If you do not want to enable auto push, you can fetch the updated configuration file manually from the HSM, select **System > System configuration > Config file options > Fetch configuration**.

Configuring log file storage

You can choose to store log files on both the HSM and RFS or on the HSM only.

To configure log file storage, use the right-hand navigation button to select **System > System configuration > Log config**. Then select one of:

1. **Log** to store log files on the HSM only.
2. **Append** to store log files on both the HSM and remote file system.

We recommend selecting **Append** because if you select **Log** you can only view the log file from the nShield Connect front panel. Moreover, the log file stored on the HSM is cleared every time it is powered down.

Setting the time and date

To set the time and date on the HSM as UTC:

1. Use the right-hand navigation button to select and display the **System** menu:

```

                                1-1
System configuration
System information
Login settings
Upgrade system
Factory state
Shutdown/Reboot
BACK                SELECT

```

2. Select **System configuration** to display the **System configuration** menu:

```

                                1-1-1
Network config
Hardserver config
Remote file system
Client config
Resilience config
Config file options
BACK                SELECT

```

3. Use the touch wheel to move the arrow to **Date/time setting**, and press the right-hand navigation button to select it. The **Set system date** screen is displayed:

```

Set system date
Please enter the
current UTC date as
DD/MM/YYYY:
    27/ 5/2013
CANCEL                NEXT

```

4. For each date field, use the touch wheel to set the value and move the cursor to the next field. When you have completed all the fields, press the right-hand navigation button to confirm the date. The **Set system time** screen is displayed:

```

Set system time
Please enter the
current UTC time as
hour/mins/seconds:
    18:08:19
CANCEL                FINISH

```

5. If you change the UTC date or time, you must reboot the HSM before the changes take effect.

Note: Setting the time and date of the HSM as UTC does not reset the value of the Real Time Clock (RTC) on the HSM. The UTC date and time settings are used only in log messages.

Keyboard layout

You can connect a keyboard to the USB connector on the nShield Connect front panel. This enables you to control the nShield Connect using a special set of keystrokes instead of the standard front panel controls.

You can connect either a US or a UK keyboard. To configure the nShield Connect for your keyboard type, select **System > System configuration > Keyboard layout** and then choose the keyboard type you require.

Configuring the nShield Connect to use the client

You must inform the HSM hardserver of the location of the client computer.

If nToken hardware is installed on the client, you can configure the client to use it. If a client attempts to connect to the nShield Connect when an nToken is in use, the HSM not only examines the client's IP address, but also requires the client to identify itself using a signing key.

Note: If an nToken is installed on a client, it can be used to both generate and protect a key that is used for the impath communication between the nShield Connect and the client. Thus a strongly protected key is used at both ends of the impath.

The client configuration process varies slightly depending on whether you are enrolling the client with or without an nToken:

1. On the nShield Connect front panel, use the right-hand navigation button to select **System > System configuration > Client config > New client**.

The following screen is displayed:

```
Client configuration
Please enter your
client IP Address
  0. 0. 0. 0
CANCEL          NEXT
```

2. Enter the IP address of the first client, and press the right-hand navigation button. You are asked to choose the permissions for the client:

```
Client configuration
Please choose the
client permissions
Unprivileged
BACK          NEXT
```

3. Use the touch wheel to display the type of connection between the internal security module of the nShield Connect and the client. The following options are available:

Option	Description
Unprivileged	Privileged connections are never allowed.
Priv. on low ports	Privileged connections are allowed only from ports numbered less than 1024. These ports are reserved for use by root on Unix-based systems.
Priv. on any ports	Privileged connections are allowed on all ports.



A privileged connection is required to administer the nShield Connect, for example to initialize a Security World. If privileged connections are allowed, the client can issue commands (such as clearing the HSM) which interfere with its normal operation. We recommend that you allow only unprivileged connections unless you are performing administrative tasks.

- When you have selected a connection option, press the right-hand navigation button. The following screen is displayed:

```
Client configuration
This client is not
configured to use an
nToken. Do you want to
enroll with an nToken?
      NO
BACK      FINISH
```

You must choose whether to enroll the client with an nToken, or without one:

- To enroll the client without nToken authentication, select **No** and press the right-hand navigation button.
- To enroll the client with nToken authentication, you must first confirm the nToken authentication key:
 - a. On the client, open a command line window, and run the command:

```
ntokenenroll -H
```

This command produces output of the form:

```
nToken module #1
nToken ESN:      3138-147F-2D64
nToken key hash: 691be427bb125f387686
                 38a18bfd2eab75623320
```

- b. Compare the nToken key hash returned by `ntokenenroll` with the hash on the nShield Connect.

Note: Write the hash down, or ensure that you can see the key hash displayed on the nShield Connect as you work on the client.
- c. On the nShield Connect, enter the number of the port on which the client is listening and press the right-hand navigation button. (The default is 9004.)
The nShield Connect display shows information of the following form, identifying the client by its ESN and displaying a key hash:

```
Client nnnnnnnnn reported the key hash:
691be427bb125f387686
38a18bfd2eab75623320
Is this EXACTLY right?
    Yes  No
CANCEL          FINISH
```

- d. Compare the hash displayed by the nShield Connect with the hash that was previously reported by `ntokenenroll` on the client. If there is an exact match, select **Yes** and then press the right-hand navigation button to configure the client.
- e. The nShield Connect displays a message reporting that the client has been configured. Press the right-hand navigation button again.

To modify or delete an existing client, select **System > System configuration > Client config** and perform the appropriate procedure.

If you want to use multiple clients with the nShield Connect, you must enable additional client licenses (see *Enabling optional features on the unit on page 81*). When you have additional client licenses enabled, to configure more clients, repeat the appropriate steps of the procedure described in this section for each client.

Remote configuration of additional clients

After you have configured the first client for your nShield Connect and provided you have enabled auto push, you can add additional clients remotely.

Note: Before you can use multiple clients with the nShield Connect, you must enable the additional clients as described in *Enabling optional features on the unit on page 81*.

To add clients remotely:

1. Save the configuration file of the nShield Connect as `config.new` in the same directory .
2. Edit the configuration file `config.new` and add a new entry to the `hs_clients` section to contain the details of the client to be added.

The following two entries must exist in the configuration file:

```
addr=client_IP
clientperm=permission_type
```

Where `client_IP` is the IP address of the client you are adding and `permission_type` is what type of commands the client can issue (`unpriv`, `priv` or `priv_lowport`).

If the client is using an nToken, two additional entries will need to be added to the configuration file:

```
esn=nToken_ESN
keyhash=nToken_keyhash
```

Where `nToken_ESN` is the ESN of the client's nToken and `nToken_keyhash` is the hash of the key that the client's nToken should authenticate itself with.

Note: Each client entry after the first must be introduced by a line consisting of one or more hyphens.

3. Load the updated configuration file on to the nShield Connect. To do this, run the following command:

```
cfg-pushnethsm --address=module_IP_address new_config_file
```

In this command, `module_IP_address` is the IP address of the nShield Connect and `new_config_file` is the location of the updated configuration file (`config.new`).

With auto push enabled, you can load the configuration file using the nShield Connect front panel. The configuration file (`config.new`) must be in the directory `/opt/nfast/kmdata/hsm-ESN/config` on the remote file system. To do this, select **System > System configuration > Config file options > Fetch configuration**.

Configuring client computers to use the nShield Connect

Each client computer must be configured to use the internal security module of your nShield Connect. There are two methods for achieving this:

- Enrolling the client with the configuration file.
- Enrolling the client with command-line utilities.

Enrolling the client with the client configuration file

The client configuration files are in the directory `/opt/nfast/kmdata/config` on the client computer's file system.

Note: For this release, you must generate a new client configuration file to take advantage of the new functionality. To generate a new client configuration file, back up your existing configuration file and run the command `cfg-mkdefault`. This generates a template for the configuration file into which you can copy the settings from your old configuration file.

The `nethsm_imports` section defines the network HSMs that the client imports (See "[nethsm_imports](#)" on page 264). It can also be set up by the `nethsmenroll` utility.

- Edit the following mandatory fields: `local_module`, `remote_ip`, `remote_esn`, `remote_keyhash` and `privileged`. The default value for `remote_keyhash` (40 zeros) specifies that no authentication should occur.
- The **ESN** and **hash** of the HSM to import can be retrieved by running the command

```
anonkneti remote_ip
```

- The fields, `timelimit` and `datalimit`, can be changed as required.
- If the client is to be enrolled with an nToken, open a command line window, and run the command: `ntokenenroll --H`. This command produces output of the form:

```
nToken module #1
nToken ESN: 3138-147F-2D64
nToken key hash: 691be427bb125f387686
                 38a18bfd2eab75623320
```

- Enter the nToken's **ESN** in the field `ntoken_esn` in the config file.
- Each HSM entry after the first must be introduced by a line consisting of one or more hyphens, i.e. `"_"`.
- At the command line run the command `cfg-reread`, to reload the hardserver's configuration.
- Verify that the client can use the nShield Connect by running `enquiry`, which reports the HSM's status.

For information about configuration file contents, see [nShield Connect and client configuration files on page 250](#).

Enrolling the client from the command line

The `nethsmenroll` command-line utility edits the client hardserver's configuration file to add the specified nShield Connect. For more information about the options available to use with `nethsmenroll`, read the following section [Client configuration utilities](#), or run the command:

```
nethsmenroll --help
```

To retrieve the nShield Connect's **ESN** and **HKNETI**, run the command

```
anonkneti <Unit IP>
```

This command produces output of the form:

```
3138-147F-2D64 691be427bb125f38768638a18bfd2eab75623320
```

If the nShield Connect's **ESN** and **HKNETI** are not specified, **nethsmenroll** attempts to contact the HSM to determine what they are, and requests confirmation.

1. If you are enrolling the client *with* an nToken, run the command:

```
nethsmenroll --ntoken-esn <nToken ESN> [Options] --privileged <Unit IP> <Unit ESN> <Unit KNETI HASH>
```

2. If you are enrolling the client *without* an nToken, run the command:

```
nethsmenroll [Options] --privileged < Unit IP> < Unit ESN> < Unit KNETI HASH>
```

These commands produce output of the form:

```
OK configuring hardserver's nethsm imports.
```

Client configuration utilities

We provide the following utilities for client configuration:

Utility	Description
nethsmenroll	This utility is used to configure the client to communicate with the nShield Connect.
config-serverstartup	This utility is used to configure the client's hardserver to enable TCP sockets.

nethsmenroll

The `nethsmenroll` command-line utility edits the client hardserver's configuration file to add the specified nShield Connect. If the nShield Connect's **ESN** and **HKNETI** are not specified, `nethsmenroll` attempts to contact the HSM to determine what they are, and requests confirmation.

Usage:

```
nethsmenroll [Options] --privileged <nShield Connect IP> <nShield Connect ESN> <nShield Connect KNETI HASH>
```

Options:

<code>-m, --module=MODULE</code>	Specifies the local HSM number to use (default is 0 for dynamic configuration by hardserver).
<code>-p, --privileged</code>	Causes the hardserver to request a privileged connection to the nShield Connect (default <code>unprivileged</code>).
<code>-r, --remove</code>	Deconfigures the specified nShield Connect.
<code>-f, --force</code>	Forces reconfiguration of an nShield Connect already known.
<code>--no-hkneti-confirmation</code>	Does not request confirmation when automatically determining the nShield Connect's ESN and HKNETI . Note: CAUTION! This option is potentially insecure and should only be used on secure networks where there is no possibility of a man-in-the-middle attack.
<code>-v, --verify-nethsm-details</code>	When the ESN and HKNETI have been provided on the command line, verifies that the selected HSM is alive, reachable and matches those details.
<code>-P, --port=PORT</code>	Specifies the port to use when connecting to the given nShield Connect (default <code>9004</code>).
<code>-n, --ntoken-esn=ESN</code>	Specifies the ESN of the LOCAL HSM to use as nToken when connecting to this nShield Connect (default <code>empty</code> , i.e. no nToken authentication used)

config-serverstartup

The `config-serverstartup` command-line utility automatically edits the `[server_startup]` section in the local hardserver configuration file in order to enable TCP ports for Java and KeySafe. Any fields for which values are not specified remain unchanged. After making any changes you are prompted to restart the hardserver.

Run `config-serverstartup` using commands of the form:

```
config-serverstartup [OPTIONS]
```

For more information about the options available to use with `config-serverstartup`, run the command:

```
config-serverstartup --help
```

Configuring the nShield Connect with configuration files

You normally configure the nShield Connect using the front panel controls. However, you can choose to export a configuration file, edit it, and then re-import it. You may prefer this approach, for example, if you are importing a number of clients.

There are two ways to configure the nShield Connect by importing edited configuration files.

If you need to configure the nShield Connect regularly, but do not require physical access to it every time you configure it, follow these steps:

1. Ensure that the nShield Connect is configured to accept a configuration from a remote computer.

To do this:

1. From the main menu, select **System configuration > Config file options > Allow auto push**.
2. Select **ON** then specify the IP address of the computer from which to accept the configuration.

The nShield Connect can now accept configuration from the computer with the specified IP address.

2. Create a copy of the configuration file from the remote file system in the `/opt/nfast/kmdata/hsm-ESN/config` directory on the remote computer.
3. Edit the file so that it contains the required configuration. For information about the contents of the nShield Connect configuration file, see [nShield Connect and client configuration files on page 250](#).
4. Run the `cfg-pushnethsm` utility on the updated configuration file, specifying the updated file and the IP address of the nShield Connect to load the new configuration. To do this, use a command similar to the following:

```
cfg-pushnethsm --address= module_IP_address config_file
```

In this command, *module_IP_address* is the nShield Connect on which to load the configuration and *config_file* is the path to, and name of the updated configuration file.

5. Check that the configuration file on the RFS has been updated with the required changes.

Note: As of the nShield Connect image file version `iiii`, the nShield Connect checks that it is able to update the configuration file on the RFS before applying changes to its own configuration. If you want the nShield Connect to apply the configuration changes even if it cannot update the configuration file on the RFS, for example, because the RFS is going to be moved to a new IP address, use the `--no-rfs-check` command line option:

```
cfg-pushnethsm --address= module_IP_address config_file --no-rfs-check
```

You will have to copy the configuration file to the RFS manually.

To permit nShield Connect configuration remotely from the RFS computer without enabling the auto push option:

1. Save the file `/opt/nfast/kmdata/hsm-ESN/config` as `config.new` in the same directory and edit it to contain the configuration you require. For more information about the contents of the configuration file, see *nShield Connect and client configuration files on page 250*.
2. On the nShield Connect, from the main menu select **System > System configuration > Config file options > Fetch configuration**.

Whichever method you use, the updated configuration file becomes the new nShield Connect configuration. It is automatically copied back to the file `/opt/nfast/kmdata/hsm-ESN/config` on the remote file system.

Setting up client cooperation

If you do not need to allow multiple clients access to your remote file system (RFS), you only need to follow the instructions provided in *Configuring the remote file system (RFS) on page 57* to initialize your RFS. If you need to allow other clients to access your RFS (that is, able to access the RFS to share key data), complete the following steps:

1. Configure the RFS to accept access by cooperating clients:
 - For every authenticated client (with write access and K_{NETI} authorization) that needs to be a client of this remote file system, run the command:

```
rfs-setup --gang-client client_IP_address EEEE-SSSS-NNNN keyhash
```

In this command:

- `client_IP_address` is the IP address of the client
- `EEEE-SSSS-NNNN` is the ESN of the unit
- `keyhash` is the hash of the K_{NETI} key on the unit.
- For every unauthenticated client (with write access but without K_{NETI} authorization), run the command:

```
rfs-setup --gang-client --write-noauth client_IP_address
```

Note: The `--write-noauth` option should be used only if you believe your network is secure. This option allows the client you are configuring to access the RFS without K_{NETI} authorization.

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 κ_{NETI} for authorization (which is generated when the HSM 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 κ_{NETI} 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.

Note: 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.

Note: For more information about the `rfs-sync` utility, see [rfs-sync on page 71](#).

3. 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_address
remote_esn=keyhash=0000000000000000000000000000000000000000000000000000000000000000
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_address
remote_esn=keyhash=0000000000000000000000000000000000000000000000000000000000000000
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_{NETI} 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 HSM to use for KNETI authorisation. The default is HSM 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 [Environment variables on page 218](#).

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. `/etc/nfast.conf` is executed by the start-up scripts of nShield services as the root user. This file should only contain shell commands that can safely be run in this context. `/etc/nfast.conf` should be created with access permissions that allow only the root user to write to the file.

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

Logging and debugging

The nShield Connect and applications that use it generate log files. You can view the logs using the unit front panel. Application log messages are handled on the client.

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 [Logging, debugging, and diagnostics on page 222](#).

Some components of the Security World Software generate separate debugging information which you can manage differently. Debugging information for applications is handled on the client. If you are setting up the unit to develop software that uses it, you should configure debugging before commencing software development.

Configuring Java support for KeySafe

To use KeySafe, follow the instructions in [Using KeySafe on page 201](#).

Routing

If you have configured only one network interface, you do not need to configure a static route for the unit, although you can do so if you wish. If you have configured a second network interface, you can choose to configure a static route.

If the unit is to connect to a remote host or network that is unreachable through the default gateway, you must set up extra static routes in the system routing table.

To set up the Ethernet interfaces (IPv4 and IPv6), see the *nShield Connect Installation Guide*.

After you have defined static routes, you should test them as described in [Testing routes on page 73](#).

Note: If you define, edit, or delete a route, you must reboot the unit before the route can be used and the routing table is updated.

Testing routes

When you have set up or edited a route, you should test the route.

Testing a route between the unit and the client

When you have installed the unit in its final location, you should test the connection between the unit and the client. You can do this from the client, as described in this section, or by using the **Ping remote host** option on the unit. To do this from the unit, select **System > System configuration > Network config > Ping remote host**.

You can also use the method described in this section to test the route between the client and a remote computer.

To test the route from the client to the unit, issue a `ping` command from the client for the IP address that you specified for the unit. (The format of the command and results may vary according to the platform that you are using.)

```
ping xxx.xxx.xxx.xxx
```

If the `ping` operation is successful, a message similar to the following is displayed:

```
Pinging xxx.xxx.xxx.xxx with 32 bytes of data
Reply from xxx.xxx.xxx.xxx: bytes=32 time=10ms TTL=125
Reply from xxx.xxx.xxx.xxx: bytes=32 time=10ms TTL=125
Reply from xxx.xxx.xxx.xxx:10ms TTL=125
Reply from xxx.xxx.xxx.xxx:10ms TTL=125
```

Testing a route between the unit and a remote host

When you have defined or edited a route from the unit to a remote computer, you should test it. To do this you can issue a `ping` command from the unit to the IP address of the host.

You can also use this method to test the connection between the unit and the client.

To test a route from the unit to a host:

1. Select **System > System configuration > Network config > Ping remote host**. The following screen appears:

```
    Ping remote host
Select IP address:
    0. 0. 0. 0
RESET          FINISH
```

2. Enter the IP address of the remote host.
3. Press **FINISH** to issue the ping. The following message appears:

```
Please wait, running ping
```

4. After a short wait, a display similar to the following should appear, showing that the unit has managed to communicate with the host:

```

Ping xxx.xxx.xxx.xxx:
#0: rtt=0.0503 ms
#1: rtt=0.0503 ms
#2: rtt=0.0503 ms
#3: rtt=0.0503 ms
4 of 4 packets back.
  min avg max SD
  0.29 0.36 0.50 0.09 ms

```

If not all of the information is visible onscreen, use the touch wheel to scroll up and down the page.

5. Press the left-hand navigation button to return to the **Network config** menu.

Tracing network routes

You can trace network routes from the unit and from clients.

Tracing the route from the unit

You can trace the route taken from the unit to a remote computer. You can also use this method to trace the route from the unit to the client.

1. Select **System > System configuration > Network config > Trace route to host**. The following screen appears:

```

Trace route
Select IP address:
  0. 0. 0. 0
CANCEL          FINISH

```

2. Press the right-hand navigation button to issue the **traceroute**. The following message appears:

```
Please wait, running traceroute.
```

3. After a short wait, a display similar to the following should appear, showing the IP addresses encountered along the route:

```

1: xxx.xx.xxx.x
   0.40 ms
2: *
3: xx.xxx.xx.xxx
   3.1 ms
4: xxx.xx.xxx.xxx
   2.4 ms
BACK

```

If not all of the information is visible onscreen, use the touch wheel to scroll up and down the

page.

4. Press the left-hand navigation button to return to the **Network config** menu.

Tracing the route from the client

You can trace the route from the client to the unit or (if the client is connected to the public Internet) to a remote computer.

To trace the route from the client to the unit, issue a **tracert** command from the client for the IP address of the unit. (The format of the command, and results, may vary depending upon the platform that you are using.)

```
tracert xxx.xxx.xxx.xxx
```

After a short wait, a display similar to the following should appear, showing the IP addresses encountered along the route:

```
Tracing route to modulename (xxx.xxx.xxx.xxx) over a maximum of 30 hops:
1 xxx.xxx.xxx.xxx (xxx.xxx.xxx.xxx) 1.457 ms 0.513 ms 0.311 ms
2 xxx.xxx.xxx.xxx (xxx.xxx.xxx.xxx) 0.773 ms 0.523 ms 1.615 ms
```

Displaying the routing table

You can view details of all the IP addresses for which the internal security module has a route stored. The routing table includes entries for static routes (which are stored permanently) and local hosts to which the module has set up temporary routing entries.

To view the routing table:

1. Select **System > System configuration > Network config > Show routing table**. A display similar to the following appears:

```
Dest Gateway Flg
1 xxx.xxx.xxx.xxx
   xxx.xxx.xxx.xxx UG
2 xxx.xx.xx.x
   xxx.x.x.xxx UG
BACK
```

If not all of the information is visible on the unit display screen, use the touch wheel to scroll up and down the page.

2. Press the left-hand navigation button to return to the **Network config** menu.

Configuring an nShield Connect for Remote Administration

The following applies to either the initial nShield Connect that is used to create a Security World, or a new HSM that is being added.

Configuring Dynamic Slots

To use Remote Administration, you need to configure the number of Dynamic Slots that are available on an HSM. The maximum number of Dynamic Slots that an HSM can have is 16.

Note: The default number of slots is 0. This disables Remote Administration on the relevant HSM.

1. Do one of the following:
 - a. Use the `dynamic_slots` section in the module configuration file to define the number of Dynamic Slots for an HSM and push the updated configuration file to the nShield Connect. See [Appendix E: nShield Connect and client configuration files on page 250](#) for more about module configuration files, [dynamic_slots on page 263](#) for more about the `dynamic_slots` section and [Configuring the nShield Connect with configuration files on page 68](#) for more about editing the module configuration file.
or:
 - c. Use the front panel controls to navigate to **Security World mgmt > Set up dynamic slots > Dynamic Slots** and follow the instructions on the screen.
2. Clear the HSM for the changes to take effect.
For example, run the `nopcLEARfail` command:

```
nopcLEARfail --clear --all
```

You can check that the HSM has Dynamic Slots by:

- Running the command:

```
slotinfo -m 1
```

For example, if four Dynamic Slots have been configured, the output from this command includes the lines:

Slot	Type	Token	IC	Flags	Details
#0	Smartcard	-	1	A	
#1	Software Tkn	-	0		
#2	smartcard	-	0	AD	
#3	smartcard	-	0	AD	
#4	smartcard	-	0	AD	
#5	smartcard	-	0	AD	

- The **D** in the **Flags** column indicates that slots #2 to #5 are Dynamic Slots.
Note: Depending upon your system configuration, it can take up to 30 seconds for the Dynamic Slots to appear.
or:
 - Using the front panel controls to navigate to **Security World mgmt > Display World**.

Using Remote Administration with applications requiring cards in slot 0

If you want to use Remote Administration, but have an application that expects cards to be presented in slot 0, you must configure a slot mapping for each affected HSM.

Do one of the the following:

- Use The `s1ot_mapping` section in the module configuration file to define a Dynamic Slot to exchange with slot 0 for an HSM and push the updated configuration file to the nShield Connect. See [Appendix E: nShield Connect and client configuration files on page 250](#) for more about module configuration files, [slot_mapping on page 263](#) for more about the `s1ot_mapping` section and [Configuring the nShield Connect with configuration files on page 68](#) for more about editing the module configuration file.
or:
- Use the front panel controls to navigate to **Security World mgmt > Set up dynamic slots > Slot mapping** and follow the instructions on the screen.

You can check the mapping by:

- Running the command:

```
slotinfo -m 1
```

For example, if dynamic slot #2 has been mapped to slot #0, the output from this command includes the lines:

Slot	Type	Token	IC	Flags	Details
#0	Smartcard	-	1	AD	
#1	Software Tkn	-	0		
#2	smartcard	-	0	A	

- The **D** in the **Flags** column indicates that slot #0 is now a Dynamic Slot
or:
- Using the front panel controls to navigate to **Security World mgmt > Display World**.

Adjusting card removal detection timers to account for network characteristics

Depending upon the characteristics of the network between nShield Remote Administration Clients and HSMs, you may need to adjust the timers that determine how long the system waits for a response, before it regards a card as having been removed. This enables you to balance the assured card removal detection time and network traffic.

Do the following:

- Use The `dynamic_slot_timeouts` section in the module configuration file to define the round trip (HSM to smartcard and back) time limit (the default is 10 seconds), and the card removal detection timeout (the default is 30 seconds). Push the updated configuration file to the nShield Connect.

See [dynamic_slot_timeouts on page 263](#) and [Configuring the nShield Connect with configuration files on page 68](#) for more information.

Using nethsmadmin to reboot an nShield Connect

An nShield Connect can be rebooted using the `nethsmadmin` command-line utility. Run the command:

```
nethsmadmin --module=>MODULE --reboot
```

In this command:

`--module=>MODULE`

Specifies the HSM to use, by its `ModuleID` (default = 1).

Enabling and disabling remote reboot

You can enable or disable the ability to remotely reboot an nShield Connect with the `nethsmadmin` utility using:

- The module configuration file
- The front panel of the nShield Connect

Enabling and disabling remote reboot using the module configuration file

See [Configuring the nShield Connect with configuration files on page 68](#) for more about editing the module configuration file.

Do the following:

- To enable remote reboot, locate the `server_settings` section of the module configuration file, and set the `enable_remote_reboot` field to `yes`.
- To disable remote reboot, set the `enable_remote_reboot` field to `no`. See [server_settings on page 256](#) for more about the `enable_remote_reboot` field.

Enabling and disabling remote reboot using the front panel of the nShield Connect

Do the following:

- To enable remote reboot, navigate to **System > System configuration > Remote config options > Remote Reboot** and set to `on`.
- To disable remote reboot, set **Remote Reboot** to `off`.

Once you have enabled remote reboot, you can reboot an nShield Connect from a computer using the `nethsmadmin` command, without accessing the unit itself.

Enabling and disabling remote mode changes

You can enable or disable the ability to make remote mode changes with the `nopclearfail` utility using:

- The module configuration file
- The front panel of the nShield Connect

Enabling and disabling remote mode changes using the module configuration file

See [Configuring the nShield Connect with configuration files on page 68](#) for more about editing the module configuration file.

Do the following:

- To enable mode change using `nopclearfail`, locate the `server_settings` section of the module configuration file, and set the `enable_remote_mode` field to `yes`.
- To disable mode change using `nopclearfail`, set the `enable_remote_mode` field to `no`. See [server_settings on page 256](#) for more about the `enable_remote_mode` field.

Enabling and disabling remote mode changes using the front panel of the nShield Connect

Do the following:

- To enable remote mode change, navigate to **System > System configuration > Remote config options > Remote mode changes** and set to `on`.
- To disable remote mode change, set **Remote mode changes** to `off`.

Once you have enabled remote mode change, you can change the mode of an nShield Connect from a computer using the `nopclearfail` command, without accessing the unit itself.

Enabling and disabling remote upgrade

You can enable or disable the ability to remotely upgrade an nShield Connect with the `netnsmadmin` utility using:

- The module configuration file
- The front panel of the nShield Connect

Enabling and disabling remote upgrade using the module configuration file

See [Configuring the nShield Connect with configuration files on page 68](#) for more about editing the module configuration file.

Do the following:

- To enable remote upgrade, locate the `server_settings` section of the module configuration file, and set the `enable_remote_upgrade` field to `yes`.
- To disable remote upgrade, set the `enable_remote_upgrade` field to `no`. See [server_settings on page 256](#) for more about the `enable_remote_upgrade` field.

Enabling and disabling remote upgrade using the front panel of the nShield Connect

Do the following:

- To enable remote upgrade, navigate to **System > System configuration > Remote config options > Remote Upgrade** and set to `on`.

- To disable remote upgrade, set **Remote Upgrade** to **off**.

Once you have enabled remote upgrade, you can upgrade an nShield Connect from a computer using the `nethsmadmin` command, without accessing the unit itself.

Authorized Card List

The use of nShield Remote Administration Cards is controlled by an Authorized Card List. If the serial number of a card does not appear in the Authorized Card List, it is not recognized by the system and cannot be used. The list only applies to Remote Administration cards and is used when a card is inserted:

- In the local slot of an HSM
- In a card reader that is associated with a dynamic slot of the HSM, through the nShield Remote Administration Client

By default, the Authorized Card List is empty following software installation. The serial numbers of Remote Administration Cards must be added to the list before they can be used.

The Authorized Card List is a text file `opt/nfast/kmdata/config/cardlist` on the RFS and each client computer. The file is read from the RFS by associated nShield Connects as and when required for front panel operations. The list applies to all Connects associated with the RFS, regardless of the Security World to which a Connect may belong, including when creating a Security World from the front panel. For client initiated card operations the Authorized Card List file on that client computer is used. The RFS and client copies of the Authorized Card List have to be kept in step manually.

Adding cards to the Authorized Card List

Add the serial numbers (16 digits with no separators) of all Remote Administrator Cards you intend to use to the Authorized Card List, with a standard text editor. The serial numbers are printed on the smart cards and are reported by using `s1otinfo -m1 -s0` when the card is in a slot, where *1* is the number of the HSM and *0* is the number of the slot.

Note: There is an option to allow any Remote Administration Card to be used, by including a wildcard (*) in the Authorized Card List. nCipher recommends that you do not use this option, except under controlled circumstances, as it effectively disables the Authorized Card List control.

Enabling optional features on the unit

nShield Connect support a range of optional features. Optional features must be enabled with a certificate that you order from nCipher. You can order the features when you purchase a unit, or you can obtain the certificate at a later date and use the front panel of the unit to enable the features.

For more information about:

- Ordering optional features, see [Ordering additional features on page 86](#)
- Feature-enabling procedures, see [Enabling features on page 86](#).

The unit 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 unit. 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 unit. A dynamic feature must be enabled again if the unit is reinitialized.

Note: If you are enabling the Remote Operator feature, you must enable it on the unit that is to be used as the unattended unit. For information about Remote Operator, see [Remote Operator on page 187](#).

Available optional features

This section lists the features that can be added to the unit. For details of all available features, contact Sales.

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.

nCipher 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. nCipher uses an activator smart card to enable this feature. Refer to [Enabling features with a smart card on page 87](#) 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 87](#). Contact Sales if you require an EC activation.

nCipher 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 ³	
	Named curves ² Custom curves ^{1, 5}	Named curves	Custom curves ^{1, 5}

HSM module type	Elliptic Curve support		Elliptic Curve offload acceleration ³	
nShield PCIe 500 and 6000; nShield Connect 500, 1500 and 6000 ² .	Yes.	Yes.	No.	No.
nShield PCIe 500+ and 6000+, nShield Connect 6000+ ² .	Yes.	Yes.	Yes, Prime curves are accelerated ^{2, 4} .	Yes.

¹ Accessed via nCore and PKCS #11 APIs.

² Both Prime and Binary named curves are supported. Refer to [Named Curves on page 83](#), below, which lists the most commonly supported elliptic curves.

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

⁴ Binary curves are supported, but are not hardware offload accelerated.

⁵ Includes support for Brainpool curves. See [Brainpool curves on page 84](#), 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) ¹ .	Microsoft CNG, PKCS#11, Java Cryptographic Engine (JCE) ¹ .	Microsoft CNG, PKCS#11, Java Cryptographic Engine (JCE) ¹ .

¹ Java elliptic curve functionality is fully supported by the nCipher security provider, nCipherKM. There is also the option to use the Sun/IBM PKCS #11 Provider with nCipherKM configured to use the nCipher PKCS#11 library.

Named Curves

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

Curve	Offload acceleration: nShield 500+, nShield 6000+ and nShield Connect 6000+
NISTP192	Yes
NISTP224	Yes
NISTP256	Yes

Curve	Offload acceleration: nShield 500+, nShield 6000+ and nShield Connect 6000+
NISTP384	Yes
NISTP521	Yes
NISTB163	—
NISTB233	—
NISTB283	—
NISTB409	—
NISTB571	—
NISTK163	—
NISTK233	—
NISTK283	—
NISTK409	—
NISTK571	—
ANSIB163v1	—
ANSIB191v1	—
SECP160r1	Yes

Custom curves

nCipher 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 83](#), 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
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 169](#)
 - Symmetric and asymmetric algorithms: [Cryptographic algorithms on page 267](#)
 - Using `generatekey` options and parameters to generate ECDH and ECDSA keys: [Key generation options and parameters on page 270](#)

Note: Java elliptic curve functionality is fully supported by the nCipher security provider, nCipherKM. There is also the option to use the Sun/IBM PKCS #11 Provider with nCipherKM configured to use the nCipher PKCS#11 library.

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 nCipher 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 nCipher 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 nCipher 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 [Remote Operator on page 187](#).

For v12 and later, nCipher recommends that you use Remote Administration, which is more flexible than the Remote Operator functionality.

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.

Client licenses

You can purchase additional client licenses that allow you to run multiple clients for the unit. Three clients are always enabled on each unit.

Ordering additional features

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

- If possible, make a note of the unit serial number. This can be found on the base of the unit.
- Make a note of the Electronic Serial Number of the unit. You can find this from the front panel menu, by going to **HSM > HSM information > Display details**.

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.

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

- nCipher e-mails you the Feature Enabling Certificate.
- nCipher 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 email your nearest Sales representative using the contact details from this guide, or the nCipher web site (<https://www.ncipher.com>).

Enabling features

You must enable features from the front panel of the unit. You cannot enable them directly from the client.

Note: When enabling static feature(s) from the front panel, either using a card or a file, the module is cleared without warning. This will cause the HSM to drop or restart any SEE machine, and lose all the application keys that were loaded. In some cases, applications may need to be restarted.

Viewing enabled features

To see which (if any) features have already been enabled on the nShield Connect, from the main menu select **HSM > HSM feature enable > View current state**.

To print this list to a file on the unit, select **HSM > HSM feature enable > Write state to file**. The resulting file is transferred when the unit configuration is pushed to the remote file system. You can find it in `/opt/nfast/kmdata/hsm-ESN/features/fet.txt`.

Enabling features with a smart card

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

1. Insert the Feature Enabling smart card into the unit's slot.
2. From the front panel, select **HSM > HSM feature enable > Read FEM from card**.

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 87](#).

Enabling features without a smart card

You can also provide the Feature Enabling Certificate information supplied by nCipher from a file.

To enable a feature without a smart card:

1. Put the file that contains the feature enabling certificate in `/opt/nfast/kmdata/hsm-ESN/features` on the remote file system. In this path, **ESN** is the ESN of the module.
You can give the file any name that you wish. You must enter the file name on the unit's front panel, so you may prefer to use a short name.
2. From the front panel, select **HSM > HSM feature enable > Read from a file**.
3. Specify the name of the file that contains the certificate.

If the feature is enabled successfully, a message confirms this.

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.

Electronic Serial Number (ESN)

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 installation media supplied with your new module for information about specific compatibility issues, new features, and bug fixes.

1. Install the module hardware. Refer to the *Installation Guide* for information on installing nCipher hardware.
2. Run the script `/opt/nfast/sbin/install`.
3. Add the module to the Security World. Refer to *Adding or restoring an HSM to the Security World on page 111*.

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 client hardserver

If necessary, you can stop the hardserver on the client, and where applicable the Remote Administration Service, by running the following command:

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

Similarly, you can restart the hardserver on the client, and where applicable the Remote Administration Service, by running the following command:

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

Resetting and testing the nShield Connect

Default configuration

To reset the unit to its default configuration, select **System > System configuration > Default config** and confirm that you want to set the default configuration.

This removes the configuration of the module but does not change its k_{NETI} .

Factory state

To reset the unit to its original (factory) state, select **Factory state** from the main menu and confirm that you want to return the unit to its factory state.

This gives a new k_{NETI} to the unit, which means that you must update the keyhash field of the unit's entry in the `nethsm_imports` section of the configuration file of all the clients that use it.

For more information about:

- The contents of the configuration files, see *nShield Connect and client configuration files* on page 250
- Configuring a new remote file system for the unit, see *Configuring the remote file system (RFS)* on page 57.

Testing the installation

To test the installation and configuration, follow these steps:

1. Log in on the client computer as a regular user, and open a command window.
2. Run the command:

```
opt/nfast/bin/enquiry
```

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

```
Module ##:
enquiry reply flags  none
enquiry reply level Six
serial number       #####-####-####
mode                 operational
version              #.#.#
speed index          #####
rec. queue           ##.##
...
version serial      ##
connection status    OK
connection info      esn = #####-####-####; addr = ####/###.##.##.###/####; ku hash =
xxxxx, mech = Any; time-limit = 24h; data-limit = 8MB
image version        #.#.#####
max exported modules 3
rec. LongJobs queue ##
SEE machine type     PowerPCSF
supported KML types  DSAP1024s160 DSAP3072s256
using impath kx grp  DHPrime3072
hardware status      OK
```

If the mode is **operational**, the unit is installed correctly.

If the **enquiry** command says that the unit is not found:

- a. Restart the client computer.
- b. Re-run the **enquiry** command.

Chapter 7: 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 HSM to manage keys.

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

- The Installation Guide for more about installing the module and software.
- [Client configuration on page 54](#)

You create a Security World with a single HSM. 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 an HSM to the Security World on page 111](#).

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 54](#).



All Security Worlds rely on you using the security features of your operating system to control the users who can access the Security World and, for example, write data to the host.

Creating a Security World

You can use the following to create Security Worlds:

- The unit front panel controls
See [Creating a Security World using the nShield Connect front panel on page 97](#).
- The `new-world` command line utility
See [Creating a Security World using new-world on page 100](#).
- KeySafe
See [Creating a Security World using KeySafe on page 106](#).

You can use the nShield Connect front panel controls, KeySafe, and the `new-world` utility to create a Security World for which you can specify different `k` values for the ACS.

The creation process

When you create a Security World:

- The HSM is erased
- A new HSM key for this Security World is generated
- A new ACS to protect this HSM key is created

- The Security World information is stored within the file system of the nShield Connect operating system and on the RFS
 - 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 Operator Cards, except for nShield Remote Administration Cards, while the previous Security World still exists. See [Erasing cards and softcards on page 134](#).

Security World files

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

For more information about the remote file system, see:

- [Remote file system \(RFS\) on page 55](#)
- [Configuring the remote file system \(RFS\) on page 57](#).

Other nCipher modules can also use a Security World created on an nShield Connect using client cooperation. For more information, see [Setting up client cooperation on page 69](#).

Location of Security World files

If a Security World operation is done on the module, the files are created or updated on the module and automatically copied to the directory specified by the environment variable `NFAST_KMLOCAL` on the remote file system. By default, this is `/opt/nfast/kmdata/local`.

If the Security World operation is done on a client machine (for example, Operator Card Set and key operations), files are created or updated in the `/opt/nfast/kmdata/local`, or the directory specified by the `NFAST_KMLOCAL` environment variable, on the client machine only.

The `/opt/nfast/kmdata/local` directory on the remote file system is updated automatically when an operation is done on the module.

If you want to make cards or keys which are normally created from the client available from the module's front panel, we recommend that you use client co-operation to automate the copying of files to the module. For information about configuring client co-operation, see [Setting up client cooperation on page 69](#).

If you do not use client cooperation, you must manually copy the appropriate card and key files from the client or host on which the card set or key was created to the `/opt/nfast/kmdata/local`'s remote file system. These files must then be updated on the module by selecting **Security World mgmt > RFS operations > Update World files** from the main menu.

To be able to create Operator Cards or keys, the user on the client must have 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 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	world (for each module in the Security World) module_ESN
Load a Security World	creates or modifies	(for each module in the Security World) module_ESN
Replace an ACS	modifies	world
Create an OCS	creates	card_HASH cards_HASH_NUMBER
Create a softcard	creates	softcard_HASH
Generate a key	creates	key_APPNAME_IDENT
Recover a key	modifies	key_APPNAME (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 client 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 CodeSafe
- 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 (SP800-131A 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 SP800-131A, Security World generation and key generation operations will take longer in SP800-131A compliant Security Worlds.

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

Note: The `--disablepkcs1pad` option will only work on SP800-131A Security Worlds.

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. The `enquiry` utility displays information related to an HSM.

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 unit on page 81](#).

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 34](#). For details about performing OCS and softcard replacement operations, see [Replacing Operator Card Sets on page 145](#) and [Replacing the Administrator Card Set on page 154](#).

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 phrase on page 144](#).

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 nShield 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 `/opt/nfast/kmdata/local` directory is not overwritten but renamed `/opt/nfast/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 `/opt/nfast/kmdata/local_N` directory from the old Security World, you must delete it manually.

Creating a Security World using the nShield Connect front panel

Note: When initiated from the Connect front panel, while a Security World is being created the Connect disconnects itself from the network to ensure that the operation is not interrupted. This means that the Remote Administration feature cannot be used to present cards from a remote location when creating a Security World from the front panel.

Before you start

Before you start to create a Security World:

- The directory `/opt/nfast/kmdata/local` on the remote file system must exist and be empty.
- Before configuring the Security World, you should know:
 - The security policy for the HSM
 - 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 94](#).
- You must have enough smart cards to form the Security World's card sets.

To create a Security World from the unit front panel:

1. From the main menu, select **Security World mgmt > Module initialization > New Security World**.

2. Enter the default quorum for the ACS. This consists of:
 - a. The maximum number of cards from the ACS required by default for an operation. This number must be less than or equal to the total number of cards in the set.
 - b. The total number of cards to be used in the ACS. This must be a value in the range 1 – 64.
Note: 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 cannot replace such an ACS if even a single card is lost or damaged.
3. If you answer the question **Specify all quorums?** by selecting:
 - a. **no** all operations and features (with the exception of passphrase recovery) will be enabled and require the maximum number of cards.
 - b. **yes** you can specify which operations and features you want to enable (including passphrase recovery) and what the required number of cards for each of these will be.
4. Select the Cipher suite for the Security World—that is, whether the Security World key is to be an AES original or AES (SP800-131A).
5. Specify whether the Security World will conform to FIPS 140-2 requirements for roles and services at level 3. If you do not specify this, the Security World complies with FIPS 140-2 requirements for level 2.



The current release of the nCipher PKCS #11 library works with a FIPS 140-2 level 3 compliant Security World. However, if you intend to use the Security World with the PKCS #11 application Sun Java Enterprise System (JES), do not specify this flag.

6. If you chose to disable individual features or require a lower number of cards required for an operation, specify these parameters now. You can select a different number of Administrator Cards (K) to be required for each operation. You can also disable recovery and replacement operations and choose to use K_{NSO} to authorize SEE (Secure Execution Engine) operations. The options for which you can specify a separate value of K are as follows:

Operation	Action allowed on HSM
Module reprogramming	Initializing an HSM into a Security World. You must specify a value of K for this operation.
OCS/softcard replacement	Recovery of keys from backup files when replacing an OCS. You can disable this operation if you are using the nShield Connect. See OCS and softcard replacement on page 95 .
Pass phrase replacement	Replacement of pass phrases from backup files when recovering an OCS. You can disable this operation. See Pass phrase replacement on page 96 .
NVRAM access	Reading from and writing to the NVRAM. You can choose to authorize this operation with K_{NSO} . See Nonvolatile memory (NVRAM) options on page 96 .
RTC access	Updating the real time clock. You can choose to authorize this operation with K_{NSO} . See Real-time clock (RTC) options on page 97 .
SEE debugging	Viewing full SEE debug information. You can specify a value of K for this operation, all it for all users or authorize it with K_{NSO} . See SEE debugging on page 97 .
FTO	Use of an Foreign Token Open (FTO) Delegate Key (ISO Smart Card Support). You can specify a value of K for this operation or authorize it with K_{NSO} .

7. Specify whether the HSM is a valid target for remote shares (that is, whether it can import slots); see [Remote Operator on page 187](#).
8. Format a card for the ACS as follows:
 - a. Insert a card for the ACS and confirm that you want to use it.
 - b. If the card is not blank, choose whether to overwrite it or to use a different card.
 - c. Choose whether to specify a pass phrase for the card. If you choose to specify a pass phrase:
 - i. Enter the pass phrase.
 - ii. Enter the pass phrase again to confirm it.
If the two pass phrases do not match, you must enter the correct pass phrase twice.
 - d. When prompted, remove the card.
9. Repeat the previous step to format additional cards for the ACS, setting their pass phrases as described, until the ACS is complete.
Each prompt screen shows how many cards are required and how many have been used.
10. At completion, a message confirms that the Security World has been created.

Creating a Security World using new-world or KeySafe

Before you start

Before you start to create a Security World:

- The HSM must be in pre-initialization mode. See [Appendix H: Checking and changing the mode on an nShield Connect on page 277](#) for more about changing the mode.
- You must be logged in to the computer that is running the RFS. The RFS should be a privileged client that has the client tools installed. For more information, see [server_settings on page 256](#).
- 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 HSM
 - The number and quorum of Administrator Cards and Operator Cards to be usedTo help you decide on the Security World you require, see [Security World options on page 94](#).
- You must have enough smart cards to form the Security World's card set.

When you have finished creating a Security World, you must restart the HSM in operational mode.

Using `nethsmadmin` to copy a Security World to an nShield Connect and check the current version

If a Security World is created using `new-world`, `KeySafe`, the CSP wizard, or the CNG wizard, the `nethsmadmin` command-line utility enables you to copy the resultant files to an nShield Connect. Run the command:

```
nethsmadmin --module=MODULE --update-world
```

`nethsmadmin` can also be used to check if the Security World files have been copied to the nShield Connect. Run the command:

```
nethsmadmin --module=MODULE --check-world
```

In these commands:

```
--module=MODULE
```

Specifies the HSM to use, by its `ModuleID` (default = 1).

Creating a Security World using `new-world`

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

Note: With Security World Software v11.72 and later, pass phrases are limited to a maximum length of 254 characters, when using `new-world`. See [Maximum pass phrase length on page 30](#) for more information.



Running the `new-world` command-line utility

Open a command prompt 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] [--pp-
min=LENGTH] FEATURES [--disablepkcs1pad]
```

In this command:

Option	Description
<code>--initialize</code>	<p>This option tells <code>new-world</code> to create a new Security World, replacing any existing <code>/opt/nfast/kmdata/local</code> directory.</p> <p>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 <code>/opt/nfast/kmdata/local</code> directory in which these reside as <code>/opt/nfast/kmdata/localN</code> (where <code>N</code> is an integer assigned depending on how many Security Worlds have been previously saved during overwrites).</p>
<code>--factory</code>	This option tells <code>new-world</code> to erase an HSM, restoring it to factory state.
<code>--no-remoteshare-cert</code>	This option tells <code>new-world</code> not to make the HSM a target for remote shares.
<code>--strict-fips-140-2-level-3</code>	<p>This options tells <code>new-world</code> 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, <code>new-world</code> creates a Security World that complies with FIPS 140-2 requirements for level 2.</p> <p>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.</p>
<code>--no-strict-rsa-keygen</code>	If you are using the <code>--strict-fips-140-2-level-3</code> flag to create a FIPS 140-2 level 3 Security World, you can use the <code>--no-strict-rsa-keygen</code> flag to <i>disable</i> UseStrongPrimes. Otherwise it is enabled by default.
<code>--strict-rsa-keygen</code>	If you are creating a FIPS 140-2 level 2 Security World, you can use the <code>--strict-rsa-keygen</code> flag to <i>enable</i> UseStrongPrimes. Otherwise, it is disabled by default.
	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.

Option	Description
--no-recovery	<p>This option tells <code>new-world</code> to disable the ability to recovery or replace OCSs and softcard (which is otherwise enabled by default). This is equivalent to setting <code>!r</code>, where the <code>!</code> operator instructs the system to turn off the specified feature (<code>r</code>).</p> <p>By default, <code>new-world</code> creates key recovery and replacement data that is protected by the cryptographic keys on the ACS. This option does not give nCipher 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> <p> We recommend that you do not disable the recovery and replacement option.</p> <p>If you set the <code>--no-recovery</code> 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>
--cipher-suite =CIPHER-SUITE	<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"> • DLf1024s160mDES3, which uses a Triple DES key. • DLf1024s160mRijndae1, which uses an AES key. • DLf3072s256mRijndae1, which uses an AES key to create a Security World compliant with SP800-131A. <p>Note: The <code>--disablepkcs1pad</code> option will only work on SP800-131A Security Worlds.</p> <p>Note: The default cipher suite is <code>DLf3072s256mRijndae1</code>. For software versions before Security World Software v12.40, the default cipher suite is <code>DLf1024s160mDES3</code>.</p>
--nso- timeout=TIMEOUT	<p>This option allows you to specify the time-out (<i>TIMEOUT</i>) for new Security Worlds. By default, an integer given for <i>TIMEOUT</i> is interpreted in seconds, but you can supply values for <i>TIMEOUT</i> in the form <i>Ns</i>, <i>Nh</i>, or <i>Nd</i> where <i>N</i> is an integer and <i>s</i> specifies second, <i>h</i> specifies hours, and <i>d</i> specifies days.</p>
--module=MODULE	<p>This option specifies the module to use (by its ModuleID). If you have multiple modules, <code>new-world</code> initializes them all together.</p>

Option	Description
<p><code>--acs-quorum=K/N</code></p>	<p>In this option, <i>K</i> specifies the minimum number of smart cards needed from the ACS to authorize a feature. You can specify lower <i>K</i> values for a particular feature. All the <i>K</i> values must be less than or equal to the total number of cards in the set. If a value for <i>K</i> is not specified, <code>new-world</code> creates an ACS that requires a single card for authorization.</p> <p>Note: 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 <i>K</i>=1.</p> <p><i>N</i> 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, <code>new-world</code> creates an ACS that contains a single card.</p> <p>Note: 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>
<p><code>--reduced-features</code></p>	<p>This option instructs <code>new-world</code> 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>
<p><code>--disablepkcs1pad</code></p>	<p>This option disables the use of PKCS#1 v1.5 padding. All attempts to use PKCS#1 v1.5 padding for encryption or decryption operations will be rejected.</p> <p>This option can only be used with a <i>Df3072s256mRijndael</i> Security World. The other two Security World types cannot be used with this option and attempts to create them will fail.</p> <p>PKCS#1 v1.5 signature operations are not affected.</p> <p>PSS and OAEP are not affected.</p>

Option	Description
<code>--pp-min=LENGTH</code>	<p>This option enables a minimum pass phrase length check for the Administrator Card Set (ACS) the Operator Card Set (OCS) and any associated softcards when you create a Security World. The minimum pass phrase length check is then applied after the Security World is created.</p> <p>When enabled and you attempt to create a card pass phrase with fewer characters than the specified minimum length, the following warning message displays:</p> <p>Warning: short passphrase</p> <p>However, the pass phrase can still be used.</p> <p>Examples:</p> <pre>new-world --initialize --acs-quorum=K/N --cipher-suite=CIPHER-SUITE --pp-min=14</pre> <p>If <code>--pp-min=<length></code> is not used, the minimum pass phrase length is set to the default value (0).</p> <p>Note: The minimum pass phrase length option is not applicable to software versions before Security World Software v11.72.</p>
<code>--pp-strength</code>	<p>This option enables pass phrases to have at least one uppercase, lowercase, number, and symbol.</p> <p>If the <code>--pp-strength</code> argument is omitted, the complexity requirements are not enforced.</p>

new-world command-line utility features

Features for the Security World can be specified using 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 a command-line option unless you have used the `-f` option or specified an HSM 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 HSM, you must set the `dseea11` flag.

The following feature names are available:

Feature name	Description
<code>m</code>	This feature makes it possible to add new HSMs into the Security World. This feature cannot be disabled.
<code>r</code>	This feature enables OCS and softcard replacement; see <i>OCS and softcard replacement on page 95</i> and <i>Replacing Operator Card Sets on page 145</i> .
<code>p</code>	This feature enables pass phrase replacement; see <i>Pass phrase replacement on page 96</i> and <i>Changing card and softcard pass phrase on page 141</i> .
<code>nv</code>	This feature specifies that ACS authorization is needed to enable nonvolatile memory (NVRAM) allocation.
<code>rtc</code>	This feature specifies that ACS authorization is needed to set the real-time clock (RTC); (see <i>Real-time clock (RTC) options on page 97</i>).
<code>dsee</code>	This feature specifies that that ACS authorization is needed to enable SEE World debugging.
<code>dseea11</code>	This feature enables SEE World debugging for <code>a11</code> users.
<code>fto</code>	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 `dseea11` 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 HSM, you must set the `dseea11` flag.



The `dseea11` 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 HSM
- 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-wor1d` 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-wor1d` displays an error and exits.

If the HSM is not in the pre-initialization mode, `new-wor1d` advises you that you must put the HSM in this mode and waits until you have changed the HSM mode before continuing. See [Appendix H: Checking and changing the mode on an nShield Connect on page 277](#) for more about changing the mode.

Note: If the HSM is in the pre-initialization mode, `new-wor1d` prompts you for smart cards and pass phrases as required.

Creating a Security World using KeySafe

To create a Security World using KeySafe:

1. Start KeySafe. (For an introduction to KeySafe and information on starting the software, see [Using KeySafe on page 201](#).)
2. Click the **World** menu button, or select **World** from the **Manage** menu. KeySafe opens the **World Operations** panel.
3. Click the **Initialize Security World** button.
4. 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 `/opt/nfast/kmdata/local` directory in which these reside as `/opt/nfast/kmdata/local N` (where *N* is an integer assigned depending on how many Security Worlds have been previously saved during overwrites)

To overwrite an existing Security World, click **Next** to go to the **Initialize Security World** panel.

5. On the **Initialize Security World** panel, you can configure the following options:
 - a. The total number of administrator cards (N) that you wish to have in the ACS. This must be a value in the range 1 – 64.
 - b. 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
 - c. The Cipher suite for the Security World—that is, whether the Security World key is to be an AES (original) or AES (SP800-131A compliant).

Note: The `--disablepkcs1pad` option will only work on SP800-131A Security Worlds.
 - d. 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.
 - e. Whether or not the Security World is to permit receipt of remote operator card shares.
6. Click **Set Advanced Options** to open the **Initialize Security World Advanced Options** panel.

Note: 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. This allows you to create key recovery and pass phrase replacement data that is protected by the cryptographic keys on the ACS. This does not give nCipher, 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.

7. On the **Initialize Security World Advanced Options** panel you can configure the following options:
- Whether to enable key recovery



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

- The number of administrator cards required to authorize key recovery
- Whether to enable passphrase recovery
- The number of administrator cards required to authorize passphrase recovery
- The number of administrator cards required to load this Security World onto a new HSM
- Whether to generate a foreign token key
- The number of administrator cards required to authorize foreign token key operations

Click **OK** to save your selections and return to the **Initialize Security World** panel, or **Cancel** to return without saving.

8. Click **SEE options** to open the **Initialize Security World - SEE Options** panel.

Note: If you wish to use SEE, you must have ordered and enabled it as described in *Enabling optional features on the unit on page 81*

9. On the **Initialize Security World - SEE Options** panel, you can configure the following options:
- Whether to generate a real-time clock (RTC) authorization key
 - The number of Administrator Cards required to change RTC details

Note: This option is not applicable unless you have purchased and installed the CodeSafe Developer kit

- Whether to generate a nonvolatile memory (NVRAM) authorization key
- The number of Administrator Cards required to change NVRAM details

Note: This option is not applicable unless you have purchased and installed the CodeSafe Developer kit

- The parameters for SEE debugging. These are:

- **Restricted**
- **Generate Authorization Key**
- **No Access Control.**

Note: To use extended debugging from the HSM, you must set SEEDebugging to **No Access Control.**

- The number of Administrator Cards required to change SEE debugging settings

Click **OK** to save these settings and return to the **Initialize Security World** panel, or **Cancel** to return without saving

10. Check that the HSM card reader (slot 0) is empty, and then click **Commit**. This opens the **Create Administrator Card Set** panel.

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

11. Insert a blank card and click the **OK** button.
 - a. If you insert a non-blank card that KeySafe can erase, the **Overwrite card** button is enabled, giving you the option of overwriting that card.
 - b. 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.
 - c. If you insert a blank card, KeySafe prompts you to click **OK** to set a passphrase.

12. Click **OK** to open the **Set Card Protection Passphrase** 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 **List Operator Card Sets** panel) or the `cardpassphrase` command-line utility.

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

- a. Select the **No** option
- b. Click **OK**.

13. 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, a warning message will be displayed and you will be prompted to insert a blank card.

14. After you have created all the Administrator Cards, Keysafe creates a new HSM key. When initialization is complete, KeySafe displays a message: **Security world successfully initialized**. Click **OK** to return to the introduction panel.
15. After you have added on HSM to the Security World, restart the HSM in the operational mode. See [Appendix H: Checking and changing the mode on an nShield Connect on page 277](#) for more about changing the mode.

If you have more than one HSM on this server, you can use the KeySafe **Reprogram Module** option (on the **World Operations** panel) to incorporate a new HSM into your Security World (or to restore an existing HSM after a firmware upgrade), as described in [Adding or restoring an HSM to the Security World on page 111](#). Alternatively, you can also incorporate a new HSM into your Security World by using the `new-world` command-line utility; see [Adding an HSM to a Security World with new-world on page 112](#)

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 154](#).



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 data is stored on the HSM and on the RFS. For more information, see [Security World files on page 92](#).

The HSM 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:

- Select **Security World mgmt> Display World info** from the main menu
- Run the `nfkminfo` command-line utility. See [Displaying information about a Security World with `nfkminfo` on page 110](#).
- Run the `nethsmadmin` command-line utility. See [Using `nethsmadmin` to copy a Security World to an nShield Connect and check the current version on page 100](#).

You can also use Keysafe to view a summarized description of the Security World.

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
<code>-r --repeat</code>	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> .
<code>-p --preload-client-id</code>	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 [nfkm_{info}: information utility on page 230](#).

Adding or restoring an HSM to the Security World

When you have created your Security World, you can add additional HSMs to it. You can restore HSMs that have previously been removed from the same Security World in the same way.

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

- After you upgrade the firmware
- If you replace the HSM.

Note: The additional HSMs can be any nShield HSMs.

To add an HSM to a Security World, you must:

- Have installed the additional HSM hardware, as described in the *Installation Guide*.
- Have a copy of the Security World data on the HSM's remote file system in the Key Management Data directory.
- Possess a sufficient number of cards from the ACS and the appropriate pass phrases.

Adding or restoring an HSM to a Security World:

- Erases the Security World data on the HSM's internal file system
- 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 remote file system
- Uses the secret from the ACS to decrypt the Security World key
- Stores the Security World key in the HSM's nonvolatile memory.

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

Note: It is not possible to program an HSM into two separate Security Worlds simultaneously.

Adding an HSM to a Security World using the nShield Connect front panel

To add an HSM to a Security World:

1. If the HSM already belongs to a Security World, erase it from the Security World to which it belongs, as described in [Erasing a module from a Security World on page 122](#).
2. From the main menu, select **Security World mgmt > Module initialization > Load Security World**.
3. Specify whether the HSM can use the Remote Operator feature import slots. For more information, see [Remote Operator on page 187](#).
4. At the prompt, insert an Administrator Card, and enter its pass phrase if required.
5. Continue to insert Administrator Cards when prompted until you have inserted the number required to authorize HSM reprogramming.

Adding an HSM 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 an HSM to an existing Security World (in the Key Management Data directory). If you have multiple HSMs available, you can use the `-m|--module=MODULE` option to specify an HSM. If you do not specify an HSM `new-world` adds all available HSMs to the Security World.
- **-s|--no-remoteshare-cert**
These options tell `new-world` not to make the HSM a target for remote shares.
- **-m|--module=MODULE**
This option specifies the HSM to use (by its `ModuleID`). If you have multiple HSMs and do not specify an HSM, `new-world` adds all available HSMs 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 HSM into a new Security World, run `new-world` with the `-i` option. If the HSM is not in the pre-initialization state, `new-world` displays an error and exits. See [Appendix H: Checking and changing the mode on an nShield Connect on page 277](#) for more about changing the mode.

If the HSM 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 HSM, restart the HSM in the operational state. See [Appendix H: Checking and changing the mode on an nShield Connect on page 277](#) for more about changing the 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 HSM is reset to the factory state. The HSM does not form part of the Security World unless you run `new-world` again.

Transferring keys between Security Worlds



You must enter Administrator Cards in the client computer to transfer keys between Security Worlds. If your security policy does not permit this, do not attempt to carry out this procedure.

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 SP800-131A Security World, use the `migrate-world` command-line utility instead; see [Migrating keys to an SP800-131A Security World on page 117](#).

Note: The `--disablepkcs1pad` option will only work on SP800-131A Security Worlds.

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 92](#).
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 FIPS 140-2 level 2 Security World. If both your source and destination security are compliant FIPS 140-2 level 3, you can create a temporary, dummy FIPS 140-2 level 2 Security World 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 (<code>key-xfer-im</code>)
Level 2	Level 2	source	destination	-
Level 2	Level 2	destination	source	-
Level 2	Level 3	source	destination	<code>--export-delete</code>
Level 3	Level 2	destination	source	<code>--export-add</code>
Level 3	Level 3	Dummy	source and destination	-

Note: In this table, Level 3 means a FIPS 140-2 level 3 compliant Security World. Level 2 means a FIPS 140-2 level 2 Security World.

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 127](#).
5. Add the HSM to the working Security World. For more information, see [Adding or restoring an HSM to the Security World on page 111](#).
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-LOCALDESTINATION-KMDATA-LOCALNEW-PROTECTKEY-FILE
[KEY-FILE ...] [NEW-PROTECTKEY-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 only available when exporting keys from a FIPS 140-2 level 3 Security World into a FIPS 140-2 level 2 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 only available when exporting keys from a FIPS 140-2 level 2 Security World into a 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 FIPS-140 level 3 and FIPS-140 level 2 worlds, the module owning world must be FIPS-140 level 2.

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 an HSM to the Security World on page 111](#).
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 (SP800-131A). 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 SP800-131A 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 SP800-131A 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.

Note: The `--disablepkcs1pad` option will only work on SP800-131A Security Worlds.

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...
<code>--version</code>	View the version number of the utility.
<code>-h, --help</code>	Obtain information about the options you can use with the utility.
<code>-m, --module</code>	Specify which module ID to use. If no module is specified, the default is module 1.
<code>-c CARDSETS</code> <code>--cardsets-at-once=CARDSETS</code>	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)
<code>-k KEYS</code> <code>--keys-at-once=KEYS</code>	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). It is recommended to limit the number of keys to be migrated at any one time to no more than 100.
<code>--source=SOURCE</code>	Specify the path to the folder that contains the source Security World data.
<code>--plan</code>	View the list of steps that will be carried out .
<code>--perform</code>	Migrate keys interactively.

Migrating keys to an SP800-131A 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 SP800-131A Security World to which you want to migrate keys.

Note: The `--disablepkcs1pad` option will only work on SP800-131A Security Worlds.

Migrating keys to an SP800-131A 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.

Preparing for migration

Before you begin:

- Install the latest version of the Security World Software from the installation disc. See the Installation Guide for more information.
- Copy the source Security World data to a location that can be accessed by the migration tool.

- Run the utility in the **p1an** 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.

Note: If you have an RFS machine, ensure it is a client and use the same machine to perform the migration. Otherwise, if you use a different client machine, the migration will fail.

Note: You must enable all your features on the target module before migration. Otherwise, the migration will fail.

Migrating keys

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

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

Note: 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 116](#).

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/locall**, as appropriate.

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 140-2 level 3 or FIPS 140-2 level 2 Security World to a FIPS 140-2 level 2 Security World.
- A FIPS 140-2 level 2 Security World to a FIPS 140-2 level 3 Security World.
- A FIPS 140-2 level 3 Security World to another FIPS 140-2 level 3 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.

See [Appendix H: Checking and changing the mode on an nShield Connect on page 277](#) for more about changing the mode.

3. If you have migrated to a FIPS Security World, you must load your destination Security World using the front panel on the module.

Any Security World to FIPS 140-2 level 2 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"> • ACS of the destination world. • ACS of the source world.
Create replacement softcards and/or Operator Card Sets, as appropriate. You can set pass phrases of your choice.	<ul style="list-style-type: none"> • ACS of the destination world. • Blank smart cards to create Operator Card Sets, if appropriate.
Migrate keys to the destination world.	The replacement softcards or Operator Card Sets and pass phrases.
Note: The module is programmed to use the new world by default.	

FIPS 140-2 level 2 Security World to FIPS 140-2 level 3 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"> • ACS of the destination world. • Blank smart cards to create the OCS, if appropriate.
Program the source world into the module.	ACS of the source world.
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"> • ACS of the source world. • ACS of the destination world.
Migrate keys to the destination world.	<ul style="list-style-type: none"> • ACS of the source world. • The replacement softcards or OCS and pass phrases.
If you want to start using the destination world, program the module to use the new Security World. You must also load the destination Security World using the front panel on the module.	ACS of the destination world.

FIPS 140-2 level 3 Security World to FIPS 140-2 level 3 Security World

Note: If both Security Worlds are FIPS 140-2 level 3, the utility creates an intermediate FIPS 140-2 level 2 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"> • ACS of the destination world. • Blank smart cards to create the OCS, if appropriate.
Create an intermediate world and card set.	Blank smart card to create the ACS for the intermediate world.

Step	Requirements
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"> • ACS of the intermediate world. • ACS of the destination world.
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"> • ACS of the intermediate world. • ACS of the destination world.
Migrate keys to the intermediate world.	ACS of the source world.
Migrate keys to the destination world.	<ul style="list-style-type: none"> • ACS of the intermediate world. • The replacement softcards or OCS and pass phrases.
If you want to start using the new world, program the module to use the destination world. You must also load the destination Security World using the front panel on the module.	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 `nfmverify` utility with the `-R` option in combination with the `preload` utility.

Example:

```
preload --admin=RE nfmverify -R -m1 <application> <key-ident>
```

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
	Any migrateable keys found in the source world already exist in the destination world.	
No keys to migrate.	<p>The migration utility returns this error if:</p> <ul style="list-style-type: none"> The keys have already been migrated. All keys are non-recoverable and therefore cannot be migrated. 	None.
<ul style="list-style-type: none"> Source world has indistinguishable cardsets or softcards. Destination world has indistinguishable keys. 	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"> Destination world has indistinguishable card sets or softcards. Source world has indistinguishable card sets or softcards. Source world has indistinguishable keys. 	There are problems with one of the Security Worlds.	Contact Support.
Cannot determine protection of keys.		
<ul style="list-style-type: none"> Source world not recoverable. Destination world not recoverable. 	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"> The path for the source world is wrong. There is no Security World data at the location that was specified when running the migration utility. 	<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>

Error	Explanation	Action
Source world is the same as the destination world.	<ul style="list-style-type: none"> An incorrect path was supplied for the source world data when running the utility. The destination world data does not exist in the default location defined by the environment variable NFAST_KMLOCAL or NFAST_KMDATA. 	<ul style="list-style-type: none"> Run the utility with the correct path to the source world data. Move the source world data to a different location and then copy the destination world data to the default location. 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.
Cannot find <i>NAME</i> utility, needed by this utility.	The software installation is partially completed.	
<i>NAME</i> utility is too old, need at least version <i>VERSION</i> .	<ul style="list-style-type: none"> The software installation is partially completed. The path (in the environment variable for the operating system) might be pointing to an old version of the software. 	<ul style="list-style-type: none"> Reinstall the software. Ensure that the path points to the latest version of the software.
nFast error: TimelimitExceeded; in response to SetKM...	The ACS time-out limit has expired.	Restart the key migration process; see Migrating keys on page 118 .

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 `nvr-am-backup` utility to enable data stored in nonvolatile memory to be backed up and restored.



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 nCipher (that is, it has a random module key and a known κ_{NSO}).

Erasing a module from the unit front panel

To erase a module from a Security World, from the main menu, select **Security World mgmt > Module initialization > Erase Security World**.

When you erase a Security World in this way, the Security World files remain on the remote file system. Delete these files if you wish to remove Security World completely. For more information, see [Security World files on page 92](#).

Erasing a module with new-world

The `new-world` command-line utility can erase any modules that are in the pre-initialization mode.

To erase modules with the `new-world` utility, run the command:

```
new-world [-e|--factory] [-m|--module=MODULE]
```

In this command:

Option	Description
<code>-e, --factory</code>	These options tell <code>new-world</code> to restore a module to its factory state.
<code>-m, --module=MODULE</code>	These options specify the <code>ModuleID</code> to use. <code>new-world</code> erases only one module at a time. To erase multiple modules, you must run <code>new-world</code> 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 [Using KeySafe on page 201](#).)
2. Click the **World** menu button, or select **World** from the **Manage** menu. KeySafe takes you to the **World Operations** panel.
3. Click the **Erase Module** button. KeySafe takes you to the **Erase Module** panel.
4. Select the module that you want to erase by clicking its listing on the **Security world status** tree, then click the **Commit** 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-km1]
```

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-km1` specifies that the module generates an AES (SP800-131A) module signing key, rather than the default key.

Note: The `--disablepkcs1pad` option will only work on SP800-131A Security Worlds.

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.

Replacing an existing Security World

When you erase a Security World from the module's front panel, all long-term key material is deleted from the module's memory and all Security World data is removed from the module's internal file system.

This operation does not remove any files from the remote file system or client machines. You should remove the files manually from the `/opt/nfast/kmdata/local` directory on the remote file system and any client computers to which the Security World was copied.



Any Operator Cards created in a previous Security World cannot be used in the 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 134](#).

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
- It is recommended that you reformat any standard nShield 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 134*.

Note: Except for nShield Remote Administration Cards, if you do not reformat the smart cards used as Operator Cards before you delete your Security World, 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 nCipher key management HSM can access your old keys.

To delete an existing Security World:

1. Remove all the HSMs 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.

Deleting the Security World using the nShield Connect front panel

When you erase a Security World using the unit front panel, all long-term key material is deleted from the HSM's memory and all Security World data is removed from the HSM's internal file system.

- You will not be able to access any of the keys that you have previously used
- Before you remove an old Security World, you must reformat any smart cards that were used previously as Operator Cards within this Security World.

Note: If you do not reformat the smart cards used as Operator Cards before you reinitialize your HSM, 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 the old 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 nCipher key management HSM, can access your old keys.

To erase a Security World using the front panel of the unit, from the main menu select **Security World mgmt > Module initialization > Erase Security World**.

This operation does not remove any files from the RFS or client machines. You should remove the files manually from the `/opt/nfast/kmdata/local` directory on the RFS and any client computers to which the Security World was copied.

Chapter 8: 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 [Creating and managing a Security World on page 91](#).

For more information about key management, see [Working with keys on page 192](#).

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 [Working with keys on page 192](#)).



To perform the tasks described in this chapter, we recommend using the unit front panel or a client on the same computer that contains the RFS. To perform these tasks on a different client, you must transfer the card data to the RFS.



If you are sharing the Security World across several client computers, you must ensure that the changes 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 69](#).

If you want to use the Remote Operator feature to configure smart cards for use with a remote unit, see [Remote Operator on page 187](#).

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 module 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/N) 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 nCipher web site: <https://www.ncipher.com>.

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 modules belonging to the same security world.

Creating (and managing) OCSs can be done with the unit front panel, as described in *Creating an Operator Card Set using the nShield Connect front panel on page 128*. However, you can also use the following tools to create an OCS:

- The `createocs` command-line utility, as described in *Creating an Operator Card Set using the command line on page 128*
- KeySafe, as described in *Creating an Operator Card Set with KeySafe on page 130*

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 local slot of the HSM, or one of its Dynamic Slots. 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 module (unless that application removes the keys).

For more information about persistent OCSs, see *Using persistent Operator Card Sets on page 29*.

An OCS to be used to authorize login on a unit must be persistent and not loadable remotely. It is recommended that such an OCS is not used to protect sensitive keys.

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 using the nShield Connect front panel

To create an OCS, follow these steps:

1. From the main menu, select **Security World mgmt > Cardset operations > Create OCS**.
You are prompted to enter the name of the OCS.
2. Enter a name and press right-hand navigation button.
3. Enter the quorum for the OCS, using the touch wheel to move from one field to the other. The quorum consists of:
 - The maximum number of cards from the OCS required by default for an operation. This number must be less than or equal to the total number of cards in the set.
 - The total number of cards to be used in the OCS. This must be a value in the range 1 – 64.
4. Press the right-hand navigation button to move to the next screen.
5. If you wish to specify a time out for the card set, enter the time out in seconds.
6. Choose whether to create a persistent card set. You can select:
 - **Not persistent** (which is the default)
 - **Persistent**
 - **Remoteable/Persistent**
7. Choose whether to name individual cards and enable pass phrase replacement by answering **Yes** or **No** to each question and then pressing the right-hand navigation button.
8. Insert a smart card to be formatted for the OCS.
If the card is not blank, choose whether to overwrite it or to use a different card. (If the card is an Operator Card from another Security World, you cannot overwrite it and are prompted to enter a different card.)
9. If you have chosen to name individual cards, you are prompted to enter the name for the card.
10. You are asked whether you wish to specify a pass phrase for the card. If you choose **Yes**, you are prompted to enter the pass phrase twice.
While the Operator Card is being created, the screen displays the message **Processing**.
If there are further cards from this OCS to be processed, the screen changes to **Waiting**. Remove the card, and repeat steps 8 through 10 for each of the remaining cards.

When all the cards in the set have been processed, you are told that the card set has been created successfully.

Creating an Operator Card Set using 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
<code>-mMODULE, --module=MODULE</code>	This option specifies the number of the hardware security module to be used to create the token. If you only have one hardware security module, <code>MODULE</code> is 1.
<code>-Q --ocs-quorum=K/N</code>	In this option, <code>K</code> is the minimum required number of cards. If you do not specify the value <code>K</code> , the default is 1. Note: 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 <code>K=1</code> . <code>N</code> is the total number of cards. If you do not specify the value <code>N</code> , the default is 1.
<code>-N --name=NAME</code>	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 <code>-M/--name-cards=NAME</code> options.
<code>-M --name-cards</code>	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 <code>--name=NAME</code> option. <code>createocs</code> prompts for the names of the cards as they are created. Not all applications can display individual card names.
<code>-p --persist</code>	This option creates a persistent card set.
<code>-P --no-persist</code>	This option creates a non-persistent card set.
<code>-R --no-pp-recovery</code>	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 <code>--pp-recovery</code> option.
<code>-q --remotely-readable</code>	This option allows this card set to be read remotely. For information on configuring Remote OCSs, see Remote Operator on page 187 . Note: Not required for Remote Administration.
<code>-T --timeout=TIME</code>	This option sets the time-out for the card set. Use the suffix <code>s</code> to specify seconds, <code>m</code> for minutes, <code>h</code> for hours, and <code>d</code> for days. If the time-out is set to 0, the OCS never times out. Otherwise, the hardware security module automatically unloads the OCS when the amount of time specified by <code>TIME</code> has passed since the OCS was loaded.
<code>-e --erase</code>	Specifying this option erases a card (instead of creating a card set). You can specify this option twice in the form <code>-ee</code> to repeatedly erase cards.

Note: With Security World Software v11.72 and later, pass phrases are limited to a maximum length of 254 characters, when using `createocs`. See [Maximum pass phrase length on page 30](#) for more information.

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.

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 module 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 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 [Using KeySafe on page 201](#).)
2. Click the **Card sets** menu button, or select **Card sets** from the menu. The **List Operator Card Sets** panel is displayed.
3. Select an HSM within the Security World from the Security World status pane.
4. Click the **Create new card set** button to open the **Create Operator Card Set** panel. You can specify the following options:
 - a. A name for the card set.
 - b. Whether pass phrase recovery will be enabled for the OCS. (Only available if the Security World has pass phrase recovery enabled.)
 - c. Whether the card set can be used remotely. (Only available if the Security World has remote sharing available.) For more information, see [Remote Operator on page 187](#).
 - d. Whether this OCS will be persistent.
 - e. Whether this OCS will have a time-out (a period after which the card set must be inserted again).
 - f. The value for the time-out, in seconds.
 - g. The total number of Operator Cards (N) that you want this OCS to have. This must be a value in the range 1 – 64.
 - h. The number of Operator Cards needed to re-create a key (K). K must be less than or equal to N .
5. When you have entered all the details, click **Commit**. KeySafe takes you to a new **Create Operator Card Set** panel.

Note: If K is equal to N , a message is displayed:
The total number of cards is equal to the required number of cards. – If the total and required number of cards are equal, losing one card will render any nonrecoverable keys unusable. Is this what you want?
Click **Yes** to confirm the values for K and N , or **No** to change them.

Note: 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.
6. Insert a blank, unformatted card into the reader.

A message is displayed, confirming that the card is blank. Click **OK** to open the **Set Card Protection Passphrase** panel.

Note: If you insert a card from another OCS, 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.

If you insert an OCS card from another security world you will get the message:
Error. Unreadable card - may be incorrectly inserted or be from another security world's operator card set. Please check.
To overcome this you must replace the card you have inserted with another card that is readable (or blank).

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.
7. 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.

8. If setting 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).

Note: 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.
9. 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.
10. 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 132](#))
- KeySafe (see [Creating softcards with KeySafe on page 133](#))

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. 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).
3. 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 [Using KeySafe on page 201](#).)
2. Click the **Softcards** menu button, or select **Softcards** from the **Manage** menu. KeySafe takes you to the **List Softcards** panel.
3. Click **Create New Softcard** to open the **Create Softcard** panel.
4. Choose parameters for the softcard:
 - a. Enter a name for the softcard. You must provide a valid name for each softcard.
 - b. Choose whether you want pass phrase replacement to be enabled for the softcard.

Note: In a Security World with passphrase recovery enabled the **Yes** radio button is selected as default and the selection can be changed between **Yes** and **No**. In a Security World with passphrase recovery disabled the **No** button is selected, and cannot be changed to **Yes**.
5. Click **Commit**.

Note: 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.

The **Set Softcard Protection Pass Phrase** pane is displayed.
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. You must provide a passphrase for each card.
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.



In the case of an OCS that uses nShield Remote Administration Cards, it is possible to reformat the cards at any time using `slotinfo --ignoreauth`. In the case of an OCS that uses standard nShield cards, it is only possible to erase or format the cards within the Security World in which they were created.

You can erase Operator Cards using the unit front panel, 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 from the `opt\nfast\kmdata\1oca1` directory 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 137](#).

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 card sets using the nShield Connect front panel

To erase a card set using the front panel, follow this procedure:

1. From the main menu select: **Security World mgmt > Card operations > Erase card**
2. Insert the card set that you want to erase. The card is read.
3. You are asked to confirm that you want to erase this card from the card set.
4. To confirm, press the right-hand navigation button.
5. You are asked once again if you want to erase this card.
6. To confirm, press the right-hand navigation button.

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 [Using KeySafe on page 201](#).)
2. Click the **Card Sets** 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 from the `opt\nfast\kmdata\local` 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 **Discard Card Set(s)** feature.

Erasing cards using 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
<code>-m --module=MODULE</code>	These options specify the module number of the module. If you only have one module, <code>MODULE</code> is 1.
<code>-e --erase</code>	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.

As an alternative, you can reformat using `slotinfo --format`.

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. Select the softcard you want to erase from the list.
4. Click the **Discard Softcard** button.
5. KeySafe asks you to confirm that you want to erase this card. Click **Yes** to confirm.
6. 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 `nfkmiinfo` (*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 the front panel, KeySafe or the `nfkmiinfo` command-line utility. To check which pass phrase is associated with a card, you can use the front panel or 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 `nfkmiinfo` command-line utilities. To check which pass phrase is associated with a softcard, you can use the `ppmk` command-line utility.

Viewing card sets using the nShield Connect front panel

You can use the unit front panel to view details of all the Operator Cards in a Security World or to view details of an individual Operator Card.

To view a list of all the card sets in the Security World, from the front panel select **Security World mgmt > Cardset operations > List cardsets**.

To view details of a single card using the unit front panel:

1. Insert the card into the unit.
2. From the main menu, select **Security World mgmt > Card operations > Card details**.
3. The type of the card (Administrator or Operator) is displayed with the number of the card in the card set.

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 [Using KeySafe on page 201](#).)
2. Click the **Card Sets** menu button, or select the **Card sets** menu item from the **Manage** menu. KeySafe takes you to the **List Operator Card Sets** panel.
3. Click **Examine/Change Card** to open 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
- Erase the card.

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 [Using KeySafe on page 201](#).)
2. Click the **card sets** menu button, or select the **Card sets** menu item from the **Manage** menu. 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:

- Examine / change a card (see [Examining a Card on page 137](#))
- Create a new card set (see [Creating an Operator Card Set with KeySafe on page 130](#))
- Replace an Operator Card Set (see [Replacing OCSs with KeySafe on page 146](#))
- Discard a card set (see [Erasing cards with KeySafe on page 134](#)).

Viewing card sets using 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: 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 134*.

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 using the nShield Connect front panel

To verify the pass phrase associated with a card using the unit front panel:

1. Insert the card into the unit.
2. From the main menu, select **Security World mgmt > Card operations > Check PIN**.
The type of the card (Administrator or Operator) is displayed with the number of the card in the card set.
3. If this is the card that you want to check, press the right-hand navigation to confirm.
4. Enter the pass phrase.
If the pass phrase that you entered is correct, a confirmation message is shown. Otherwise, an error is reported.

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
<code>--check</code>	This option tells <code>cardpp</code> to check the pass phrase.
<code>--module=MODULE</code>	This option specifies the number of the module to use. If you only have one module, <code>MODULE</code> is 1. If you do not specify a module number, <code>cardpp</code> 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 phrase

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 a pass phrase 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 phrase can be changed using the front panel, KeySafe or the `cardpp` command-line utility; softcard pass phrase 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 phrase

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 a pass phrase and others do not.

Note: Prior to Security World Software v11.72, we set no absolute limit on the length of a pass phrase. However, some applications may not accept a pass phrase 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. nCipher recommends that your password only contains 7-bit ASCII characters:

```
A-Z, a-z, 0-9, ! @ # $% ^ & * - _ + = [] {} | \ : ' , . ? / ` ~ " < > ( ) ;
```

See [Maximum pass phrase length on page 30](#) for more about pass phrase length when using Security World Software v11.72.

Changing known pass phrase from the unit front panel

To change the pass phrase of a card using the unit front panel:

1. Insert the card.
2. From the main menu, select **Security World mgmt > Card operations > Change PIN**.
3. Select the card whose pass phrase you want to change.
4. Enter the old pass phrase, and then enter it again to confirm it.
5. Enter the new pass phrase. If you do not want this card to have a pass phrase, select **NO** at the prompt.

Changing known pass phrase 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 [Using KeySafe on page 201](#).)
2. Click **Card sets**, or select **Card sets** from the **Manage** menu. The **List Operator Card Sets** panel is displayed.
3. Click **Examine / change card** to open the **Examine / Change Card** panel.
4. Click **Change pass phrase**. The Set Card Protection pass phrase panel is displayed.
5. Enter the old pass phrase, and click the **OK** button.
6. A screen is displayed asking **Do you want to set a pass phrase?**. Select **Yes**.
7. Enter your new pass phrase, and enter it again in the second box as confirmation of the change.
8. 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 [Using KeySafe on page 201](#).)
2. Click the **Softcards** menu button, or select **Softcards** from the **Manage** menu. KeySafe takes you to the **List Softcards** panel.
3. Select the softcard for which you want to change the pass phrase, and click the **Change Pass phrase** button. KeySafe takes you to the **Change/Recover Softcard Pass phrase** panel.
Note: If a softcard is listed as PIN Recovery Enabled = No, then you will be unable to change the pass phrase.
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 phrase 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 a pass phrase and others do not. A pass phrase can be of any length and can

contain any characters that you can type.

Note: With Security World Software v11.72 and later, pass phrases are limited to a maximum length of 254 characters, when using `cardpp`. See [Maximum pass phrase length on page 30](#) for more information.

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 HSM, `MODULE` is 1. If you do not specify an HSM number, `cardpp` uses all HSMs 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 phrase with `ppmk`

Note: With Security World Software v11.72 and later, pass phrases are limited to a maximum length of 254 characters, when using `ppmk`. See [Maximum pass phrase length on page 30](#) for more information.

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 phrase

Changing unknown card pass phrase 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 module to use. If you only have one hardware security module, *MODULE* is 1. If you do not specify a number, `cardpp` uses all hardware security modules 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 module that is connected to a trusted server.

Replacing unknown pass phrase 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 phrase does 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 module 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, or you want to migrate from standard nShield cards to nShield Remote Administration Cards, you should use one of the following:

- The `rocs` utility
- The KeySafe **Replace Operator Card Set** option.
Accessed from the **card operations** panel.

Note: You cannot mix standard nShield cards with nShield Remote Administration Cards. in the same set.

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 134](#).

Deleting the information about an OCS from the client 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, nCipher supplies several utilities that can recover the keys protected by the lost Operator Card to another OCS.

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 Security World 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, you cannot recover keys from lost or damaged smart cards or softcards.
- Have created the original OCS using the front panel of the unit, `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 the front panel of the unit, `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 client 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 69](#).

Replacing OCSs from the unit front panel

To replace an OCS from the unit front panel, follow these steps:

1. From the main menu, select **Security World mgmt > Admin operations > Recover keys**.
2. Select **all** to recover all keys in the Security World, or select the application for which you want to recover the keys.
3. If you selected an application, select the keys that you want to recover.
4. Insert the required number of Administrator Cards to recover keys, and enter their pass phrases if required.
5. Insert the required number of Operator Cards, and enter their pass phrases if required. When you have inserted the required number of cards, details of the recovered key are displayed.
6. Check the key details are correct and then scroll down and select **Recover key**. You can also select **More info** to see more information about the keys.

A message is displayed when the keys are recovered.

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 127](#).

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 [Using KeySafe on page 201](#).)
2. Click the **Card Sets** menu button, or select **Card Sets** from the **Manage** menu. KeySafe takes you to the **List Operator Card Sets** panel.
3. Click **Replace card set**. KeySafe takes you to the **Replace card set** panel. This panel lists existing OCSs in tabular form. For each card set it displays:

Attribute	Description
Name	The name of the card set.
Required (K)	The number of cards needed to re-create a key.
Total (N)	The total number of cards in the set.
Persistent	Indicates whether or not the card set is persistent.
Timeout	The timeout value of the card, in seconds
Recoverable Key Count	The number of private keys protected by this card set that are recoverable.
Nonrecoverable Key Count	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, and click **Replace card set**.
 - Note:** If an OCS does not have any recoverable keys, it cannot be replaced.
- 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.
- 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 module, 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.
- 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 using 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 module to use by using the `module` command, which is described in the section [module number on page 151](#).
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 149](#).
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 152](#).

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 150](#).
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 151](#).
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 152](#).
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 151](#). `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 module 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 152](#)). 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 151](#)).

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
No.	The card set or softcard number, which you can use to identify this card set in rocs commands.
Name	The OCS or softcard name.
Keys	The number of keys protected by this OCS or softcard.
(recov)	The number of keys protected by this OCS or softcard.
Sharing	The <i>K</i> of <i>N</i> parameters for this OCS.
persistent	The OCS is persistent and does not have a time-out set.
### minute timeout	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
No.	The key number, which you can use in mark and unmark commands.
Name	The key name.
App	The application with which the key is associated.
Protected by	This indicates the protection method (see table below).

In this output, the protection methods include:

Method	Description
module	Key protected by the Security World.
name	Key protected by the named OCS or softcard.
name->name2	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 (name->name2)	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 153](#).

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 module to be used. The module number must correspond to a hardware security module in the current Security World. If the hardware security module 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 module 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` using 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
<code>-m, --module=MODULE</code>	These options specify the number of the hardware security module to use.
<code>-t, --target=CARDSET-SPEC</code>	These options specify the OCS or softcard to be used to protect the keys. For more information, see Specifying card sets on page 153 .
<code>-k, --keys=KEYS-SPEC</code>	These options select the keys to be recovered. For more information, see Specifying keys on page 153 .
<code>-c, --cardset=CARDSET-SPEC</code>	These options select all keys that are protected by the given OCS or softcard. For more information, see Specifying card sets on page 153 .
<code>-i, --interactive</code>	These options force <code>rocs</code> 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 module, 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 147](#).

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
<code>[number] cardset-number</code>	A value of this form selects the OCS or softcard with the given number from the list produced by the <code>list cardsets</code> command.
<code>[name] cardset-name</code>	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).
<code>hash cardset-hash</code>	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` using 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
<code>mark key-number</code>	A value of this form selects the key with the given number from the list produced by the <code>list keys</code> command. Examples of usage are: <hr/> <code>rocs -t target_OCS -k key_number</code> <hr/> and <hr/> <code>rocs -t target_OCS -k "mark 56"</code> <hr/>

Value	Description
<code>appname:keyident</code>	<p>A value of this form selects keys by their internal application name and ident. You must supply at least one of appname or keyident, but you can use wildcard patterns for either or both in order to select all matching keys. An example of usage is:</p> <hr/> <pre>rocs -t target_OCS --keys="simple:simplekey"</pre> <hr/>
<code>hash keyhash</code>	<p>A value of this form selects the key with the given key hash. An example of usage is:</p> <hr/> <pre>rocs -t target_OCS --keys="hash e364[...]"</pre> <hr/>
<code>--cardset cardset-spec</code>	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, or you want to migrate from standard nShield cards to nShield Remote Administration Cards, you should use one of the following to create a new set:

- The `racs` utility
- The front panel of the nShield Connect
- The KeySafe **Replace Administrator Card Set** option
Accessed from the **Card Operations** panel

Note: If further cards are damaged, you may not be able to re-create your Security World.

Note: You cannot mix standard nShield cards with nShield Remote Administration Cards. in the same set.



Replacing the ACS modifies the `world` file. In order to use the new ACS on other machines in the Security World, you must copy the updated `world` file to all the machines in the Security World after replacing the ACS. Failure to do so could result in loss of administrative access to the 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 Administrator Card Set using the nShield Connect front panel

To replace an ACS:

1. From the main menu, select **Security World mgmt > Admin operations > Replace ACS**.
2. Insert one of the remaining cards from the card set that you want to replace and press the right-hand navigation button.
Continue to insert cards until you have inserted the number of cards required to authorize the process.
3. When prompted, insert a card for the replacement card set and press the right-hand navigation button.
4. If required, specify a pass phrase for the card.
5. Insert cards until the card set is complete. A message confirms that the card set has been created.
6. At this point the modified `world` file has been pushed to the RFS, so make a backup of the modified `world` file on the RFS, preferably in a way that distinguishes it from previous backups.
7. Copy the `world` file to any other Connects in the same Security World, either using the **Security World mgmt > RFS operations > Update World files** option on the Connect concerned or using the `nethsmadmin` utility, see [Using nethsmadmin to copy a Security World to an nShield Connect and check the current version on page 100](#).
8. If client cooperation is not enabled, copy the modified `world` file onto any client machines where it is needed.
9. Check that the new Administrator Cards are usable and that their passphrases have been set as intended, see [Verifying the pass phrase of a card or softcard on page 140](#)
10. Erase the Administrator Cards from the old card set. For more information, see [Erasing cards and softcards on page 134](#).

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 [Using KeySafe on page 201](#)).
2. Click the **Card sets** menu button, or select **Card sets** from the **Manage** menu. KeySafe takes you to the **List Operator Card Sets** 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:
 - a. Select the **Yes** option.
 - b. Enter the same pass phrase in both text fields.
 - c. 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 **List Operator Card Sets** panel) or the `cardpp` command-line utility.
9. If you do not want to set a pass phrase for this Administrator Card:
 - a. Select the **No** option.
 - b. Click the **OK** button.
10. After you have created all the Administrator Cards, KeySafe displays a message confirming that the ACS has been successfully replaced.
11. Click the **OK** button, and KeySafe returns you to its introduction panel.

When you have finished replacing the ACS:

1. If you ran KeySafe on a client machine, ensure that there is a copy of the modified `world` file on the RFS.
2. Make a backup of the `world` file, preferably in a way that distinguishes it from previous backups.
3. Copy the `world` file to any other Connects in the same Security World, for example using the `nethsmadmin` utility, see [Using `nethsmadmin` to copy a Security World to an nShield Connect and check the current version on page 100](#).
4. If client cooperation is not enabled, copy the modified `world` file onto any other client machines where it is needed.
5. Check that the new Administrator Cards are usable and that their passphrases have been set as intended, see [Verifying the pass phrase of a card or softcard on page 140](#).
6. Erase the old Administrator Cards; for more information, see [Erasing cards and softcards on page 134](#).

Replacing an Administrator Card Set with `racs`

The `racs` utility creates a new ACS to replace a set that was created on the module.

1. Ensure the hardware security module 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.
5. If you ran `racs` on a client machine, ensure that there is a copy of the modified `world` file on the RFS.
6. Make a backup of the `world` file, preferably in a way that distinguishes it from previous backups.
7. Copy the `world` file to any other Connects in the same Security World, for example using the `nethsmadmin` utility, see [Using `nethsmadmin` to copy a Security World to an nShield Connect and check the current version on page 100](#).
8. If client cooperation is not enabled, copy the modified `world` file onto any other client machines where it is needed.
9. Check that the new Administrator Cards are usable and that their passphrases have been set as intended, see [Verifying the pass phrase of a card or softcard on page 140](#).
10. Erase the old Administrator Cards; for more information, see [Erasing cards and softcards on page 134](#).

Chapter 9: Application interfaces

This chapter explains how to use an HSM 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 *Working with keys on page 192*). 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 HSM to the group `nfast` before the application runs.

Note: If you create keys on a client that is not on the same computer as the RFS, you must copy the key data to the RFS before the nShield HSM can use these keys.

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).

By default the library makes use of all hardware security modules that it finds when starting up. With module firmware version 2.65.2 or later, if your CHIL application only uses module protected keys, you can use HSM Pool mode with multiple hardware security modules to make your application tolerant of hardware or network errors as long as one module remains functional. For CHIL, HSM Pool mode is enabled by setting the `NFAST_HWCRHK_HSM_POOL` environment variable in the environment of the process running the application that wants to use HSM Pool mode, see *Setting environment variables on page 72*.

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 192](#). 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 (`-`).

nCipherKM 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 HSMs. 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.

See the Installation Guide for more about:

- The bundles and components supplied on your Security World Software installation media
- The versions of Java that are supported by the Security World Software

Detailed documentation for the JCE interface can be found on the Oracle Technology web page <https://docs.oracle.com/javase/8/docs/technotes/guides/security/crypto/CryptoSpec.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 client hardserver on page 89](#).

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)	<code>\$JAVA_HOME/jre/lib/ext</code>
Java Runtime Environment (JRE)	<code>\$JAVA_HOME/lib/ext</code>

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)	<code>\$JAVA_HOME/jre/lib/security</code>
Java Runtime Environment (JRE)	<code>\$JAVA_HOME/lib/security</code>

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 nCipherKM 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 nCipherKM provider first in the list permits the nCipherKM 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 nCipherKM 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 nCipherKM 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 91](#).

Note: If you have a Java Enterprise Edition Application Server running, you must restart it before the installed nCipherKM 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 **nCipherKM** 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 159](#).

If the **nCipherKM** 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 159](#). For more information about hardserver configuration file settings, see [server_startup on page 260](#).
- 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 91](#).
- If you have already created a Security World, repeat the **nCipherKM JCA/JCE CSP** installation process as described in [Installing the nCipherKM JCA/JCE CSP on page 159](#).

After making any changes to the **nCipherKM JCA/JCE CSP** installation, run the `InstallationTest` command again and check the output.

Whether or not the nCipherKM 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.
```

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 159](#).

keytool

You can use either the Oracle `keytool` utility or the IBM `keytool` utility to read and edit an nShield KeyStore. These utilities are shipped with the Oracle and IBM JVMs. You must specify the correct `nCipher.world` KeyStore type when you run the `keytool` utility, and you must specify the correct package name for the Oracle or IBM `keytool` utility.

To generate a new key in an OCS-protected KeyStore with the Oracle or IBM `keytool` utility, run the appropriate command:

- Sun Microsystems `keytool` utility:

For Java 8, use the following command:

```
java sun.security.tools.keytool.Main -genkey -storetype nCipher.world -keyalg RSA -sigalg SHA1withRSA -storepass KeyStore_passphrase -keystore KeyStore_path
```

For Java 7 or earlier, use the following command:

```
java sun.security.tools.KeyTool -genkey -storetype nCipher.world -keyalg RSA -sigalg SHA1withRSA -storepass KeyStore_passphrase -keystore KeyStore_path
```

- IBM `keytool` utility:

```
java com.ibm.crypto.tools.KeyTool -genkey -storetype nCipher.world -keyalg RSA -sigalg SHA1withRSA -storepass KeyStore_passphrase -keystore KeyStore_path
```

In these example commands, `KeyStore_passphrase` is the pass phrase for the OCS that protects the KeyStore and `KeyStore_path` is the path to that KeyStore.

To generate a new key in a module-protected KeyStore with the Oracle or IBM `keytool` utility, run the appropriate command:

- Sun Microsystems **keytool** utility:

For Java 8, use the following command:

```
java -Dprotect=module -DignorePassphrase=true sun.security.tools.keytool.Main -genkey -storetype nCipher.sworld -keyalg RSA -sigalg SHA1withRSA -keystore KeyStore_path
```

For Java 7 or earlier, use the following command:

```
java -Dprotect=module -DignorePassphrase=true sun.security.tools.KeyTool -genkey -storetype nCipher.sworld -keyalg RSA -sigalg SHA1withRSA -keystore KeyStore_path
```

- IBM **keytool** utility:

```
java -Dprotect=module -DignorePassphrase=true com.ibm.crypto.tools.KeyTool -genkey -storetype nCipher.sworld -keyalg RSA -sigalg SHA1withRSA -keystore KeyStore_path
```

In these example commands, *KeyStore_path* is the path to the KeyStore.

By default, the **keytool** utilities use the **MD5withRSA** signature algorithm to sign certificates used with a KeyStore. This signature mechanism is unavailable on modules with firmware version 2.33.60 or later.

Using keys

Only the nCipherKM provider can use keys stored in an nShield KeyStore because the underlying key material is held separately in the Security World.

You can always store nShield keys in an nShield KeyStore. You can also store keys generated by a third-party provider into an nShield KeyStore if both of the following conditions apply:

- the key type is known to the nCipherKM provider
- the Security World is *not* compliant with FIPS 140-2 level 3.

When you generate an nShield key (or create it from imported key material), that key is associated with an ACL (Access Control List). This ACL prevents the key from being used for operations for which it is unsuited and enforces requirements that certain tokens be presented; for example, the ACL can specify that signing key cannot be used for encryption.

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
JCECSP_DEBUG	This property is a bit mask for which different values specify different debugging functions; the default value is 0 . For details about the effects of setting different values for this property, see JCECSP_DEBUG property values on page 166 .
JCECSP_DEBUGFILE	This property specifies a path to the file to which logging output is to be written. Set this property if the JCECSP_DEBUG property is set to a value other than the default of 0 . For details about the effects of setting different values for this property, see JCECSP_DEBUG property values on page 166 . In a production environment, we recommend that you disable debug logging to prevent sensitive information being made available to an attacker.
protect	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 module , softcard:IDENT or cardset . OCS protection (cardset) uses the card from the first slot of the first usable hardware security module. To find the logical token hash IDENT of a softcard, run the command nfkminfo --softcard-list .
module	This property lets you override the default HSM and select a specific HSM to use for HSM and OCS protection. Set the value of this property as the ESN of the HSM you want to use.
slot	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.
ignorePassphrase	If the value of this property is set to true , the nCipherKM provider ignores the pass phrase provided in its KeyStore implementation. This feature is included to allow the Oracle or IBM keytool utilities to be used with module-protected keys. The keytool utilities require a pass phrase be provided; setting this property allows a dummy pass phrase to be used.

Property	Function for different values
<code>seeintegname</code>	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.
<code>com.ncipher.provider.announcemode</code>	The default value for this property is <code>auto</code> , which uses firmware auto-detection to disable algorithms in the provider that cannot be supported across all installed HSMs. Setting the value of this property to <code>on</code> forces the provider to advertise all mechanisms at start-up. Setting the value of this property to <code>off</code> forces the provider to advertise no mechanisms at start-up.
<code>com.ncipher.provider.enable</code>	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.
<code>com.ncipher.provider.disable</code>	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 <code>com.ncipher.provider.disable</code> property overrides the same mechanism if it is supplied in the value for the <code>com.ncipher.provider.enable</code> 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
0	If this property has no bits set, no debugging information is reported. This is the default setting.
1	If this property has the bit 1 set, minimal debugging information (for example, version information and critical errors) is reported.
2	If this property has the bit 2 set, comprehensive debugging information is reported.
4	If this property has the bit 3 set, debugging information relating to creation and destruction of memory and HSM resources is reported.
8	If this property has the bit 4 set, <code>debugFunc</code> and <code>debugFuncEnd</code> generate debugging information for functions that call them.
16	If this property has the bit 5 set, <code>debugFunc</code> and <code>debugFuncEnd</code> display the values for all the arguments that are passed in to them.
32	If this property has the bit 6 set, context information is reported with each debugging message (for example, the <code>ThreadID</code> and the current time).

JCECSP_DEBUG value	Function
64	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.
128	If this property has the bit 8 set, debugging information for NFJAVA is reported in the debugging file.
256	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 HSMs, providing greater scalability.

Note: We recommend that you use load-sharing unless you have existing code that is designed to run with multiple HSMs. To share keys with load-sharing, you must create a 1/N OCS with at least as many cards as you have HSMs. All the cards in the OCS must have the same pass phrase.

Note: The nCipherKM JCA/JCE CSP does not support HSM Pool mode. If you want to use HSM Pool mode with a Java application that only uses module protected keys, one option may be to use the Sun PKCS #11 provider to access the nCipher PKCS #11 library instead of using nCipherKM JCA/JCE CSP.

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 nCipher Web site: <https://www.ncipher.com>. 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 *nCipher PKCS #11 library on page 168*.

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 mode. This feature provides support for multiple hardware security modules that are connected to a single server, spreading the load of cryptographic operations between the HSMs in order to provide scalability in terms of performance.

To share OCS protected keys with load-sharing mode, you must create a 1/*N* OCS that contains at least as many cards as you have HSMs. All the cards on the OCS must have the same pass phrase.

With module firmware version 2.65.2 or later, if your application only uses module protected keys, you can use HSM Pool mode as an alternative to using load-sharing mode. HSM Pool mode supports returning or adding a hardware security module to the pool without restarting the system.

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 HSM can provide.

Generating random numbers and keys

The nShield HSM 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 HSM to generate random numbers and keys.

Digital signatures

The nCipher PKCS #11 library can use the nShield HSM to sign and verify messages using the following algorithms:

- DSA
- RSA
- DES3_MAC

- AES
- ECDSA (if the appropriate feature is enabled)

An nCipher hardware security module is specifically optimized for public key algorithms, and therefore it will provide significant acceleration for DSA, RSA and ECDSA signature generation and verification. You should always choose to perform asymmetric signature generation and verification with an nShield HSM.

Asymmetric encryption

The nCipher PKCS #11 library can use an nShield HSM to perform asymmetric encryption and decryption with the RSA algorithm.

The nShield HSM is specifically optimized for asymmetric algorithms, so you should always choose to perform asymmetric operations with the nShield HSM.

Symmetric encryption

The nCipher PKCS #11 library can use the nShield HSM 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 HSM than on the host computer. However, although the nShield HSM 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 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 mechanisms currently supported by the nCipher PKCS #11 library, including some vendor-supplied mechanisms, are listed in the *Cryptographic API Integration Guide*.

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 HSM 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 176).

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 176.

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 HSM. 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 HSMs or PKCS #11 slots.

Whether or not the library uses load-sharing mode depends on the value of the `CKNFAST_LOADSHARING` environment variable, described in [CKNFAST_LOADSHARING on page 175](#). Whether or not the library uses HSM Pool mode depends on the value of the `CKNFAST_HSM_POOL` environment variable, described in [CKNFAST_HSM_POOL on page 175](#).

nCipher PKCS #11 library with load-sharing mode

If load-sharing mode is enabled, the nCipher PKCS #11 library creates a virtual slot for each 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 174](#)).

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 175](#)). 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 belong 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 HSMs 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 with HSM Pool mode

If HSM Pool mode is enabled, the nCipher PKCS #11 library exposes a single pool of HSMs and a single virtual slot for a fixed token with the label `accelerator`. This accelerator slot can be used to create module protected keys and to support session objects. HSM Pool mode does not support token protected keys, any pre-existing OCS or softcard protected keys are hidden from PKCS #11. In FIPS 140-2 level 3 Security Worlds, keys cannot be created in HSM Pool mode, however keys created outside HSM Pool mode, for example using `generatekey` or a non-Pool mode PKCS #11 application, can be used in HSM Pool mode.

nCipher PKCS #11 library without load-sharing

There will be two entries for each HSM, 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 a 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 172](#)). 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 218](#)).

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 HSM with the same card set.

If the loaded card set is non-persistent, then a card must be left in each HSM 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_CONCATENATIONKDF_X963_COMPLIANCE
- CKNFAST_DEBUG
- CKNFAST_DEBUGDIR
- CKNFAST_DEBUGFILE
- CKNFAST_FAKE_ACCELERATOR_LOGIN
- CKNFAST_HSM_POOL
- CKNFAST_LOADSHARING
- CKNFAST_NO_ACCELERATOR_SLOTS
- CKNFAST_NO_SYMMETRIC
- CKNFAST_NO_UNWRAP
- CKNFAST_NONREMOVABLE
- CKNFAST_NVRAM_KEY_STORAGE
- CKNFAST_OVERRIDE_SECURITY_ASSURANCES
- CKNFAST_SEED_MAC_ZERO
- 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 of the client.

Note: The `cknfastrc` file should be saved without any suffix (such as `.txt`).

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 nShield 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 `hk1tu`. For more information about using `nfkminfo`, see [nfkminfo: information utility on page 230](#).

CKNFAST_CONCATENATIONKDF_X963_COMPLIANCE

Sets the correct use of ECDH derive with concatenate KDF using the ANSI X9.63 specification as per the PKCS#11 standard.

Note: The default will be the old SP800-56a to maintain backward compatibility.

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
0	None (default setting)
1	Fatal error
2	General error
3	Fix-up error
4	Warnings
5	Application errors
6	Assumptions made by the nCipher PKCS #11 library
7	API function calls
8	API return values
9	API function argument values
10	Details
11	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 HSM can enter an arbitrary PIN and obtain the certificate.

CKNFAST_HSM_POOL

HSM Pool mode is determined by the state of the `CKNFAST_HSM_POOL` environment variable.

Set the environment variable to `1`, `y` or `Y` to enable HSM Pool mode for the PKCS #11 application, or set to `0`, `n` or `N` to explicitly disable HSM Pool mode for the PKCS #11 application.

HSM Pool mode takes precedence over load-sharing mode. HSM Pool mode only supports module protected keys so do not use `CKNFAST_NO_ACCELERATOR_SLOTS` to disable the accelerator slot.

CKNFAST_LOADSHARING

Load-sharing mode 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` and ensure that the `CKNFAST_HSM_POOL` environment variable is not set. The virtual slot behavior then operates.

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. Using 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 180*.

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 HSM 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 HSM'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 HSM'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 HSM 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 HSM 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 `a11` 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 `a11`, 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_SEED_MAC_ZERO

Set this variable to use zero padding for the Korean SEED MAC mechanisms (`CK_SEED_MAC` and `CKM_SEED_MAC_GENERAL`). If this variable is not set, or is set to `n`, then the SEED MAC mechanisms will use the default PKCS#5 padding scheme.

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.

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 HSMs 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.##"
      implementation version 1.##
      Strictfips 140 enabled
      Load sharing and Failover enabled

slot  Status          Label
=====
0     Fixed token     "accelerator"
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.##** refers to the version of the nCipher PKCS #11 library
- **Loadsharing and Failover enabled** is shown if load-sharing has been enabled. Alternatively **Pool mode enabled** is shown if Pool mode has been enabled.

Slots that contain a valid Operator Card are indicated by the status **Operator card** and the card's label. A fixed token is always available and is listed as slot 0.

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 there is no card in a slot, `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 an operator card or there are no physical slots, enter a fixed token slot number,
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 `R`
 - choose a fixed token slot
 - Press `E` 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 `Enter`. 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 `Enter`.
`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 an HSM and never leave that HSM. 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 HSM key	well known HSM key

Operator Card Set

The object is stored as an nShield 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 nShield key blob encrypted by the security world key. This object can be loaded on to any HSM 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 HSM key. This encryption is for programming convenience only and does not provide security. These keys can be loaded on any nShield HSM.

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 unit on page 81*.

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.

To use a standalone application:

1. Ensure that the SEE machine for the application is in the directory `/opt/nfast/custom-seemachines` on the remote file system.

Note: If an SEE machine has previously been loaded on the HSM, press the Clear button on the front of the unit before proceeding to the next step. This clears the current SEE machine from memory.

2. From the main menu on the front panel of the HSM, select **codeSafe**.

3. To enable the HSM to publish the SEE World for multiple clients, enter the following information when prompted:
 - The name of the SEE machine file.
 - The name of the user data file, if required.
 - The type of custom SEE machine you are using (select **SEElib** or **BSDlib sockserv**).

Note: This option is only available if you have provided a valid user data file in step 2. If **BSDlib sockserv** is selected, `worldid_pubname`, `postload_prog`, and `postload_args` will be passed to `load_seemachine`. For detailed descriptions of the options in this section, see [load_seemachine on page 261](#).
 - The ID of the SEE World to create.

Note: This option is only available if you have selected the **SEElib** option in the previous step.

Note: To use `see-sock-serv` directly, you must select **BSDlib sockserv**.

Remotely loading and updating SEE machines

The SEE remote push facility allows the remote deployment of CodeSafe SEE machines to an nShield Connect, negating the need to physically visit the HSM to load or update the SEE machine. This is achieved by editing the configuration file on the RFS for a specific nShield Connect to specify the new SEE machine, then setting a configuration flag in the config file to `true`. This initiates the push of the config file to the target HSM.

Before configuring a module to autonomously run an SEE machine and accept updates using the RFS, that module must first be set up to accept remotely-pushed configurations; see [Configuring the nShield Connect with configuration files on page 68](#).

The auto push feature allows future unit configuration to be performed remotely (that is, without access to the front panel of the unit). To enable auto push, use the right-hand navigation button on the front panel of the nShield Connect to select **System > System configuration > Config file options > Allow auto push** and select **ON**. (See [Configuring auto push on page 59](#) for more information.)

For more information about configuring log file storage options, see [Configuring log file storage on page 59](#).

Note: Both **SEElib** and **BSDlib sockserv** SEE machines are supported on the nShield Connect.

To configure an nShield Connect module to autonomously run an SEE machine and accept updates using the RFS:

1. Copy the existing config file to a new file called `config.new`.
2. In the `load_seemachine` section of the `config.new` file for the remote module, add or amend the following settings:

```
pull_rfs=true
machine_file=mymachinename.sar
userdata=myuserdata.sar
worldid_pubname=publ_name
```

Note: These settings specify the type, name and user data of the SEE machine you wish to load. For more information about each setting, see [load_seemachine on page 261](#).

Note: For CodeSafe Direct, the `userdata` file must be packed as a SAR file.

Note: The remote module will load the new SEE machine in place of any existing SEE machine. If no `machine_file` value is set, then pushing the config file will remove any existing machines on the unit.

3. In the `sys_log` section of the `config.new` file for the remote module, add or amend the following settings:

```
behaviour=push
push_interval=1
```

Note: These settings control how and where log messages are written. Using the example above, messages will be written to the `system.log` and `hardserver.log` files of the module, which are accessible using the remote file system.

You may wish to revise the `push_interval` to a higher value once the nShield Connect has successfully loaded the new SEE machine.

4. Run `nopclearnfail` to clear the module, followed by `enquiry` to check that the module is ready.
5. Run `cfg-pushnethsm` to push the new config file to the module.

To load a new SEE machine to multiple nShield Connect modules, we recommend scheduling down time for each HSM, upgrading them on a per HSM basis. Each nShield Connect configuration file is specific to an individual HSM and each configuration file should be updated separately to load the new SEE machine.

Chapter 10: Remote Operator

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 unit on page 81](#). 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*) to be securely transmitted and loaded onto another module (an *unattended module*). 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. (You can import or export slots only if the 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 HSMs for Remote Operator.
The HSMs 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 HSM must be in operational mode before they can import or export slots. See [Appendix H: Checking and changing the mode on an nShield Connect on page 277](#) for more about changing the mode.
2. Configure the HSMs for slot import and export, as appropriate.

Note: Dynamic Slots cannot be exported or imported as Remote Operator slots.

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 HSMs for Remote Operator

1. Ensure both HSMs are initialized into the same Security World; see [Adding or restoring an HSM to the Security World on page 111](#).

Note: By default, HSMs are initialized with remote card-set reading enabled. If you do not want an HSM 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 HSM's ID number).

2. Check whether firmware version 2.6.10 or greater is installed in both the attended and unattended HSMs by running the `enquiry` command-line utility. If an earlier version of firmware is installed, you must upgrade the HSM firmware as necessary; see [Upgrading the nShield Connect image file and associated firmware on page 281](#).
 - If you upgrade an HSM's firmware, you must reinitialize it into the Security World, by using the front panel of an nShield Connect, KeySafe, or the `new-world` command-line utility.
3. For the unattended HSM:
 - a. Check whether the Remote Operator feature is enabled by running the `enquiry` command-line utility. The output for the HSM must include `Remote Share` in its list of Features.
 - b. Check whether the HSM has permission to allow loading of Remote OCSs by selecting **Security World mgmt > Display World info**. The output from this selection 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
  
```

This information can also be output by running the `nfkminfo` command-line utility.

Configuring slot import and export

Before you can configure slot import and export, ensure that you have configured the attended and unattended HSMs for Remote Operator as described in [Configuring HSMs for Remote Operator on page 188](#).

Ensure that your network firewall settings are correct. See the *Installation Guide* for more about firewall settings.

Configuring slot import and export using the nShield Connect front panel

When the HSMs have been configured, follow these steps to configure slot import and export:

1. Configure the attended HSM to export a slot by following these steps:
 - a. From the main menu, select **Security World mgmt > Set up remote slots > Export slot**.
 - b. Specify the HSM to which the slot is being export by supplying values for:
 - The IP address of the unattended HSM
 - The ESN of the unattended HSM.
2. Configure the unattended HSM to import the slot that you are exporting from the attended HSM by following these steps:
 - a. From the main menu, select **Security World mgmt > Set up remote slots > Import slot**.
 - b. Specify the details of the Remote Operator slot by supplying values for:
 - The IP address of the HSM from which the slot is being exported
 - The ESN of the HSM from which the slot is being exported
 - The ID of the slot on the importing HSM
 - The port to use to connect to the hardserver hosting the attended HSM.

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 Operator 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 HSM in the usual way.

For the application to be able to load the OCS onto the unattended HSM, 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 69](#)).

The same applies for any keys that an application on an unattended HSM needs to load but that were not generated on that machine.

Creating OCSs and keys for Remote Operator

When you have configured the HSMs and slot import and export, 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 HSM.

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|-remote|y_readable` option specified. The cards in a Remote OCS must be created as persistent; see [Persistent Operator Card Sets on page 127](#).

To check whether the card in a slot is from a Remote OCS, select **Security World mgmt > Display World info** from the main menu or 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: 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 HSMs 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

Once configured, the Remote Operator slots can be used by all the standard nShield libraries. A Remote Operator 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 [Application interfaces on page 158](#). For more information about Remote Operator slots, see [Remote Operator on page 187](#).

Note: After an OCS has been inserted into a Remote Operator 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 [Working with keys on page 192](#).

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 HSMs and slot import and export, 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.

Managing Remote Operator slots using the unit front panel

Editing Remote Operator slots

You can change the details of a Remote Operator slot. You must always update the details of both the exported slot on the local module and the imported slot on the remote module.

To update an exported slot on the module:

1. From the main menu, select **Security World mgmt > Set up remote slots > Edit exported slot**.
2. Select the exported slot that you want to update. Slots are identified by the IP address of the remote module.
3. Update the details of the slot.

To update an imported slot on the unit:

1. From the main menu, select **Security World mgmt > Set up remote slots > Edit imported slot**.
2. Select the imported slot that you want to update. Slots are identified by the IP address of the remote module.
3. Update the details of the slot.

Deleting Remote Operator slots

You can delete Remote Operator slots.

To delete an exported slot, from the main menu, select **Security World mgmt > Set up remote slots > Delete exported slot** and select the slot you want to delete.

To delete an imported slot, from the main menu, select **Security World mgmt > Set up remote slots > Delete imported slot** and select the slot you want to delete.

Chapter 11: 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
- The unit front panel.

You cannot generate keys from the front panel on the unit. You can generate keys on the client using the methods described in this chapter and view them on the module.

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 using the command line

Keys are generated using 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 using 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 270](#) in [Key generation options and parameters on page 270](#).
- 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 271](#) in [Key generation options and parameters on page 270](#).

For details of the available application types (`APPNAME`) and parameters that control other key properties (`NAME=VALUE`), see [Key generation options and parameters on page 270](#).

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. 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 [Using KeySafe on page 201](#).)
2. Click the **Keys** menu button, or select **Keys** from the **Manage** menu. KeySafe takes you to the **Keys** panel, which shows the keys in the security world.
3. Click the **Create** button to open 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.
6. When you have supplied your desired key generation parameters, click the **Commit** 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.

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, which must be consistent with regulatory requirements, to protect your NVRAM-stored keys. Do *NOT* use Remote Administration to back-up keys to a smart card, as the keys would no longer be stored solely within the physical boundary of the HSM.

Note: An NVRAM-stored key can only be loaded successfully by using the `preload` 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 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
<code>--import</code>	This option specifies key importation.
<code>OPTIONS</code>	You can specify particular options when running <code>generatekey</code> that control details of key importation.
<code>APPNAME</code>	This option specifies the name of the application for which the key is to be imported. This must be an application for which <code>generatekey</code> can generate keys.
<code>NAME=VALUE</code>	This specifies a list of parameters for the application.

For RSA keys, you can include `pemreadfile=filename` in the command 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 [Using KeySafe on page 201](#).)
2. Click the **Keys** menu button, or select **Keys** from the **Manage** menu. KeySafe takes you to the **Keys** panel.
3. Click **Import** to open 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.

6. When you have supplied parameters for the key that you want to import, click the **Commit** 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
<code>--retarget</code>	This option specifies key importation.
<code>OPTIONS</code>	This option specifies any options to include when the command is run. Run the command <code>generatekey --help</code> for details about the available options.

Option	Description
<code>APPNAME</code>	This option specifies the name of the application for which the key is to be generated. This must be an application for which <code>generatekey</code> can generate keys.
<code>from-application=appname</code>	This option specifies the name of the application with which the key is currently associated.
<code>from-ident=keyident</code>	This option specifies the identifier of the key to be retargeted. You can find this identifier by using the <code>nfkminfo</code> 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, the unit front panel, or the `nfkminfo` command-line utility.

Viewing information about keys on the unit front panel

You can view keys that have been created on the client on the same computer as the RFS with SEE machines. You cannot view other keys until they are transferred to the RFS.

To view keys:

1. From the main menu, select **Security World mgmt > Keys > List keys**.
2. Select the application to which the key belongs.
3. Select a key to view its full details.
4. If you wish, select **Verify key ACLs** to verify the key's ACL.

Viewing keys with KeySafe

In order to view a list of keys on the client computer on which you are running KeySafe, follow these steps:

1. Start KeySafe. (For an introduction to KeySafe and for information on starting the software, see [Using KeySafe on page 201](#).)
2. Click the **Keys** menu button, or select **Keys** from the **Manage** menu. KeySafe takes you to the **Keys** panel, which lists all the keys in the security world on this client computer. 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 **Keys** panel, you can choose to:

- Create a new key (see [Generating keys with KeySafe on page 194](#))
- import a key (see [Importing keys with KeySafe on page 196](#))
- discard a key from the security world (see [Discarding keys on page 200](#))

Viewing keys using the command line

The `nfkminfo` command-line utility is used to list keys. 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 `nfkm` 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 '

```

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

KeySafe provides a GUI based interface to perform many of the main tasks required to use an nCipher security world. 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.

Setting up KeySafe

1. You must have Java JRE/JDK 1.6, 1.7 or 1.8. 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://www.oracle.com/technetwork/java/>. If your security policy does not allow the use of downloaded software, these components can be obtained on removable media from Oracle 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.

Note: After you have set up the path, check that you are using the correct Java version by running `java` with the `-version` option.

Example:

```
>>java -version
java version "1.8.0_05"
Java(TM) SE Runtime Environment (build 1.8.0_05-b13)
Java HotSpot(TM) 64-Bit Server VM (build 25.5-b02, mixed mode)
```

2. The Security World Software must be installed.
3. In the configuration file at `%NFAST_KMDATA%\config\config`, set the following port values in the `server startup` section:

```
nonpriv_port=9000
priv_port=9001
```

You must restart the hardserver after this change.

Note: See the Installation Guide for more about ports and firewall settings.

Starting KeySafe

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).

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 Security World status pane (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).

Note: The options listed below are also available from the **Manage** menu.

Menu buttons

There are five menu buttons at the top of the sidebar:

Menu button	Description
Introduction	Clicking the Introduction menu button opens the introductory panel that KeySafe displays at startup.
World	<p>Clicking the World menu button opens the World Operations panel, from which you can:</p> <ul style="list-style-type: none"> • Initialize a Security World • Add modules to a Security World • Remove modules from a Security World. <p>Note: You cannot perform these operations on a module that is not in the pre-initialization mode.</p>

Menu button	Description
Card Sets	<p>Clicking the Card Sets menu button opens the List Operator Card Sets panel, from which you can:</p> <ul style="list-style-type: none"> • Examine or change an Operator Card Set or its pass phrase • Create a new Operator Card Set • Replace an Operator or Administrator Card Set • Discard an Operator Card Set.
Softcards	<p>Clicking the Softcards menu button opens the List Softcards panel, from which you can:</p> <ul style="list-style-type: none"> • Create a new softcard • Change or recover the pass phrase on a softcard • Discard a softcard
Keys	<p>Clicking the Keys menu button opens the Keys panel, from which you can:</p> <ul style="list-style-type: none"> • Create a key • Import a key • Discard a key • View details of a key.

While KeySafe is executing a command, the menu buttons are disabled. Their normal functionality returns when the command is completed.

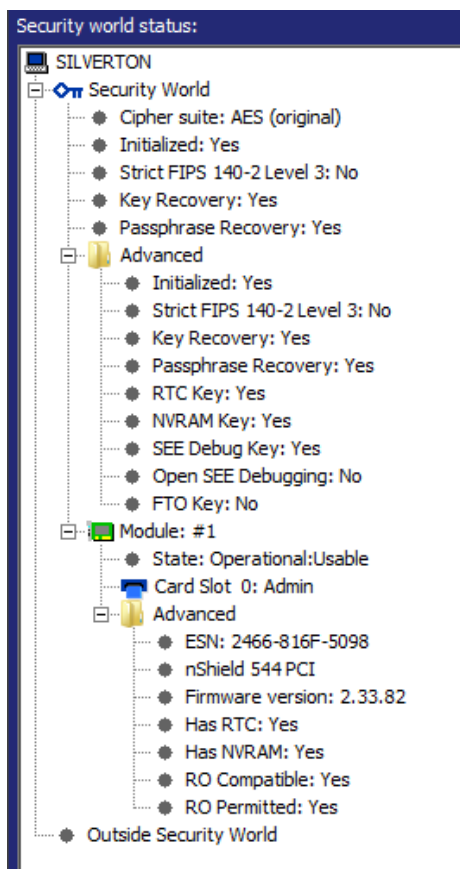
Menus

Three menu options are available from the menu bar at the top of the screen:

- **File**
 - **Exit** - displays a dialog asking whether you are sure you wish to quit KeySafe. Click **Yes** (or press the `Enter` key) to close KeySafe. Click **No** to close the dialog and return to your KeySafe session.
- **Manage**
 - **I**ntroduction - opens the **Introduction** panel. See [Introduction on page 202](#).
 - **W**orld - opens the **World Operations** panel. See [World on page 202](#).
 - **C**ard sets - opens the **List Operator Card Sets** panel. See [Card Sets on page 203](#).
 - **S**oftcards - opens the **List Softcards** panel. See [Softcards on page 203](#).
 - **K**ey - opens the **Keys** panel. See [Keys on page 203](#).
- **Help**
 - **A**bout KeySafe - opens the **About KeySafe** panel, which displays current version numbers for KeySafe, kmjava and nfjava. You will need to quote these version numbers if you contact Support about KeySafe.

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 shown 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, or node. By default, when KeySafe starts, these nodes are collapsed and show a minus sign. Click the node to display expanded information about the Security World or module. Click the node again to collapse this information.

Security World information

At the top level of the Security World tree, the following information is displayed:

- **Cipher suite** — the type of key protecting the Security World
- **Initialized** — whether the Security World is initialized (Yes or No)
- **Strict FIPS 140-2 Level 3** — whether the Security World is operating at FIPS 140-2 level 3 (Yes or No)
- **Key Recovery** — whether key recovery is enabled (Yes or No)
- **Passphrase Recovery** — whether passphrase recovery is enabled (Yes or No). For more information, see [Pass phrase replacement on page 96](#)

When the **Advanced** node is expanded, the following additional information is displayed:

- **RTC Key** — whether a real-time clock authorization key has been generated (Yes or No)
- **NVRAM Key** — whether a non-volatile memory authorization key has been generated (Yes or No)
- **SEE Debug Key** — whether SEE debugging has been enabled (Yes or No)

- **Open SEE Debugging** — whether Open SEE debugging has been enabled (Yes or No)
- **FTO Key** — whether a foreign token key has been generated (Yes or No)

Module information

Module information may be displayed either inside or outside the Security World. Modules that have not been incorporated into a Security World will be shown beneath the **Outside Security World** node.

At the top level of the Module tree, the following information is displayed:

- The module's state, which is one of the following:

Mode	Description
PreInitMode	The module is in pre-initialization mode.
InitMode	The module is in initialization mode.
Operational:Useable	The module is in the current Security World and useable for key operations.
Operational:Unknown	The mode of the module cannot be determined.
Operational:Uninitialized	The module key is set and the module must be initialized before use.
Operational:Factory	The module key is set to the factory default.
Operational:Foreign	The module is from an unknown Security World.
Operational:AccelOnly	The module is an acceleration-only module.
Operational:Unchecked	Although the module appears to be in the current Security World, KeySafe cannot find a module initialization certificate (a module_ESN file) for this module
Failed	The module has failed.
PreMaintMode	The module is in the pre-maintenance mode.
MaintMode	The module is in the maintenance mode.

- the status of the smart card reader slot(s).
 - Note:** For 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:

- **ESN** — 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 **HSM type** and model number
- **Firmware version** — the version of the module's firmware
- **Has RTC** — whether the module has a Real Time Clock (RTC)
- **Has NVRAM** — whether the module has nonvolatile memory (NVRAM).
- **RO Compatible** —
- **RO Permitted** —

Main panel area

The KeySafe main panel area is used to display information and options pertaining to a chosen operation. For example, clicking the **World** menu button takes you to the **World Operations** panel in the main panel area.

Navigation and command buttons

On each **Operations** panel there are a number of *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 **World Operations** panel, for example, clicking the **Erase Module** navigation button does not itself erase a module, but rather loads the **Erase Module** panel.

On the next panel, the **Commit** button executes an operation, while the **Back** button returns to the previous panel. For example, on the **Erase Module** panel, clicking the **Commit** button will erase the module, while clicking the **Back** button will return to the **World Operations** panel.



Clicking the **Commit** button tells KeySafe to write or delete data: it is not necessarily possible to reverse such changes even if you subsequently cancel the operation. In some cases, clicking the **Commit** 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 you select options by means of radio buttons or that you enter data into text fields before clicking the **Commit** button. For example, if you click the **Reprogram Module** button on the **World Operations** panel, the next panel prompts you to choose whether the module can receive remote operator card shares.

Input may be in the form of radio buttons (allowing several options, one of which — the *default* — will be already selected) or text boxes (allowing you to enter text: no default value is set). If you click the **Commit** 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 the **Commit** button, the menu buttons are disabled until the requested command is completed.

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.

Error message	Possible causes	Suggested solutions
Unable to establish KeySafe session. Please ensure that the hardserver is running and accepting TCP connections. Click OK to exit.	<ul style="list-style-type: none"> The hardserver is unable to receive TCP connections. The server program communicates with clients by using named pipes or TCP sockets. The hardserver is not running, or is physically disconnected. 	<ul style="list-style-type: none"> Check the hardserver configuration file settings: see server_startup on page 260. To restart the hardserver: <ol style="list-style-type: none"> Exit KeySafe Restart the server (as described in Stopping and restarting the client hardserver on page 89) Restart KeySafe.
Unable to generate key: Error reported by nShield hardware module nFast error: UnknownFlag, in response to GenerateKeyPair	Your hardware or firmware may not be up to date.	<p>To update your firmware:</p> <ol style="list-style-type: none"> Exit KeySafe Update the firmware as described in Upgrading the nShield Connect image file and associated firmware on page 281 Restart KeySafe <p>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 <code>kmdata</code> directory must be implemented.</p>

if you receive any error message titled **Unexpected Error**, contact Support with details of what you were doing, and the exact error message.

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...
<code>enquiry</code>	<p>Obtain information about the hardserver (Security World Software server) and the modules connected to it.</p> <p>Use this utility to:</p> <ul style="list-style-type: none"> • Check if the software has been installed correctly • Check the firmware version • Check if the Remote Operator feature is enabled • Check the hardware status of internal security modules <p>See Testing the installation on page 90 for more information.</p>
<code>checkmod</code>	Check modulo exponentiations performed on the module against the test data located in the <code>opt/nfast/testdata</code> directory.
<code>cfg-mkdefault</code>	Create a default client configuration file for the hardserver configuration sections.
<code>cfg-reread</code>	Load the hardserver configuration from the configuration file.

Utility	Enables you to...
fet	<ul style="list-style-type: none"> • Activate features • View the status of features • Verify that a feature has been successfully enabled on a connected module <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>Note: To enable features, and view the status of or verify features on an nShield Connect unit, use the front panel rather than the fet utility.</p> <p>For more information, see Enabling features with a smart card on page 87.</p>
ncdate	View, set, and update the time on a module's real-time clock.
ncversions	<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"> • Versions of all components, irrespective of whether they are installed individually or as part of a component bundle • Version of each component bundle
nfdiag	<p>Obtain information about the module and the host on which it is installed. This diagnostic utility can save information to either a ZIP file or a text file.</p> <p>For more information, see nfdiag: diagnostics utility on page 228.</p> <p>Note: Run this utility only if requested to do so by Support.</p>
nopclearfail	Clear an HSM, put an HSM into the error state, retry a failed HSM, or change the HSM mode.
nvrām-backup	Copy files between a module's NVRAM and a smart card, allowing files to be backed up and restored.
nvrām-sw	View and modify information about NVRAM areas.
pubkey-find	Obtain information of the public key from a certificate or certificate request (in a Base-64 encoded PEM file).
randchk	Run a universal statistical test on random numbers returned by the module.
rtc	View and set the module's real-time clock.
slotinfo	<ul style="list-style-type: none"> • Obtain information about tokens in a module • Format a smart card

Utility	Enables you to...
snmpbulkwalk snmpget snmpgetnext snmpwalk snmpset snmpstat snmptranslate snmpwalk	Obtain system, module, connection and software information from the SNMP agent. For more information, see Using the SNMP command-line utilities on page 317
stattree	Obtain statistics gathered by the Security World Software server and modules. For more information, see stattree: information utility on page 243 .

Hardware utilities

Use the following utilities to manage the firmware installed on an nShield HSM.

Utility	Enables you to...
fwcheck	Verify the firmware installed on a module.
nfloadmon	Upgrade the module monitor and firmware of nShield Edge and nShield Solo modules.

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 FIPS 140-2 level 2 Security Worlds.

Utility	Enables you to...
cryptest	Test all defined symmetric cryptographic mechanisms.
des_kat	Perform DES known-answer tests. This utility indicates if any of them fail.
floodtest	Perform hardware speed-testing by using modular exponentiation.
kptest	Test the consistency of encryption and decryption, or of signature and verification, with the RSA and DSA algorithms.
ncthread-test	Stress test modules and test nCore API concurrent connection support.
perfcheck	Run various tests to measure the cryptographic performance of a module. For more information, see perfcheck: performance measurement checking tool on page 240 .
sigtest	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...
<code>bulkerase</code>	<p>Erase multiple smart cards including Administrator Cards, Operator Cards, and FEM activation cards, in the same session.</p> <p>Note: Do not use the <code>bulkerase</code> utility to erase Administrator Cards from the current Security World.</p>
<code>cardpp</code>	<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"> • Verifying the pass phrase of a card with <code>cardpp</code> on page 140. • Changing known pass phrase with <code>cardpp</code> on page 142. • Changing unknown or lost pass phrase on page 144.
<code>createocs</code>	<p>Create and erase an OCS.</p> <p>For more information, see:</p> <ul style="list-style-type: none"> • Creating an Operator Card Set using the command line on page 128. • Erasing cards using the command line on page 135.
<code>initunit</code>	<p>Initialize an nShield module.</p> <p>For more information, see Erasing a module with <code>initunit</code> on page 124.</p>
<code>generatekey</code>	<p>Generate, import, or retarget keys. This utility is included in the <code>core Tools</code> bundle, which contains all the Security World Software utilities.</p> <p>For more information, see:</p> <ul style="list-style-type: none"> • Generating keys using the command line on page 192. • Importing keys from the command line on page 195. • Example of key generation with <code>generatekey</code> on page 193, for an example of key generation in batch mode. • Example of key importation with <code>generatekey</code> on page 196, for an example of importing an RSA key. • Listing supported applications with <code>generatekey</code> on page 197. • Retargeting keys with <code>generatekey</code> on page 197.

Utility	Enables you to...
<code>key-xfer-im</code>	<p>Transfer keys between Security Worlds, when used in conjunction with the <code>mk-reprogram</code> utility. During the key transfer process, the <code>key-xfer-im</code> utility imports the module key that has been programmed into a new Security World.</p> <p>For more information, see Transferring keys between Security Worlds on page 112.</p> <p>Note: You can also use the <code>migrate-world</code> utility to transfer keys between Security Worlds.</p>
<code>kmfile-dump</code>	Obtain key management information from a Security World's key management data file.
<code>migrate-world</code>	<p>Migrate existing keys to a destination Security World.</p> <p>For more information, see Security World migration on page 116.</p>
<code>mkac1x</code>	<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 <code>generate-key</code> utility.</p> <p>Note: Ensure that you run the <code>mkac1x</code> 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>
<code>mk-reprogram</code>	<p>Transfer keys between Security Worlds, when used in conjunction with the <code>key-xfer-im</code> utility. During the key transfer process, the <code>mk-reprogram</code> utility programs the module key of a Security World to another Security World.</p> <p>For more information, see Transferring keys between Security Worlds on page 112.</p> <p>Note: You can also use the <code>migrate-world</code> utility to transfer keys between Security Worlds.</p>
<code>new-world</code>	Create and manage Security Worlds on nShield modules.
<code>nfkmcheck</code>	Check Security World data for consistency.
<code>nfkminfo</code>	<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"> • Displaying information about a Security World with <code>nfkminfo</code> on page 110. • Viewing card sets using the command line on page 138. • Viewing softcards with <code>nfkminfo</code> on page 139. • Viewing keys using the command line on page 199. • <code>nfkminfo</code>: information utility on page 230.
<code>nfkmverify</code>	Perform Security World verification.

Utility	Enables you to...
<code>postrocs</code>	<p>Transfer PKCS #11 keys to a new card set in the new Security World. When transferring keys by using either the <code>key-xfer-im</code> utility or the <code>migrate-world</code> utility, run the <code>postrocs</code> utility if there are any PKCS #11 keys that are protected by OCSs.</p> <p>Note: PKCS #11 keys either have <code>keys_pkcs_um</code> or <code>key_pkcs_uc</code> as the prefix.</p>
<code>ppmk</code>	<ul style="list-style-type: none"> • Create and manage softcards. Use this utility to: • View details of a softcard • Create and delete a softcard • View, change, and recover the pass phrase of a softcard <p>For more information, see:</p> <ul style="list-style-type: none"> • Creating a softcard with ppmk on page 132. • Erasing softcards with ppmk on page 136. • Viewing softcards with ppmk on page 139. • Verifying the pass phrase of a softcard with ppmk on page 141. • Changing known softcard pass phrase with ppmk on page 143. • Replacing unknown pass phrase with ppmk on page 144.
<code>preload</code>	<p>Load keys into a module before an application is run in another session.</p> <p>For more information, see Transferring keys between Security Worlds on page 112.</p>
<code>racs</code>	<p>Create a new ACS to replace an existing ACS.</p> <p>For more information, see Replacing an Administrator Card Set with racs on page 156.</p>
<code>rocs</code>	<ul style="list-style-type: none"> • Restore an OCS from a quorum of its cards • Restore softcards <p>For more information, see:</p> <ul style="list-style-type: none"> • Replacing OCSs or softcards with rocs on page 147. • Using rocs from the command line on page 152.

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...
<code>elftool</code>	Convert ELF format executables into a format suitable for loading as an SEE machine.

Utility	Enables you to...
<code>hsc_loadseemachine</code>	Load an SEE machine into each module that is configured to receive one, then publishes a newly created SEE World, if appropriate.
<code>loadsee-setup</code>	Set up the configuration of auto-loaded SEE machines.
<code>modstate</code>	View the signed module state.
<code>see-sock-serv</code>	
<code>see-stdioe-serv</code>	Activate or enable standard IO and socket connections for SEE machines using the <code>bsdlib</code> architecture.
<code>see-stdioesock-serv</code>	
<code>see-stdoe-serv</code>	
<code>tct2</code> (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...
<code>ckcerttool</code>	Import a certificate as a PKCS #11 <code>CKO_CERTIFICATE</code> object of type <code>CKC_X_509</code> , and optionally, associate it with the corresponding private key.
<code>ckcheckinst</code>	Verify the installation of the nCipher PKCS #11 libraries. For more information, see Checking the installation of the nCipher PKCS #11 library on page 182 .
<code>ckimportbackend</code>	Generate keys for use with PKCS #11 applications. When you run the <code>generatekey</code> utility to generate PKCS #11 keys, the <code>ckimportbackend</code> utility is executed in the background. Note: Do not run this utility unless directed to do so by Support.
<code>cknfkmid</code>	View values of attributes of PKCS #11 objects.
<code>ckshahmac</code>	Perform a PKCS #11 test for vendor-defined <code>SHA1_HMAC</code> key signing and verification capabilities.
<code>cksigtest</code>	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...
<code>ckinfo</code>	View PKCS #11 library, slot and token information. Use this utility to verify that the library is functioning correctly.

Utility	Enables you to...
<code>cklist</code>	View details of objects on all slots. If invoked with a PIN argument, the utility lists public and private objects. If invoked with the <code>-n</code> (<code>--nopin</code>) option, the utility lists only the public objects. This utility does not output any potentially sensitive attributes, even if the object has <code>CKA_SENSITIVE</code> set to <code>FALSE</code> .
<code>ckmechinfo</code>	View details of the supported PKCS #11 mechanisms provided by the module.
<code>ckrsagen</code>	Test RSA key generation. You can use specific PKCS #11 attributes for generating RSA keys.

nShield Connect utilities

The utilities described in this section are used with nShield Connect only. Use these utilities to:

- Create and manage client configuration files.
- Enroll nTokens with an nShield Connect.
- Set up a Remote File System (RFS) and synchronize Security World data between an nShield Connect and the RFS.
- Administer an nShield Connect without using the front panel

Utility	Enables you to...
<code>anonkneti</code>	View the ESN and <code>HK_{NETI}</code> key hash from a module identified by its IP address. For more information, see Configuring the remote file system (RFS) on page 57 .
<code>cfg-pushnethsm</code>	Copy a specified configuration file from a remote file system to the file system on a specified module. For more information, see: <ul style="list-style-type: none"> • Remote configuration of additional clients on page 64. • Configuring the nShield Connect with configuration files on page 68.
<code>config-serverstartup</code>	Edit the <code>[server_startup]</code> section of the configuration file for the client's hardserver to enable or disable TCP sockets. For more information, see: <ul style="list-style-type: none"> • After software installation on page 52. • config-serverstartup on page 67.

Utility	Enables you to...
nethsmadmin	<p>Administer an nShield Connect without using the front panel. Options include:</p> <ul style="list-style-type: none"> • Check the Security World files on a specified nShield Connect • Copy Security World files from the RFS to the nShield Connect • Command the specified nShield Connect to reboot • Command the nShield Connect to upgrade using the specified nShield Connect image file from its RFS • Retrieve a list of nShield Connect image files available on the RFS • Retrieve a list of feature certificates available on the RFS for a specified nShield Connect • Command the nShield Connect to apply a specified feature certificate from the RFS <p>For more information, see:</p> <ul style="list-style-type: none"> • <i>Using nethsmadmin to copy a Security World to an nShield Connect and check the current version on page 100.</i> • <i>Upgrading the nShield Connect image file and firmware from a privileged client on page 284</i> • <i>Using nethsmadmin to reboot an nShield Connect on page 79</i> • <i>Remotely enabling dynamic feature certificates including nShield Connect client licenses on page 282</i>
nethsmenroll	<p>Edit the local hardserver configuration file to add the specified nShield Connect 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.</p> <p>For more information, see:</p> <ul style="list-style-type: none"> • <i>nethsm_imports on page 264.</i> • <i>Configuring the nShield Connect to use the client on page 61.</i> • <i>nethsmenroll on page 67.</i>
ntokenenroll	<p>Enroll a locally attached nToken with an nShield Connect 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.</p> <p>For more information, see <i>Configuring the nShield Connect to use the client on page 61.</i></p>

Utility	Enables you to...
<code>rfs-setup</code>	<p>Create a default RFS hardserver configuration on the client. Run this utility when you first configure a client.</p> <p>For more information, see:</p> <ul style="list-style-type: none"> • Configuring the remote file system (RFS) on page 57. • Setting up client cooperation on page 69.
<code>rfs-sync</code>	<ul style="list-style-type: none"> • Synchronize the Security World data of a client with the RFS, when you run the utility on the client • Synchronize the Security World data of the RFS with the client, when you run the utility on the RFS <p>For more information, see:</p> <ul style="list-style-type: none"> • Setting up client cooperation on page 69. • rfs-sync on page 71. <p>Note: You can use this utility with nShield modules if an nShield Connect unit is present.</p>

Developer-specific utilities

Use the following utilities to ensure that the HSMs are functioning as expected and to test the cryptographic functionality at the nCore level.

Utility	Enables you to...
<code>pollbare</code>	<p>Obtain information about state changes. The functionality of this test utility depends on whether the server or an HSM supports nCore API poll commands.</p> <p>Note: To know if your server or HSM supports nCore API poll commands, run the <code>enquiry</code> utility.</p>
<code>trial</code>	<p>Test the nCore API commands. You can use this utility interactively or from a script file.</p>

Appendix C: Environment variables

This appendix describes the environmental variables used by Security World Software:

Variable	Description
KERNEL_HEADERS	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.
NFAST_CERTDIR	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 NFAST_HOME , NFAST_KMDATA , and NFAST_LOGDIR .
NFAST_DEBUG	This variable enables debug logging for the PKCS #11 library. You must set NFAST_DEBUG equal to a value in the range 1 – 7 for debug messages to be logged (see <i>Logging, debugging, and diagnostics on page 222</i>). For more information, see also <i>Logging and debugging information for PKCS #11 on page 225</i> .
NFAST_DEBUGSYSLOG	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.
NFAST_HOME	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 NFAST_KMDATA , NFAST_CERTDIR , and NFAST_LOGDIR .
NFAST_HWCRHK_HSM_POOL	This variable configures HSM Pool mode for CHIL (Cryptographic Hardware Interface Library). Set the variable to 1, y or Y to enable HSM Pool mode for the CHIL-enabled application, or set to 0, n or N to explicitly disable HSM Pool mode for the CHIL-enabled application. Do not configure this variable with applications using other APIs.

Variable	Description
<code>NFAST_HWCRHK_LOGFILE</code>	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 <code>hwcrhk</code> API, which vary according to the application in use.
<code>NFAST_KMDATA</code>	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 <code>NFAST_HOME</code> , <code>NFAST_CERTDIR</code> , <code>NFAST_LOGDIR</code> , and <code>NFAST_KMLOCAL</code> .
<code>NFAST_KMLOCAL</code>	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 <code>local</code> subdirectory of the Key Management Data directory. See <code>NFAST_KMDATA</code> .
<code>NFAST_LOGDIR</code>	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 <code>NFAST_HOME</code> , <code>NFAST_KMDATA</code> , and <code>NFAST_CERTDIR</code> .
<code>NFAST_NFKM_TOKENSFILE</code> <code>NFAST_NFKM_TOKENSSELECT</code>	This variable sets the default values for a file in which <code>ClientID</code> and <code>KeyIDs</code> are stored by the <code>preload</code> command-line utility.
<code>NFAST_SEE_MACHINEENCKEY_DEFAULT</code>	This variable is the name of the <code>SEEConf</code> key needed to decrypt SEE-machine images. Running the command <code>loadmache --encryptionkey=IDENT</code> (or <code>loadmache --unencrypted</code>) overrides any value set by this variable.
<code>NFAST_SEE_MACHINEENCKEY_module</code>	This variable is the name of the <code>SEEConf</code> key needed to decrypt the SEE-machine image targeted for the specified module. It overrides <code>NFAST_SEE_MACHINEENCKEY_DEFAULT</code> for the specified module. Running the command <code>loadmache --encryptionkey=IDENT</code> (or <code>loadmache --unencrypted</code>) overrides any value set by this variable.
<code>NFAST_SEE_MACHINEIMAGE_DEFAULT</code>	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 <code>machine-filename</code> parameter when running the <code>loadmache</code> command-line utility overrides this variable. This variable is not affected when running the <code>loadsee-setup</code> or <code>hsc_loadseemachine</code> utilities.

Variable	Description
NFAST_SEE_MACHINEIMAGE_module	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 NFAST_SEE_MACHINEIMAGE_DEFAULT for the specified module. Supplying the <i>machine-filename</i> parameter when running the loadmache command-line utility overrides the NFAST_SEE_MACHINEIMAGE_module variable. This variable is not affected when running the loadsee-setup or hsc_loadseemachine utilities.
NFAST_SEE_MACHINESIGHASH_DEFAULT	This variable is the default key hash of the vendor signing key (seeinteg) 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 loadmache --sighash=HASH any value set in this variable.
NFAST_SEE_MACHINESIGHASH_module	This variable is the key hash of the vendor signing key (seeinteg) that signs SEE machine images for the specified module. It overrides NFAST_SEE_MACHINESIGHASH_DEFAULT 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 loadmache --sighash=HASH any value set in this variable.
NFAST_SERVER	If these variables are set in the hardserver's environment, the values specify the pathnames of the TCP sockets that the hardserver uses for ordinary/privileged client connections to the hardserver.
NFAST_PRIVSERVER	These variables are available for this purpose for backward compatibility only; you should configure sockets in the hardserver configuration file, see server_startup on page 260 . If you set these variables they override the values in hardserver configuration file.
NFAST_SERVER_PORT	If these variables are set in the hardserver's environment, the values specify the TCP port numbers that the nFast server uses for connections over TCP sockets.
NFAST_SERVER_PRIVPORT	These variables are available for this purpose for backward compatibility only: you should configure ports in the hardserver configuration file, as described in server_startup on page 260 . If you set these variables, they override the values in the hardserver configuration file.

Variable	Description
NFAST_SERVER_SYSLOG	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 + enables traditional logging and syslog simultaneously.
NFLOG_CATEGORIES	This variable is used to filter log messages by supplying a colon-separated list of allowable message categories; see <i>Logging, debugging, and diagnostics on page 222</i> . If no value is supplied, all message categories are logged.
NFLOG_SEVERITY	This variable is used to filter log messages by supplying a minimum severity level to be logged; see <i>Logging, debugging, and diagnostics on page 222</i> . If no value is supplied, the default severity level is WARNING .
NFLOG_DETAIL	This variable is used to filter log messages by supplying a bitmask of detail flags; see <i>Logging, debugging, and diagnostics on page 222</i> . The default is time+severity+writeable .
NFLOG_FILE	This variable is used to specify a filename (or file descriptor) in which log messages are to be written; see <i>Logging, debugging, and diagnostics on page 222</i> . The default is stderr (the equivalent of file descriptor &2).
OPENSSL_HWCERHK_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 D: 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

The nShield Connect and the applications that use it generate log files. You can view log files using the front panel. Application log messages are handled on the client.

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 **DEBUGN** is only likely to be useful during debugging, and we recommend not setting the severity level to **DEBUGN** 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 256](#)):

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 <code>time_t</code> (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 <code>NFLOG_SEVERITY</code> environment variable) with the log entry. It is on by default.	msg_severity
0x00000200	This flag shows the message category (a category as used by the <code>NFLOG_CATEGORIES</code> 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

Hexadecimal flag	Function	logdetail flags
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 StackPush and StackPop 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 dbg_stub 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 dbg_stubbignum 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 dbg_stubinit 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 dbg_dumpenv flag with the logging functionality found in previous Security World Software releases.)
nfkm-getinfo	Logs nfkm-getinfo messages.
nfkm-newworld	Logs messages about world generation.
nfkm-admin	Logs operations using the Administrator Card Set.
nfkm-kmdata	Logs file operations in the kmdata directory.
nfkm-general	Logs general NFKM library messages.
nfkm-keys	Logs key loading operations.
nfkm-preload	Logs preload operations.
nfkm-ppmk	Logs softcard operations.
serv-general	Logs general messages about the local hardserver.

Category	Description
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 256](#).

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` variable. A debug output file (with path) can also be set using the `CKNFAST_DEBUGFILE` variable. These variables can be set in the environment or `/opt/nfast/cknfastrc` file. Normally settings in the environment should override the same settings (if any) in the `cknfastrc` file. You can specify any appropriate PKCS #11 categories using the `NFLOG_CATEGORIES` environment variable.

To produce PKCS #11 debug output, the `CKNFAST_DEBUG` variable can be a given value from 1 through 11, where the greater the value the more detailed debug information is provided. A value of 7 is a

reasonable compromise between too little and too much debug information. A value of 0 switches the debug output off.

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 "

Debugging information for Java

This section describes how you can specify the debugging information generated by Java.

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 Java debugging with the command line

You can set the Java debug options by immediately preceding them with a `-D`. Use the `NJAVA_DEBUGFILE` property to direct output to a given file name, for example:

```
java -DNFJAVA_DEBUGFILE=myfile -DNFJAVA_DEBUG=129 -classpath ... classname
```

Note: Do not set `NFJAVA_DEBUG` or `NFJAVA_DEBUGFILE` in the environment because Java does not pick up variables from the environment.

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 an nCipher HSM. 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 nCipher 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.

Option	Actions to take
any additional diagnostic file(s)	<p>By default, nfdiag asks if you want to supply any additional diagnostic files. If you do not want to supply additional diagnostic files, type <code>N</code>, or <code>Enter</code> to continue.</p> <p>If you want to supply additional diagnostic files:</p> <ul style="list-style-type: none"> • Type <code>Y</code>. • When prompted, supply the name of the file to be attached. Attached files must be in plain text format. • After you have supplied the name of a file to attach, nfdiag asks again if you want to supply any additional diagnostic files. Repeat the process to attach additional files. • When you do not want to attach any more files, type <code>N</code>, or <code>Enter</code> to continue.

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.

nShield Connect tamper log

The nShield Connect tamper log contains:

- The date and time of any tamper events
- Whether the tamper detection functionality has been disabled or re-enabled.

You view the tamper log using the nShield Connect front panel, by selecting **System > System information > View tamper log**. You cannot erase the tamper log.

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 `nfmkinfo` 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
hkns0      hash_kns0
hkm         hash_km
hkmwk      hash_knwk
hkrc       hash_krc
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     LTNS0(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 HSM in this Security World on this host.

!Usable

This indicates that there are no usable HSMs 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 HSMs 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 HSMs that use Security Worlds and is not secret).

hkre

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 HSM in the Security World, `nfmkinfo` reports:

generation

This indicates the version of the HSM data.

state

This indicates one of the following:

PreInitMode

This indicates that the HSM is in the pre-initialization state.

InitMode

This indicates that the HSM is in the initialization state.

Unknown

This indicates that the HSM's state could not be determined.

Usable

This indicates that the HSM is programmed in the current Security World and can be used.

Uninitialized

This indicates that the HSM does not have the Security Officer's key set and that the HSM must be initialized before use.

Factory

This indicates that the HSM has module key zero only and that the Security Officer's key is set to the factory default.

Foreign

This indicates that the HSM is from an unknown Security World.

AccelOnly

This indicates that the HSM is acceleration only.

Unchecked

This indicates that, although the HSM appears to be in the current Security World, `nfkminfo` could not find a module initialization certificate (a `module_ESN` file) for this HSM.

Failed

This indicates that the HSM has failed.

For internal security modules running firmware 2.61.2 and above, use the `enquiry` utility for further information about the failure reason, or look for hardware errors in the `hardserver` log.

MaintMode

This indicates that the HSM is in the maintenance state.

flags

This displays `ShareTarget` if the HSM has been initialized to allow reading of remote card sets.

n_slots

This indicates the number of slots on the HSM (there is one slot for each physical smart card reader, one slot for each soft token, one slot for each available Remote Operator slot and one slot for each associated Dynamic Slots).

esn

This indicates the electronic serial number of the HSM (if the HSM is not in the `Usable` state, the electronic serial number may not be available).

hkm1

This indicates the hash of the HSM signing key (if the HSM is not in the `Usable` state, this value may not be available).

Slot

For each slot on the HSM, `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. Single letters in parentheses are the flag codes reported by the `s1otinfo` utility:

0x2

(A) SupportsAuthentication

This indicates that the slot supports token-level challenge-response authentication.

0x40000

(R) RemoteSlot

This indicates that the slot is a Remote Operator slot that has been imported from a remote HSM.

0x80000

(D) DynamicSlot

This indicates that it is a Dynamic Slot.

0x100000

(a) Associated

This indicates that a nShield Remote Administration Client has associated a card reader with this Dynamic Slot.

0x200000

(t) TimedOut

This indicates that no response has been received from the smartcard in this Dynamic Slot within the configured timeout.

0x400000

(f) SecureChannelFailed

This indicates that the secure channel between the HSM and the smartcard in this Dynamic Slot has failed in some way.

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_{NSO} , 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 HSM 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.

hk1tu

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 nCipher HSM.

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
Reps	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 <code>--repetitions=REPS</code> option.

Value	Description
Total time	This value shows the total time in seconds taken to run all repetitions of the test.
Mean time	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 Total time by the value of Reps).
Std dev	This value shows the standard deviation in the Mean time . 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 perfcheck with the --no-std-dev option; in some cases, setting --no-std-dev can increase throughput.
Latency	This value is a measure of how long in milliseconds a single command takes to travel from the host computer to the HSM and back to the host computer again. This includes the time the command spends going over the network.
Rate	This value is a measure of throughput. It is calculated by dividing the value of Reps by the value of Total time , and displayed in terms of a unit appropriate to the type of test (for example, for signing tests, the number of signatures performed each second).

How perfcheck calculates statistics

When an nCore command is submitted to an HSM by a client application, it is processed as follows:

1. The command is passed to the client hardserver.
2. The client hardserver encrypts the command.
3. The encrypted command is sent to the unit hardserver over the network.
4. The unit hardserver decrypts the command and queues it.
5. When the internal security module is free, the command is submitted from the hardserver queue.
6. The command is executed by the HSM, and the reply is given to the unit hardserver.
7. The unit hardserver encrypts the command.
8. The unit hardserver sends the command back to the client hardserver over the network.
9. The client hardserver decrypts the reply and queues it.
10. When the client application is ready, the queued reply is returned to it.

Because an HSM can execute several commands at once, throughput is maximized by ensuring there is always at least one command in the unit hardserver queue (so that there are always commands available to give to the HSM).

The **perfcheck** utility sends multiple simultaneous nCore commands to keep the HSM 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 HSMs.

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 HSM.

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
```

`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 HSM:

Category	Contains
ModuleJobStats	This tag holds statistics for the Security World Software commands (jobs) processed by this HSM.
ModulePCIStats	This tag does not apply to an nShield Edge. This tag holds statistics for the PCI connection between the HSM and the host computer.
ServerGlobals	Aggregate statistics for all commands processed by the hardserver since it started. The standard statistics (as described below) apply to the commands sent from the hardserver to HSMs. Commands processed internally by the server are not included here. The Uptime statistic gives the total running time of the server so far.
Connections	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 Information: New client #24 connected , the client's statistics appear under node #24 in the <i>stattree</i> output.
PerModule	Statistics kept by the HSMs. There is one instance node for each HSM, numbered using the standard HSM numbering. The statistics provided by each HSM depend on the HSM type and firmware version.

Category	Contains
ModuleObjStats	Statistics for the HSM's Object Store, which contains keys and other resources. These statistics may be useful in debugging applications that leak key handles, for example.
ModuleEnvStats	General statistics for the HSM's operating environment.
HostEnvStats	Environmental statistics for the HSM.
HostSysInfo	Further statistics for the HSM.

Statistics IDs

ID	Value
Uptime	The length of time (in seconds) since an HSM was last reset, the hardserver was started, or a client connection was made.
CmdCount	The total number of commands sent for processing from a client to the server, or from the server to an HSM. Contains the number of commands currently being processed.
ReplyCount	The total number of replies returned from server to client, or from HSM to server.
CmdBytes	The total length of all the command blocks sent for processing.
ReplyBytes	The total length of all the reply blocks received after completion.
CmdMarshalErrors	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 HSM that the HSM does not understand), or that the data transfer mechanism is faulty.
ReplyMarshalErrors	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 HSM that the HSM does not understand), or that the data transfer mechanism is faulty.
ClientCount	The number of client connections currently made to the server. This appears in the hardserver statistics.
MaxClients	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.
DeviceFails	The number of times the hardserver has declared a device to have failed. The hardserver provides a diagnostic message when this occurs.
DeviceRestarts	The number of times the hardserver has attempted to restart an HSM after it has failed. The hardserver provides a Notice message when this occurs. The message does not indicate that the attempt was successful.

ID	Value
QOutstanding	The number of commands waiting for an HSM to become available on the specified client connection. When an HSM accepts a command from a client, this number decreases by 1 and DevOutstanding increases by 1. Commands that are processed purely by the server are never included in this count.
DevOutstanding	The number of commands sent by the specified client that are currently executing on one or more HSMs. When an HSM accepts a command from a client, this number decreases by 1 and QOutstanding increases by 1. Commands that are processed purely by the server are never included in this count.
LongOutstanding	The number of LongJobs sent by the specified client that are currently executing on one or more HSMs. When an HSM accepts a LongJobs command from a client, this number increases by 1 and QOutstanding decreases by 1. Commands that are processed purely by the server are never included in this count.
RemoteIPAddress	The remote IP address of a client who has this connection. A local client has the address 0.0.0.0.
HostWriteCount	The number of write operations (used to submit new commands) that have been received by the HSM from the host machine. One write operation may contain more than one command block. The operation is most efficient when this is the case.
HostWriteErrors	The number of times the HSM 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 HSM's interface.
HostWriteBadData	Not currently reported by the HSM. Attempts to write bad data to the HSM are reflected in HostWriteErrors .
HostWriteOverruns	Not currently reported by the HSM. Write overruns are reflected in HostWriteErrors .
HostWriteNoMemory	Not currently reported by the HSM. Write failures due to a lack of memory are reflected in HostWriteErrors .
HostReadCount	The number of times a read operation to the HSM was attempted. The HSM can defer a read if it has no replies at the time, but expects some to be available later. Typically the HSM reports HostReadCount in two places: the number under ModuleJobStats counts a deferred read twice, once when it is initially deferred, and once when it finally returns some data. The number under ModulePCISStats counts this as one operation.
HostReadErrors	The number of times a read to an HSM 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.
HostReadEmpty	The number of times a read from the HSM returned no data because there were no commands waiting for completion. In general, this only happens infrequently during HSM startup or reset. It can also happen if PauseForNotifications is disabled.
HostReadUnderruns	Not currently reported by the HSM.

ID	Value
HostReadDeferred	The number of times a read operation to the HSM was suspended because it was waiting for more replies to become available. When the HSM is working at full capacity, a sizeable proportion of the total reads are likely to be deferred.
HostReadTerminated	The number of times an HSM had to cancel a read operation which has been deferred. This normally happens only if the clear key is pressed while the HSM is executing commands. Otherwise it might indicate a device driver, interface, or firmware problem.
PFNIssued	The number of PauseForNotifications commands accepted by the HSM 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.
PFNRejected	The number of PauseForNotifications commands rejected by the HSM when received from the hardserver. This can happen during HSM startup or reset, but not in normal use. It indicates a hardserver bug or configuration problem.
PFNCompleted	The number of PauseForNotifications commands that have been completed by the HSM. Normally, this is one less than the PFNIssued figure because there is normally one such command outstanding.
ANIssued	The number of Asynchronous Notification messages issued by the HSM to the hardserver. These messages indicate such things as the clear key being pressed and the HSM being reset. In later firmware revisions inserting or removing the smartcard or changing the non-volatile memory also generate asynchronous notifications.
ChanJobsIssued	The number of fast channel jobs issued to the HSM. The fast channel facility is unsupported on current HSMs. This number should always be 0.
ChanJobsCompleted	The number of fast channel jobs completed by the HSM. The fast channel facility is unsupported on current HSMs. This number should always be 0.
CPULoadPercent	The current processing load on the HSM, represented as a number between 0 and 100. Because an HSM 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, HSMs 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%.
HostIRQs	On PCI HSMs, the total number of interrupts received from the host. On current HSMs, approximately equal to the total of HostReadCount and HostWriteCount .
ChanJobErrors	The number of low-level (principally data transport) errors encountered while processing fast channel jobs. Should always be 0 on current HSMs.
HostDebugIRQs	On PCI HSMs, the number of debug interrupts received. This is used only for driver testing, and should be 0 in any production environment.
HostUnhandledIRQs	On PCI HSMs, the number of unidentified interrupts from the host. If this is nonzero, a driver or PCI bus problem is likely.

ID	Value
HostReadReconnect	On PCI HSMs, the number of deferred reads that have now completed. This should be the same as HostReadDeferred , or one less if a read is currently deferred.
ObjectsCreated	The number of times a new object has been put into the object store. This appears under the HSM's ModuleObjStats node.
ObjectsDestroyed	The number of items in the HSM's object store that have been deleted and their corresponding memory released.
ObjectCount	The current number of objects (keys, logical tokens, buffers, SEE Worlds) in the object store. This is equal to ObjectsCreated minus ObjectsDestroyed . An empty HSM contains a small number of objects that are always present.
CurrentTempC	The current temperature (in degrees Celsius) of the HSM main circuit board. First-generation HSMs do not have a temperature sensor and do not return temperature statistics.
MaxTempC	The maximum temperature recorded by the HSM's temperature sensor. This is stored in non-volatile memory, which is cleared only when the unit is initialized. First-generation HSMs do not have a temperature sensor and do not return temperature statistics.
MinTempC	The minimum temperature recorded by the HSM's temperature sensor. This is stored in non-volatile memory, which is cleared only when the unit is initialized. First-generation HSMs do not have a temperature sensor and do not return temperature statistics.
MemTotal	The total amount of RAM (both allocated and free) available to the HSM. This is the installed RAM size minus various fixed overheads.
MemAllocKernel	The total amount of RAM allocated for kernel (that is, non-SEE) use in an HSM. This is principally used for the object store (keys, logical tokens, and similar) and for big-number buffers.
MemAllocUser	The total amount of RAM allocated for user-mode processes in the HSM (0 for non-SEE use). This includes the size of the SEE Machine image, and the total heap space available to it. The HSM'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.
SystemFans	The fan speed (RPM) for each fan in the HSM.

How data is affected when a module loses power and restarts

nCipher 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 nCipher 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 E: nShield Connect and client configuration files

This appendix describes the configuration files that store the current configuration of an nShield Connect or client.

The *module configuration files* are stored on the internal file system of the nShield Connect. They are updated automatically when the nShield Connect is configured from the front panel. The configuration files are also exported automatically to the remote file system, where they can be edited manually and reloaded on the nShield Connect, if required. For more information, see [Configuring the nShield Connect with configuration files on page 68](#).

The *client configuration files* are stored in the client's file system. The client's hardserver can also be set up using environment variables. Environment variable settings override settings in the client configuration files. For more information, see [Setting environment variables on page 72](#).

Location of client configuration files

The client configuration files are in the directory `$NFAST_KMDATA/config/` on the client computer's file system. This directory can contain the following files:

File	Description
<code>config</code>	The master configuration file. This contains the current configuration for the client. It is always present in the directory.
<code>config-default</code>	The default configuration file. This can be used to restore factory settings for the client. It is created by the <code>cfg-mkdefault</code> utility.

You can also save backup copies of configuration files in this directory.

Location of module configuration files

When you configure the nShield Connect from the front panel, the configuration files are exported automatically to the remote file system defined in the `remote_file_system` section of the module configuration. The exported files are in the directory `/opt/nfast/kmdata/hsm-ESN/config`.

In this path, *ESN* is the electronic serial number of the network nShield Connect from which the files were exported. This directory contains the master configuration file `config`, which specifies the current configuration for the nShield Connect. It is always present in the directory.

Structure of configuration files

The 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 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 nShield Connect or client behavior. Each section begins with its name in square brackets, as in this example:

```
[slot_imports]
```

Module configuration files and client configuration files have some sections in common, and some specific to the type of file:

Both	Module file only	Client file only
server_settings	nethsm_settings	server_performance
module_settings	nethsm_eth	nethsm_imports
server_remotecomms	nethsm_gateway	rfs_sync_client
server_startup	nethsm_route	remote_file_system
load_seemachine	nethsm_eth1_enable	remote_administration_ service_startup
slot_imports	nethsm_enable	
slot_exports	cosmod	
dynamic_slots	hs_clients	
slot_mapping	rfs_client	
dynamic_slot_timeouts	sys_log	
	config_op	
	ui_lockout	

You can update the parameters defined in most of these sections to configure the way that the hardserver handles secure transactions between the nShield Connect and applications that run on the client.

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.

Note: Multiple clients can be added to one configuration file by separating each client entry from the next with a line consisting of one or more hyphens.

If a particular section is not present in the configuration file, it is assumed to have no entries.

Sections only in module configuration files

nethsm_settings

The `nethsm_settings` section defines settings specific to the nShield Connect. The section contains the following fields:

Field	Description
<code>enable_impauth_resilience</code>	When set to the default yes value, this field enables impauth resilience for the module. Setting this field to no disables impauth resilience.
<code>impauth_resilience_timeout</code>	This field specifies the interval of time that must pass for a resumable impauth resilience session to expire. In the event of network errors, clients can reconnect and resume use of keys and other loaded objects until the specified interval has passed (after which all previously loaded objects must be reloaded). Specify this interval either in the form Nt , where N is an integer and t is s for seconds, m for minutes, h for hours, d for days, or w for weeks, or as never (in which case sessions never expire). If you specify N but not t , the seconds are assumed. The default setting 604800 (604800 seconds is 1 week).

nethsm_eth

The `nethsm_eth` section defines the Ethernet interfaces for the nShield Connect. Each interface is defined by an entry as follows:

Field	Description
<code>iface</code>	The identifier for the interface. This value must be 1 or 0.

Field	Description
<code>addr</code>	The IP address of the nShield Connect. The default is 0.0.0.0.
<code>netmask</code>	The net mask for the nShield Connect. The default is 0.0.0.0.
<code>gateway</code>	This field is retained for backwards compatibility only. The IP address of the gateway is stored in the <code>nethsm_gateway</code> section and this field is set to 0.0.0.0.
<code>linkspeed</code>	This field specifies the link speed setting. It can be one of <code>auto/1Gb</code> (nShield Connect only), <code>10BaseT</code> , <code>10BaseT-FDX</code> , <code>100BaseTX</code> , or <code>100BaseTX-FDX</code> . We recommend that you accept the default <code>auto/1Gb</code> or <code>auto</code> setting. On the nShield Connect, this setting can auto-negotiate 1Gb Ethernet.

nethsm_gateway

The `nethsm_gateway` section defines the default Ethernet gateway for the nShield Connect. There is one field, as follows:

Field	Description
<code>gateway</code>	The IP address of the gateway for the nShield Connect. The default is 0.0.0.0.

nethsm_route

The `nethsm_route` section defines the static routes for the nShield Connect. Each route is defined by an entry as follows:

Field	Description
<code>addr</code>	The IP address of the route destination. The default is 0.0.0.0.
<code>masklen</code>	The length to mask for the route.
<code>gateway</code>	The IP address of the gateway for the route. The default is 0.0.0.0.

nethsm_eth1_enable

The `nethsm_eth1_enable` section defines if the eth1 interface is enabled.

Field	Description
<code>enable</code>	The indicator of whether the eth1 interface is enabled or disabled (default 'no').

nethsm_enable

The `nethsm_enable` section defines whether ipv4 and or ipv6 are enabled or disabled for the interfaces of the unit. The enable fields are:

Field	Description
<code>iface</code>	The ethernet interface (0 or 1) to which the following fields apply.
<code>enable_ipv4</code>	Indicator of whether the IPv4 protocol on the interface is enabled (default: 'yes').
<code>enable_ipv6</code>	Indicator of whether the IPv6 protocol on the interface is enabled (default: 'no').
<code>enable_ipv6_static_addr</code>	Indicator of whether the static IPv6 address for the interface is enabled (default: 'no').
<code>enable_ipv6_slaac</code>	Indicator of whether SLAAC on the interface is enabled (default 'no').

cosmod

The `cosmod` section defines the input configuration for the nShield Connect. The configuration is defined by an entry as follows:

Field	Description
<code>keymap</code>	The selected layout for the keyboard connected to the nShield Connect front panel. This value is either UK or US .

hs_clients

The `hs_clients` section defines the clients that are allowed to connect to the nShield Connect. Each client is defined by an entry as follows:

Field	Description
<code>addr</code>	The IP address of the client. The default is 0.0.0.0 .
<code>clientperm</code>	The type of connection permitted from the client. This is one of the following: <ul style="list-style-type: none"> unpriv (non-privileged connections) priv (privileged connections) priv_lowport (privileged connections on ports lower than 1024) The default is unpriv .
<code>keyhash</code>	The hash of the key that the client's nToken should present to authenticate itself. The default is 40 zeros, which means that no key authentication is required.
<code>esn</code>	The ESN of the client's nToken. (Only applicable if nToken authentication is used.)
<code>timelimit</code>	The timelimit in seconds for the impath connection. The connection time is unlimited if this is set to 0 . The default is 60*60*24s (1 day).
<code>datalimit</code>	The datalimit in bytes for the impath connection. The data allowed over the connection is unlimited if this is set to 0 . The default is 1024*1024*8b (8Mb).

rfs_client

The `rfs_client` section defines the remote file system where the module configuration is backed up and the master copy of the Security World data is located, as follows:

Field	Description
<code>remote_ip</code>	The IP address of the computer on which the remote file system resides.
<code>remote_port</code>	The port number on which the remote hardserver is listening.

sys_log

The `sys_log` section defines how the nShield Connect logging works:

Field	Description
<code>behaviour</code>	<p>The way the log is written. How the log is written is decided by setting either of the following:</p> <ul style="list-style-type: none"> • <code>log</code> • <code>push</code> <p>If <code>log</code> is set, the log is written only to the file system of the nShield Connect. It is lost if the nShield Connect is rebooted. Logging stops when the file system is full. If <code>push</code> is set, the log is automatically appended to the log file in the remote file system at the interval specified in <code>push_interval</code>.</p>
<code>push_interval</code>	The number of minutes between the log being appended when <code>behaviour</code> is set to <code>push</code> . The default is <code>60</code> . The minimum is <code>1</code> .

config_op

The `config_op` section defines whether a client computer is allowed to update the module configuration automatically (to push a configuration) from files on the client:

Field	Description
<code>push</code>	<p>Whether the client is allowed to push configuration files to the nShield Connect. This is decided by setting one of the following:</p> <ul style="list-style-type: none"> • <code>on</code> • <code>off</code> <p>If <code>on</code>, the client specified in the <code>remote_ip</code> section is allowed to update the configuration of the nShield Connect remotely.</p>
<code>remote_ip</code>	The IP address of the client that is allowed to push config files. The default is <code>0.0.0.0</code> .
<code>remote_keyhash</code>	The hash of the key with which the authorized client is to authenticate itself, or (as the default) 40 zeros to indicate no key authentication required.



The default value for `remote_keyhash` (40 zeros) specifies that no authentication should occur. We recommend specifying a key hash in place of this default.

ui_lockout

The `ui_lockout` section defines whether you can lock the nShield Connect using login settings:

Field	Description
<code>lockout_mode</code>	<p>Controls front panel access to the nShield Connect. Set to:</p> <ul style="list-style-type: none"> • locked Enables UI lockout without requiring a logical token. • locked_lt Enables UI lockout with a logical token (OCS) (requires a valid <code>ltui_hash</code> to be set). • unlocked No UI lockout (default).
<code>ltui_hash</code>	The hash of the logical token (LTUI) required to authorize access to the nShield Connect menu structure when <code>lockout_mode</code> is set to locked_lt .
<code>panel_poweroff</code>	This controls the function of the Power button on the nShield Connect front panel when it is in operational mode. The default setting of yes enables the Power button. When set to no , the Power button is disabled.

Sections in both module and client configuration files

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 `NFLAG_DETAIL` environment variable (see [Environment variables to control logging on page 222](#)).

The section contains the following fields:

Field	Description
<code>loglevel</code>	<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"> • <code>info</code> • <code>notice</code> • <code>client</code> • <code>remoteserver</code> • <code>error</code> • <code>serious</code> • <code>internal</code> • <code>startup</code> • <code>fatal</code> • <code>fatalinternal</code> <p>The default is <code>info</code>. For more information, see <i>Logging, debugging, and diagnostics on page 222</i>.</p> <p>If the <code>NFAST_SERVERLOGLEVEL</code> environment variable is set, it overrides any <code>loglevel</code> value set in the configuration file.</p> <p>Note: <code>NFAST_SERVERLOGLEVEL</code> is a legacy debug variable.</p>
<code>logdetail</code>	<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>
<code>connect_retry</code>	<p>This field specifies the number of seconds to wait before retrying a remote connection to a client hardserver. The default is 10.</p>
<code>connect_maxqueue</code>	<p>This field specifies the maximum number of jobs which can be queued on the hardserver. The default is 4096: this is also the maximum value. Setting <code>connect_maxqueue</code> to a high value allows high throughput, but may cause long latency if the hardserver goes down.</p>
<code>connect_broken</code>	<p>This field specifies the number of seconds of inactivity allowed before a connection to a client hardserver is declared broken. The default is 90.</p>
<code>connect_keepalive</code>	<p>This field specifies the number of seconds between <code>keepalive</code> packets for remote connections to a client hardserver. The default is 10.</p>
<code>accept_keepidle</code>	<p>This field specifies the number of seconds before the first <code>keepalive</code> packet for remote incoming connections. The default is 30.</p> <p>Ideally, <code>accept_keepalive</code> should be at least twice the value of the <code>connect_keepalive</code> setting on the unattended machines.</p>

Field	Description
<code>accept_keepalive</code>	<p>This field specifies the number of seconds between keepalive packets for remote incoming connections. The socket will be closed after up to ten consecutive probe failures. The default is 10.</p> <p>Ideally, <code>accept_keepalive</code> should be a value such that $(10 * \text{accept_keepalive}) > \text{connect_broken}$ on the unattended machine. Using the default values for both these fields will fulfil this requirement.</p>
<code>connect_command_block</code>	<p>When the nShield Connect has failed, this field specifies the number of seconds the hardserver should wait before failing commands directed to that HSM with a <code>NetworkError</code> message. For commands to have a chance of succeeding after the nShield Connect has failed this value should be greater than that of <code>connect_retry</code>. If it is set to 0, commands to an nShield Connect are failed with <code>NetworkError</code> immediately. The default is 35.</p>
<code>max_pci_if_vers</code>	<p>This field specifies the maximum PCI interface version number. If <code>max_pci_if_vers</code> is set to 0 (the default), there is no limit.</p>
<code>enable_remote_mode</code>	<p>If this field is set to <code>yes</code> (the default value) in the module configuration file, nShield Connect mode changing using the <code>nopclearnfail</code> utility is enabled. If set to <code>no</code>, mode changing using the <code>nopclearnfail</code> utility is disabled.</p> <p>Note: Do not set <code>enable_remote_mode</code> in the client configuration file.</p>
<code>enable_remote_reboot</code>	<p>If this field is set to <code>yes</code> (the default value) in the module configuration file, the nShield Connect remote reboot using the <code>nethsmadmin</code> utility is enabled. If set to <code>no</code>, remote reboot using the <code>nethsmadmin</code> utility is disabled.</p> <p>Run <code>cfg-pushnethsm</code> to push the new config file to the module.</p>
<code>enable_remote_upgrade</code>	<p>If this field is set to <code>yes</code> (the default value) in the module configuration file, the nShield Connect remote upgrade using the <code>nethsmadmin</code> utility is enabled. If set to <code>no</code>, remote upgrade using the <code>nethsmadmin</code> utility is disabled.</p> <p>Run <code>cfg-pushnethsm</code> to push the new config file to the module.</p>

These flags are those used by the `NFLAG_DETAIL` environment variable (see [Environment variables to control logging on page 222](#)).

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
<code>external_time</code>	<p>This flag specifies the external time (that is, the time according to your machine's local clock) with the log entry.</p>

Flag	Description
<code>external_date</code>	This flag specifies the external date (that is, the date according to your machine's local clock) with the log entry.
<code>external_tid</code>	This flag specifies the external thread ID with the log entry.
<code>external_time_t</code>	This flag specifies the external <code>time_t</code> (that is, the time in machine clock ticks rather than local time) with the log entry.
<code>stack_backtrace</code>	This flag specifies the stack backtrace with the log entry.
<code>stack_file</code>	This flag specifies the stack file with the log entry.
<code>stack_line</code>	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.
<code>msg_severity</code>	This flag specifies the message severity (a severity level as used by the <code>NFLOG_SEVERITY</code> environment variable) with the log entry.
<code>msg_categories</code>	This flag specifies the message category (a category as used by the <code>NFLOG_CATEGORIES</code> environment variable) with the log entry.
<code>msg_writeable</code>	This flag specifies message writeables, and extra information that can be written to the log entry, if any such exist.
<code>msg_file</code>	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.
<code>msg_line</code>	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.
<code>options_utc</code>	This flag showing the date and time in UTC (Coordinated Universal Time) instead of local time.
<code>unix_file_descriptor_max</code>	This field sets the number of file descriptors the hardserver must be capable of having open concurrently on UNIX. The value must be an integer. If <code>unix_file_descriptor_max</code> is set to 0 (the default), the value will be ignored by the hardserver. If it is set to a positive value, the hardserver will refuse to start if the file descriptor hard limit on the system is less than that value. This configuration entry can be used to ensure that the hardserver is capable of satisfying the maximum number of hardserver connections that applications may make use of.

module_settings

The `module_settings` section defines the settings for the nShield Connect that can be changed while the hardserver is running. The section contains the following fields:

Field	Description
<code>esn</code>	This field specifies the electronic serial number of the nShield Connect.
<code>priority</code>	This field specifies the priority of the nShield Connect. 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
<code>impath_port</code>	This field specifies the port on which the hardserver listens for incoming <code>impath</code> connections. The default is 9004. Setting this field to 0 specifies that the hardserver does not listen for incoming connections. Ensure that firewall settings are consistent with port settings. See the Installation Guide for more about firewall settings.
<code>impath_addr</code>	This field specifies the IP address at which the hardserver listens for incoming <code>impath</code> connections. If this field is set to <code>inaddr_any</code> (which is the default), the hardserver listens on all IP addresses.
<code>impath_interface</code>	This field specifies a string representing the name of the Ethernet interface on which the hardserver listens. This field is only examined if <code>impath_addr</code> is set to <code>inaddr_any</code> . By default, the <code>impath_interface</code> field is empty, which means that the hardserver listens on all interfaces. If the string is not recognized as the name of one of the interfaces of the nShield Connect, the hardserver does not listen.

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 client hardserver on page 89](#).

The section contains the following fields:

Field	Description
<code>unix_socket_name</code>	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> . If the <code>NFAST_SERVER</code> environment variable is set, it overrides any value set for <code>unix_socket_name</code> in the hardserver configuration file. For more information about environment variables, see Environment variables on page 218 .
<code>unix_privsocket_name</code>	This field specifies the name of the UNIX socket to use for privileged connections on UNIX. The default is <code>/dev/nfast/privnserver</code> . If the <code>NFAST_PRIVSERVER</code> environment variable is set, it overrides any value set for <code>unix_privsocket_name</code> in the hardserver configuration file.
<code>nt_pipe_name</code>	This field is not used on Unix systems.
<code>nt_pipe_users</code>	This field is not used on Unix systems.
<code>nt_privpipe_name</code>	This field is not used on Unix systems.
<code>nt_privpipe_users</code>	This field is not used on Unix systems.

Field	Description
nonpriv_port	<p>This field specifies the port on which the hardserver listens for local non-privileged TCP connections. The value 0 (which is the default) specifies none. Java clients default to connecting to port 9000.</p> <p>Ensure that your network firewall settings are correct. See the Installation Guide for more about firewall settings.</p> <p>If the NFAST_SERVER_PORT environment variable is set, it overrides any value set for nonpriv_port in the hardserver configuration file.</p>
priv_port	<p>This field specifies the port on which the hardserver listens for local privileged TCP connections. The value 0 (which is the default) specifies none. Java clients default to connecting to port 9001.</p> <p>If the NFAST_SERVER_PRIVPORT environment variable is set, it overrides any value set for priv_port in the hardserver configuration file.</p>

load_seemachine

The **load_seemachine** section of the hardserver configuration file defines SEE machines that the nShield Connects connected to the client should load and, if required, start for use by other clients. Each SEE machine is defined by the following fields:

Field	Description
module	This field specifies the nShield Connect on to which to load the SEE machine. The value must be an integer. A nShield Connect with this ID must be configured on the client computer.
machine_file	This field specifies the file name of the SEE machine.
userdata	This field specifies the userdata 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.
worldid_pubname	This field specifies the PublishedObject name to use for publishing the keyID of the started SEE machine. If this field is blank (" "), the keyID is not published. This field is ignored if the value of the userdata field is blank.
postload_prog	<p>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 -m module#.</p> <p>To run see-sock-serv directly on the nShield Connect, set this field to sockserv.</p>
postload_args	<p>This field specifies arguments to pass to the program specified by the postload_prog field. The argument -m module# is automatically passed as the first argument. The postload_args field is ignored if postload_prog is not specified or is blank.</p> <p>To run see-sock-serv directly on the nShield Connect, set this field to -p pubname.</p>

Field	Description
	This field specifies whether the SEE machine name and userdata should be pulled from the RFS. The default is <code>0</code> : set to <code>1</code> to pull the SEE machine and userdata from the RFS before loading on the remote nShield Connect.
<code>pull_rfs</code>	This field will be ignored if set on client machine configurations. Note: 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 <code>load_seemachine</code> section of your existing configuration file.

slot_imports

The `slot_imports` section defines slots from remote nShield Connects that will be available to the client. Each slot is defined by the following fields:

Field	Description
<code>local_esn</code>	This field specifies the ESN of the local nShield Connect importing the slot.
<code>local_slotid</code>	This field specifies the <code>slotID</code> to use to refer to the slot when it is imported on the local nShield Connect. The default is <code>2</code> .
<code>remote_ip</code>	This field specifies the IP address of the machine that hosts the slot to import.
<code>remote_port</code>	This field specifies the port for connecting to the nShield Connect.
<code>remote_esn</code>	This field specifies the ESN of the remote nShield Connect from which to import the slot.
<code>remote_slotid</code>	This field specifies the <code>slotID</code> of the slot to import on the remote nShield Connect. The value of this field must be an integer. The default is <code>0</code> .

slot_exports

The `slot_exports` section defines the slots on the HSMs connected directly to the client that the client hardserver should export for other network HSMs to import. Each local slot has an entry for each network nShield Connect that can import it, consisting of the following fields:

Field	Description
<code>local_esn</code>	This field specifies the ESN of the local nShield Connect whose slot can be imported by a network nShield Connect.
<code>local_slotid</code>	This field specifies the <code>slotID</code> of the slot that is allowed to be exported. The value must be an integer. The default is <code>0</code> .
<code>remote_ip</code>	This field specifies the IP address of the network nShield Connect that is allowed to import the slot. Use <code>0.0.0.0</code> to allow all machines. The default is <code>0.0.0.0</code> .
<code>remote_esn</code>	This field specifies the ESN of the nShield Connect allowed to import the slot. Leave the value blank to allow all permitted nShield Connects in the security world. The default is blank.

dynamic_slots

The `dynamic_slots` section defines the number of Dynamic Slots that each HSM supports.

Field	Description
<code>esn</code>	ESN of the HSM to be configured with Dynamic Slots.
<code>slotcount</code>	The number of Dynamic Slots that the HSM is to support. If set to 0 (default) the HSM does not support Remote Administration.

slot_mapping

The `slot_mapping` section defines, for each specified HSM, a slot that is exchanged with slot 0 of the HSM. Slot 0 becomes a Dynamic Slot and the local slot becomes the specified slot number. This enables applications and utilities that only support slot 0 to use Remote Administration.

Field	Description
<code>esn</code>	ESN of the HSM to which the the mapping is applied.
<code>slot</code>	<p>The slot number to be swapped with slot 0, so that:</p> <ul style="list-style-type: none"> Slot 0 refers to a Dynamic Slot The specified slot number refers to the local slot of the HSM. <p>If <code>slot</code> is set to 0 (default) there is no slot mapping.</p>

dynamic_slot_timeouts

The `dynamic_slot_timeouts` section defines timeout values that are used to specify expected smartcard responsiveness for all HSMs associated with the relevant host or client, when using the Remote Administration.

Field	Description
<code>round_trip_time_limit</code>	<p><code>round_trip_time_limit > 5s + network latency time</code></p> <p>Round trip (HSM to smartcard and back) time limit in seconds. The card is regarded as removed, if no response has been received within the allowed time. Expected network delays need to be taken into account when setting this. The default is ten seconds.</p>
<code>card_remove_detect_time_limit</code>	<p><code>card_remove_detect_time_limit >= round_trip_time_limit * 2</code></p> <p>Maximum number of seconds that can pass without a response from the smartcard, before it is regarded as removed and all the keys that it protects are unloaded. Lower values increase network traffic. The default is 30 seconds.</p>

Sections only in client configuration files

server_performance

The `server_performance` section defines the performance settings for the client hardserver. These are read only at hardserver start-up. This section contains the following fields:

Field	Description
<code>enable_scaling</code>	This field determines whether multi-threaded performance scaling is enabled or not. If this field is set to auto (or not set), the hardserver automatically chooses the best option for the available hardware (enabled when using nShield Connects, for which scaling is currently optimized, and disabled if using nShield Solos or any other local devices). It can explicitly be enabled by setting to yes , and explicitly disabled by setting to no .
<code>target_concurrency</code>	This field allows the level of concurrency to be tuned. The value must be an integer and will only come into effect when multi-threaded performance scaling is enabled. If <code>target_concurrency</code> is set to 0 (the default), the value will be automatically configured by the hardserver based on the available number of physical CPU cores. The target concurrency configured is written to the hardserver log.

nethsm_imports

The `nethsm_imports` section defines the network nShield Connects that the client imports. It can also be set up by the `nethsmenro11` utility. Each nShield Connect is defined by the following fields:

Field	Description
<code>local_module</code>	This field specifies the ModuleID to assign to the imported nShield Connect. The value must be an integer. An nShield Connect with this ID must not be already configured on the client computer.
<code>remote_ip</code>	This field specifies the IP address of the nShield Connect to import.
<code>remote_port</code>	This field specifies the port for connecting to the nShield Connect.
<code>remote_esn</code>	This field specifies the ESN of the imported nShield Connect.
<code>keyhash</code>	This field specifies the hash of the key that the nShield Connect should use to authenticate itself.
<code>timelimit</code>	This field specifies the time interval in seconds between renegotiations of the session key. The value 0 means no limit. The default is 60*60*24 seconds (that is, one day).

Field	Description
<code>dataLimit</code>	This field specifies the amount of data in bytes to encrypt with a given session key before renegotiating a new one. The value <code>0</code> means no limit. The default is <code>1024*1024*8</code> bits (that is, 8 megabytes). This field applies to Triple DES session keys only. AES session keys heed the <code>timeLimit</code> field but not the <code>dataLimit</code> field.
<code>privileged</code>	The value in this field specifies whether the client can make a privileged connection to the nShield Connect. The default is <code>0</code> , which specifies no privileged connections. Any other value specifies privileged connections.
<code>ntoken_esn</code>	This field specifies the ESN of this client's nToken, 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
<code>remote_ip</code>	The IP address of the remote server against which to synchronize.
<code>remote_port</code>	This field specifies the port for connecting to the nShield Connect.
<code>use_kneti</code>	Setting this option to <code>yes</code> requires the client to use an authenticated channel to communicate with the RFS server.
<code>local_esn</code>	This is only required if <code>use_kneti</code> is set to <code>yes</code> . It is the ESN of the local nShield Connect 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 nShield Connects allowed to access the file system on this client. Each nShield Connect is defined by an entry consisting of the following fields:

Field	Description
<code>remote_ip</code>	This field specifies the IP address of the remote nShield Connect that is allowed to access the file system on this client.
<code>remote_esn</code>	This field specifies the ESN of the remote nShield Connect allowed to access the file system on this client.

Field	Description
<code>keyhash</code>	This field specifies the hash of the key with which the client must authenticate itself to the nShield Connect. The default is 40 zeros, which means that no key authentication is required.
<code>native_path</code>	This field specifies the local file name for the volume to which this entry corresponds.
<code>volume</code>	This field specifies the volume that the remote host would access to use this entry.
<code>allow_read</code>	If this field is set to yes , it means that a remote server is allowed to read the contents of the file. The default is no .
<code>allow_write</code>	If this field is set to yes , it means that a remote server is allowed to write to the file. The default is no .
<code>allow_list</code>	If this field is set to yes , it means that a remote server is allowed to list the contents of the file. The default is no .
<code>is_directory</code>	If this field is set to yes , it means that this entry represents a directory. The default is no .
<code>is_text</code>	If this field is set to yes , it means that line endings should be converted to and from the Unix convention for transfers.

Note: If you upgrade from an earlier software version to v12 and are using Remote Administration, you need to manually add the following sections to your configuration file.

remote_administration_slot_server_startup

The `remote_administration_slot_server_startup` section defines the communication settings that are applied at start-up to the Remote Administration Service.

Field	Description
<code>port</code>	Which port to use to connect to the Remote Administration Service. The default is 9005.

Appendix F: Cryptographic algorithms

Symmetric algorithms

In the following tables, the column labelled `nfkmverify` documents the implemented behavior of the `nfkmverify` command-line utility (and its underlying library). An entry *N* in this column means that `nfkmverify` always rejects the algorithm, while *Y* means that `nfkmverify` always accepts the algorithm. A constraint on the number of bits means that `nfkmverify` only accepts keys of at least that many bits.

Symmetric Algorithms						
Algorithm	FIPS approval	Key length in bits	Use for ...			nfkmverify
			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	N	≥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.

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 nShield MSCAPI, CHIL, and PKCS #11 libraries).

The NIST-approved algorithms that nShield modules offer are:

- DSA
- ECDSA
- ECDH
- 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
- ARIA
- Camellia
- Arc Four
- CAST 6
- DES
- SEED
- SHA-160
- MD2
- MD5
- RIPEMD 160
- HMAC (MD5, RIPEMD160)

Appendix G: 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 using the command line on page 192* in *Working with keys on page 192*.

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
<code>simple</code>	Specifying the <code>simple</code> application type generates an nShield-native key. No special action is taken after the key is generated.
<code>custom</code>	Specifying the <code>custom</code> application type generates a key for custom applications that require the key blob to be saved in a separate file. Specifying <code>custom</code> also causes the generation of a certificate request and self-signed certificate. However, we recommend that you specify the <code>simple</code> (instead of <code>custom</code>) application type whenever possible.
<code>pkcs11</code>	Specifying the <code>pkcs11</code> 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: <ul style="list-style-type: none"> • <code>DES3</code> • <code>DH</code> • <code>DSA</code> • <code>ECDH</code> • <code>ECDSA</code> • <code>HMACSHA1</code> • <code>RSA</code> • <code>Rijndae1 (AES)</code> <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 <code>--no-verify</code> option.</p>

Parameter	Description
embed	<p>Specifying the embed application type generates a key for use with CHIL applications that:</p> <ul style="list-style-type: none"> Do not support hwcrhk key storage Have a key importation facility capable of reading PEM-format RSA key files. <p>Specify the hwcrhk application type for CHIL applications that support hwcrhk key storage.</p> <p>You can use a key of the embed application type like a PEM-format RSA/DSA key file, even though it is really a specially encoded reference to a key stored in <code>/opt/nfast/kmdata/local</code>. This allows you to use an embed key when integrating with applications that normally require software RSA keys. For example, you can supply an embed 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>
hwcrhk	<p>Specifying the hwcrhk application type generates a key for Cryptographic Hardware Interface Library (CHIL) applications that do not require embed keys. Only RSA, DSA, and DH key types are supported</p>
kpm	<p>Specifying the kpm application type generates a key for delivery by an nForce Ultra key server. The generatekey utility automatically creates a special ACL entry that permits a kpm to be delivered to an nForce Ultra's enrolled internal hardware security module.</p>
seeinteg	<p>Specifying the seeinteg 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>
seeconf	<p>Specifying the seeconf 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
alias	The VALUE for alias specifies an alias to assign to the key.

Option	Description
blobsavefile	When using the custom application type, the <i>VALUE</i> for blobsavefile specifies a file name of the form <i>FILENAME.EXT</i> 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 <i>ROOT_inf.txt</i> ; for asymmetric key types, the public key blob is also saved to a file whose name has the form <i>ROOT_pub.EXT</i> .
cardset	The <i>VALUE</i> for cardset specifies an OCS that is to protect the key (if protect is set to token). 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 nfkminfo).
certreq	<p>Setting certreq 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.</p> <p>To generate a certificate request you must set the <i>VALUE</i> for certreq to yes, which makes generatekey 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 <i>FILENAME_req.ext</i> (where <i>FILENAME</i> is a name of your choice).</p> <p>An extra file with a name of the form <i>FILENAME.ext</i> 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 certreq with the --retarget option to generate a self-signed certificate for an existing key.</p>
checks	For RSA key generation only, this specifies the number of checks to be performed. Normally, you should leave <i>VALUE</i> empty to let the module pick an appropriate default.
curve	For ECDH and ECDSA key generation only, the <i>VALUE</i> for curve 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.
embedconvfile	The <i>VALUE</i> for embedconvfile specifies the name of the PEM file that contains the RSA key to be converted.
embedsavefile	<p>When using the embed application type, the <i>VALUE</i> for embedsavefile 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.</p> <p>A certificate request and a self-signed certificate are also written. If the filename is <i>ROOT.EXT</i> then the request is saved to <i>ROOT_req.EXT</i> and the self-signed certificate is saved to <i>ROOT_selfcert.EXT</i>.</p>
from-application	When retargeting a key, the <i>VALUE</i> for from-application specifies the application name of the key to be retargeted. Only applications for which at least one key exists are acceptable.

Option	Description
from-ident	When retargeting a key, the <i>VALUE</i> for from-ident specifies the identifier of the key to be retargeted (as displayed by the nfkminfo command-line utility).
hexdata	The <i>VALUE</i> for hexdata 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 in the command line.
ident	The <i>VALUE</i> for ident specifies a unique identifier for the key in the Security World. For applications of types simple or hwcrhk , this is the key identifier to use (the exact identifier for simple , for hwcrhk 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.
keystore	The <i>VALUE</i> for keystore specifies the file name of the key store to use. This must be an nShield key store.
keystorepass	The <i>VALUE</i> for keystorepass specifies the password to the key store to use.
module	The <i>VALUE</i> for module 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). Note: You can also specify a module by setting the --module option.
paramsreadfile	The <i>VALUE</i> for paramsreadfile 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 <i>VALUE</i> for paramsreadfile is not specified, the module uses a default file.
pemreadfile	The <i>VALUE</i> for pemreadfile 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.
plainname	The <i>VALUE</i> for plainname 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 kmdata and not used otherwise.
protect	The <i>VALUE</i> for protect specifies the protection method, which can be module for security-world protection, softcard for softcard protection or token for Operator Card Set protection. The default is token , except for seconf keys, where the default is module . seeinteg keys are always token-protected. The softcard option is only available when your system has at least one softcard present.
pubexp	For RSA key generation only, the <i>VALUE</i> for pubexp 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
recovery	The <i>VALUE</i> for recovery enables recovery for this key and is only available for card-set protected keys in a recovery-enabled world. If set to yes , the key is recoverable. If set to no , key is not recoverable. The default is yes . Non-recoverable module-protected keys are not supported.
seeintegname	If present, the <i>VALUE</i> for seeintegname identifies a seeinteg key. The ACL of the newly generated private key is modified to require a certificate from the seeinteg key for its main operational permissions, such Decrypt and Sign (DuplicateHandle , ReduceACL , and GetACL are still permitted without certification.)
selfcert	The <i>VALUE</i> for selfcert 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 selfcert to yes , which makes generatekey 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 <i>FILENAME.ext</i> . You can use this parameter with the --retarget option to generated a self-signed certificate for an existing key.
size	For key types with variable-sized keys, the <i>VALUE</i> for size specifies the key size in bits. The range of allowable sizes depends on the key type and whether the --no-verify option is used. The default depends on the key type; for information on available key types and sizes, see <i>Cryptographic algorithms on page 267</i> . This parameter does not exist for fixed-size keys, nor for ECDH and ECDSA keys which are specified using curve .
strict	For DSA key generation only, setting the <i>VALUE</i> for strict to yes enables strict verification, which also limits the size to exactly 1024 bits. The default is no .
type	The <i>VALUE</i> for type 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 <i>Cryptographic algorithms on page 267</i> . The --verify option limits the available key types.
x509country	The <i>VALUE</i> for x509country specifies a country code, which must be a valid 2-letter code, for the certificate request.
x509dnscommon	The <i>VALUE</i> for x509dnscommon specifies a site domain name, which can be any valid domain name, for the certificate request.
x509email	The <i>VALUE</i> for x509email specifies an email address for the certificate request.
x509locality	The <i>VALUE</i> for x509locality specifies a city or locality for the certificate request.
x509org	The <i>VALUE</i> for x509org specifies an organization for the certificate request.
x509orgunit	The <i>VALUE</i> for x509orgunit specifies an organizational unit for the certificate request.

Option	Description
x509province	The <i>VALUE</i> for x509province specifies a province for the certificate request.
xsize	The <i>VALUE</i> for xsize 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								
	generate	import	retarget	custom	embed	hwcrhk	pkcs11	seconf	seeinteg	sessl	simple	kpm
alias	X	X	X									
blobsavefile	X	X	X	X								
cardset	X	X		X	X	X	X				X	X
certreq							X					
checks	X			X	X	X	X				X	X
curve	X			X	X	X	X	X	X		X	
embedconvfile		X			X							
embedsavefile	X	X	X		X		X					
from-application			X	X	X	X	X				X	X
from-ident			X	X	X	X	X				X	X
hexdata		X		X	X	X	X				X	
ident	X	X				X					X	X
keystore	X	X	X									
keystorepass	X	X	X									
module	X	X		X	X	X	X			X	X	X
nvrn	X	X		X	X	X	X				X	
paramsreadfile	X			X	X	X	X	X	X		X	
pemreadfile		X		X		X					X	X
plainname	X	X	X	X	X		X	X	X	X	X	X
protect	X	X		X	X	X	X	X	X	X	X	X
pubexp	X			X	X	X	X				X	X
qsize	X			X	X	X	X				X	X
recovery	X	X		X	X	X	X	X	X		X	X

Property	Action			Application								
	generate	import	retarget	custom	embed	hwcrhk	pkcs11	seeconf	seeinteg	seessl	simple	kpm
seeintegname				X							X	
selfcert							X					
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 H: Checking and changing the mode on an nShield Connect

This appendix tells you how to check and change the mode on an nShield Connect. You must change the mode to perform certain configuration tasks.

nShield Connect front panel controls

See [Chapter 3: The nShield Connect user interface on page 38](#) for a description of the nShield Connect user interface, including the front panel controls.

Note: We recommend that you use a keyboard to manage the front panel menu options and enter text. See [Using a keyboard to control the unit on page 43](#) for more information.

Available modes

The following modes are available:

- **Operational**
 - The default setting for day-to-day use
- **Initialization**
 - Sets the nShield Connect to start in pre-initialization mode
 - Allows you to use the nShield Connect to create a Security World or add the module to an existing one

Note: You cannot select **Maintenance** mode. It is managed by the nShield Connect and cannot be set by a user.

Identifying the current mode

You can check the current mode of an nShield Connect:

- At the nShield Connect itself
- By using the `enquiry` command-line utility from a client computer
- By using KeySafe from a client computer

Checking the mode at the nShield Connect

The status LED

The nShield Connect Status LED indicates the operational status of the module.

Status LED	Description
On, occasionally blinks off.	<p>Status: Operational mode</p> <p>The module is in Operational mode and accepting commands. The more frequently the Status LED blinks off, the greater the load on the module.</p>
Flashes two short pulses, followed by a short pause.	<p>Status: Initialization mode</p> <p>Existing Security World data on the module has been erased.</p> <p>The module is automatically placed in Initialization mode after a Security World is created. For more information, see the <i>nShield Connect User Guide</i>.</p>
Flashes two long pulses followed by a pause.	<p>Status: Maintenance mode</p> <p>Used for reprogramming the module with new firmware.</p> <p>The module only goes into Maintenance mode during a software upgrade.</p>

The front panel display screen

The nShield Connect screen shows a color-coded footer at the bottom of the display when it is not in Operational mode.

Footer color	Text in footer	Meaning
Yellow	Initialization	The system is rebooting or waiting for an Administrator Card to be inserted.
Blue	Maintenance	An administrative task is being performed. This mode is only entered during firmware upgrades.
Red	HSM Failed	The internal module has failed.

Checking the mode using enquiry

You can use the `enquiry` command-line utility to display information about the hardserver and the status of the nShield Connect. The `enquiry` utility is in the `bin` subdirectory of the `ncipher` directory. This is usually `/opt/nfast`.

To check the mode using `enquiry`:

1. Log in on the client computer as a user, and open a command window.
2. Run the command:

```
opt/nfast/bin/enquiry
```

The following is an example of the `enquiry` command output:

```

server:
enquiry reply flags  none
enquiry reply level  Six
serial number        #####-####-####-####
mode                  operational
version               #.#.#
speed index          ###
rec. queue           ##.##
...
version serial       #
remote server port   #####
Module #1:
enquiry reply flags  none
enquiry reply level  Six
serial number        #####-####-####-####
mode                  operational
version               #.#.#
speed index          ###
rec. queue           ##.##
...
rec. LongJobs queue  ##
SEE machine type     PowerPCSYF

```

In this example, the `mode` line (highlighted) shows that the nShield Connect is in `operational` mode.

Checking the mode by using KeySafe

You can use the **Module Status tree** of the KeySafe GUI to identify the current mode of the nShield Connect.

To check the mode using KeySafe:

1. Start KeySafe on a client computer.
2. Locate the **Module Status tree** (part of the **Security World status** panel) positioned to the bottom left of the KeySafe window.
3. Expand the **Security World** and/or **Outside Security World** nodes as required.
4. Locate the appropriate nShield Connect (**Module**).

The current mode of the module is displayed in the **State** field.

See [Appendix A: Using KeySafe on page 201](#) for more about using KeySafe. See [Module information on page 205](#) for more about checking the mode.

Changing the mode

You can change the mode of an nShield Connect using:

- The front panel controls of the nShield Connect
- The `nopc1earfai1` command-line utility from a client computer

Changing the mode using the front panel controls of an nShield Connect

To change the mode of an nShield Connect, use the front panel menu screens and dialogs to do the following:

1. Navigate to **HSM > Set HSM mode**.
2. Select **Initialisation** or **Operational** as required.

Changing the mode using remote mode and nopclearfail

You can enable or disable the ability to make remote mode changes, see [Enabling and disabling remote mode changes on page 79](#).

Once you have enabled remote mode changes, you can change the mode of an nShield Connect from a computer using the `nopclearfail` command, without accessing the unit itself.

Available commands

You can use the following commands to change the mode of a module:

Command	Resulting mode
<code>nopclearfail --operational -O</code>	Operational
<code>nopclearfail --initialization -I</code>	Pre-initialization

To change the mode, do the following:

1. Run either:
 - a. The `nopclearfail --operational | -O` command.
 - or:
 - b. The `nopclearfail --initialization | -I` command.
 When finished, the system responds with **OK**.

Note: The system responds with **OK**, regardless of whether the mode of the nShield Connect has changed or not. To confirm that state of the module, do the following:

2. Run the `enquiry` command.
The mode line of the `Module` section displays the current mode.

Appendix I: Upgrading the nShield Connect image file and associated firmware

The nShield image file package for the nShield Connect is a single software bundle that consists of the following components:

- The internal module firmware
- The unit operating system image and host side software.

The image file package is installed on the remote file system when you install the client software. The following sections describe how to load this firmware package onto your nShield module.

Version Security Number (VSN)

Each nShield image file has a Version Security Number (VSN). In addition, the module firmware has its own individual VSN. This number is increased whenever we improve the security of the image file and/or firmware.

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 an image file with a lower VSN than the currently installed version.



You can never load module firmware with a lower VSN than the currently installed firmware.

Ensuring you use an image file package with the highest available VSN allows you to benefit from security improvements and enhanced functionality. It also prevents future downgrades of the image file and/or that could potentially weaken security. However, you may choose to install an image file or associated 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 package, 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.

Key data

Security Worlds and key data are preserved on the RFS host computer only when you upgrade the unit image and/or firmware. You must restore the unit to the Security World if you wish to continue using the key data.

Note: Adding or restoring a module will require authorisation from a quorum of Administrator cards.

Note: When upgrading the Connect image file, client licenses and features activations on the nShield Connect will persist unless you deliberately factory state the unit (from the front panel menu).

For more information, see [Adding or restoring an HSM to the Security World on page 111](#).

Upgrading the nShield Connect image file and firmware using the front panel

Before you upgrade the image file and/or firmware, ensure that you have installed the latest Security World Software on the client computer and that the remote file system has an up-to-date set of Security World files.

To upgrade the image file and/or firmware:

1. Copy the directory that contains the image file and/or firmware which you require to the directory `/opt/nfast/nethsm-firmware`.
The latest image file and/or firmware is placed here by default when you install the Security World Software.
2. From the main menu on the unit, select **System > Upgrade system**.
3. Confirm that you want to upgrade the image file and/or firmware.
4. Select the directory that contains the image file and/or firmware which you require. You are informed that the files are being transferred.
5. Verify the image version, HSM (firmware) version, and image VSN that are displayed, and confirm the upgrade when prompted.
6. When the image file and/or firmware upgrade is complete, the front panel may be slow to respond. nCipher recommends a full power off and power on to complete the upgrade procedure and to restore optimum front panel responsiveness. Follow these steps:
 - a. On the front panel select **System > Shutdown/Reboot > Shutdown**.
 - b. Switch the power supplies, at the rear of the Connect, to **0**, then back to **1**.

Remotely enabling dynamic feature certificates including nShield Connect client licenses

Feature certificates contained on the remote file system (RFS) can be applied to the nShield Connect. The main use case for applying feature certificates is for enabling the client licenses dynamic feature which have been purchased after the initial nShield Connect purchase, although both static and other dynamic feature certificates can be applied.

To apply a dynamic feature certificate, e.g. nShield Connect client license, do the following:

Note: Feature certificates must be present on the RFS in the folder `%NFAST_KMDATA%\hsm-ESN\features`.

1. Use the `nethsmadmin` utility to list the nShield Connect feature files on the RFS. Run the command:

```
nethsmadmin --module=>MODULE --rfs=RFS_IP --list-features
```

In this command:

--module=>MODULE

Specifies the HSM to use, by it's ModuleID (default = 1)

--rfs=RFS_IP.

Specifies the IP address of the RFS.

2. Use the `nethsmadmin` utility to make the nShield Connect use a specific feature file from the RFS.

Run the command:

```
nethsmadmin --module=>MODULE --rfs=RFS_IP --apply-feature=feature_file
```

In this command:

--module=>MODULE

Specifies the HSM to use, by it's ModuleID (default = 1)

--rfs=RFS_IP.

Specifies the IP address of the RFS.

--apply-feature=FEATURE_FILENAME

Applies the feature certificate called *FEATURE_FILENAME*.

Note: *feature_file* must be the path to the feature file that is displayed when you run the `nethsmadmin --module=>MODULE-rfs_RFS_IP -list-features` command. Errors are reported if you use either just the feature name, or the full path. The file must be alphanumeric, and no longer than 150 characters.

The following is an example of the output expected when applying a dynamic feature:

```
Applying feature DYNAMIC_FEATURE to module MODULE_NO ...
Feature DYNAMIC_FEATURE application process started on module MODULE_NO
*DYNAMIC_FEATURE DETECTED*
Please restart you clientside hardserver and check the enquiry output to ensure the dynamic
feature has been applied correctly!
For the client licences feature check the 'max exported modules' section in enquiry to see
if the new client number has been applied correctly.
```

The following is an example of the output expected when applying a static feature:

```
Applying feature STATIC_FEATURE to module MODULE_NO ...
Feature STATIC_FEATURE application process started on module MODULE_NO
*STATIC_FEATURE DETECTED*
To be able to use the static feature please clear module MODULE_NO.

Use the fet utility to verify the feature was applied correctly.
```

In the above output:

DYNAMIC_FEATURE

Specifies the name of the dynamic feature file applied.

STATIC_FEATURE

Specifies the name of the staic feature file applied.

MODULE_NO.

Specifies the HSM that the feature was applied to.

Upgrading the nShield Connect image file and firmware from a privileged client

Before you upgrade the image file and/or internal module firmware, ensure you have installed the latest Security World Software on the client computer and that the remote file system has an up-to-date set of Security World files.

You can check the version of the current image file on the nShield Connect using the `enquiry` command and checking the reported `image version`.

To upgrade the image file and/or internal module firmware, do the following:

1. Use the `nethsmadmin` utility to list the `nethsm` image files on the RFS. Run the command:

```
nethsmadmin --list-images=rfs_IP_address
```

In this command:

`--list-images=rfs_IP_address`

Specifies the IP address of the RFS.

2. If the image file you require is not on the RFS, copy the directory that contains the required image file to the `/opt/nfast/nethsm-firmware` directory on the RFS.
3. Use the `nethsmadmin` utility to make the nShield Connect use a specific feature file from the RFS. Run the command:

```
nethsmadmin --module=>MODULE --upgrade-image=image_name
```

In this command:

`--module=>MODULE`

Specifies the HSM to use, by its `ModuleID` (default = 1).

`--upgrade-image=image_name`

Specifies the image file to use from the RFS.

Note: `image_name` must be the path to the image file that is displayed when you run the `nethsmadmin --list-images=rfs_IP_address` command. Errors are reported if you use either just the image name, or the full path.

The following is an example of the output:

```
nethsmadmin --upgrade-image=nethsm-firmware/0.4.11cam2/nCx3N.nff
Initiating appliance image upgrade using file nethsm-firmware/0.4.11cam2/nCx3N.nff
Upgrade operation state changed to: Image Transfer Initiated
Upgrade operation state changed to: Image Transferred
Upgrade operation state changed to: Image Verified
Not able to contact appliance because of reason(23): CrossModule,#1-NetworkError
Upgrade operation final state: Image Verified
Image upgrade completed. Please wait for appliance to reboot.
Please wait for approximately half an hour for the appliance to internally upgrade.
```

The following line is expected and requires no action:

```
Not able to contact appliance because of reason(23): CrossModule,#1-NetworkError
```

Note: If the nShield Connect suffers a loss of power while you are upgrading the image file and/or internal module firmware, you should exit the `nethsmadmin` utility and try to restart the process at Step 1, once power has been restored to the HSM.

When the upgrade is complete, the front panel of the nShield Connect may be slow to respond. If this becomes a problem, then at the point you wish to use the front panel, nCipher recommends that you carry out step 6. from [Upgrading the nShield Connect image file and firmware using the front panel on page 282](#).

Enabling and disabling remote upgrade

You can enable or disable the ability to remotely upgrade an nShield Connect, see [Enabling and disabling remote upgrade on page 80](#).

Once you have enabled remote upgrade, you can upgrade an nShield Connect from a computer using the `nethsmadmin` command, without accessing the unit itself.

After firmware installation

After you have installed new firmware, you can create a new Security World with the HSM or reinitialize the HSM into an existing Security World.

If you are initializing the HSM into a new Security World, see [Creating a Security World on page 91](#).

If you are re-initializing the HSM into an existing Security World, see [Adding or restoring an HSM to the Security World on page 111](#).

Appendix J: SNMP monitoring agent

This appendix describes the Simple Network Management Protocol (SNMP) monitoring agent. The SNMP monitoring agent provides you with components that you can add to your (third-party) SNMP manager application.

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 MIB files.

SNMP defines the following SNMP messages:

Message	Description
<code>get</code>	This message is sent by a manager to retrieve the value of an object at the agent.
<code>set</code>	This message is sent by a manager to set the value of an object at the agent.
<code>trap</code>	This message is sent by an agent to notify a management station of significant events.

The SNMP monitoring agent is based on the open-source Net-SNMP project, version 5.7.3. 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 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 the SNMP agent

The SNMP agent is installed with the installation of the Security World Software and starts automatically.

Default installation settings

When installing Security World Software, you may be prompted to select Security World Software components from a list. If you select **a11** components, then the SNMP agent is installed as part of a full Security World Software installation. The default installation directory for the nShield 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 you already have another SNMP agent running, you must configure the ports used by the agents in order to avoid conflicts before enabling the SNMP agent. A port is assigned by editing the **agentaddress** entry in the **snmpd.conf** file or by editing the **defaultPort** entry in **snmpd.conf** file. If both files have been edited, the **agentaddress** entry in **snmpd.conf** file takes priority for **snmpd**, and the **defaultPort** entry in **snmpd.conf** is ignored.

If no existing SNMP agent is found, the SNMP agent runs on the default port 161. If an existing SNMP agent is detected, and no SNMP agent configuration files are found (implying a fresh installation), the installer automatically configures the SNMP agent to use the first unused port above 161 by creating a new **snmpd.conf** configuration file with the appropriate directive. It then displays a message indicating the number of the port that is has selected.

If an existing SNMP agent is found and an existing 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** file or editing the **defaultPort** entry in **snmpd.conf** file. If both files have been edited, the **agentaddress** entry in **snmpd.conf** file takes priority for **snmpd**, and the **defaultPort** entry in **snmpd.conf** is ignored.

Starting the SNMP agent

The SNMP agent is started automatically however it can be stopped and started manually. To stop, start, or restart (stop and immediately start again) the SNMP daemon:

```
/etc/init.d/nc_ncsnmpd stop|start|restart
```

See [SNMP agent command-line on page 316](#) for more information on additional parameters accepted by **snmpd**.

Basic configuration

Protecting the SNMP installation

The SNMP agent allows other computers on the network to connect to it and make requests for information. The SNMP agent is based on the NET-SNMP code base, which has been tested but not fully reviewed by nCipher. We strongly recommend that you deploy the SNMP agent only on a private network or a network protected from the global Internet by appropriate network protection systems (e.g. a firewall, a network Intrusion Detection/Prevention System, etc.).

The default nShield SNMP installation allows read-only access to the Management Information Base (MIB). 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 SNMP agent.



We strongly advise that you use the SNMP User-based Security Model (USM) with Authentication and Privacy protocols selected, to ensure only authorised users can obtain information from the SNMP agent and the confidentiality and data integrity of the transferred information is protected.

Care has also been taken to ensure that malicious attackers are unable to inundate 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 SNMP agent sends commands to the module is throttled. For more information on setting the cache time-outs, see [Basic configuration on page 288](#).

Configuring the 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 SNMP agent to a host that has an existing SNMP agent installation, you may need to edit the SNMP configuration files (`snmpd.conf` and `snmp.conf`) associated with the 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 287](#).

Note: Make sure you protect access to the configuration files, since these contain information that defines the security parameters of the SNMP system.

By default, the SNMP configuration files are located in the `/opt/nfast/etc/snmp/` directory.

Re-reading SNMP configuration files

The SNMP agent reads its configuration files on startup, and any changes made after this point will have no effect. If new directives are added and need to be applied, the 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)
- kill -HUP signal sent to the snmpd agent process
- stop then restart the SNMP agent.

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 SNMP agent operates. It is required only if an agent is running.

The `snmpd.conf` file can contain any of the directives available for use in the `snmp.conf` file and may also contain the following Security World Software-specific directives:

Directive	Description
<code>statstimeout</code>	This directive specifies the length of time for which statistics commands are cached. The default is 5 seconds.
<code>admintimeout</code>	This directive specifies the length of time for which administrative commands are cached. The default is 60 seconds.
<code>keytable</code>	This directive sets the initial state of the key table to <code>none</code> , <code>all</code> , or <code>query</code> . See <code>listKeys</code> in Administration sub-tree overview on page 303 .
<code>enable_trap_zero_suffix</code>	This directive appends the '.0' suffix to object identifiers (OIDs) for backward compatibility. The default is 0 (disabled); the directive can be set to 1 to restore the suffix. Valid values are 0 and 1.
<code>memoryUsageOkThreshold</code>	This directive specifies the threshold (as a percentage) below which HSM memory usage is considered to be ok. The default is 0. See Memory usage monitoring on page 302 for more details.
<code>memoryUsageHighThreshold</code>	This directive specifies the threshold (as a percentage) at which HSM memory usage is considered to be too high. The default is 0. See Memory usage monitoring on page 302 for more details.

Note: There may be a tolerance gap between the `memoryUsageOkThreshold` and the `memoryUsageHighThreshold` values.

Note: The timeouts should be set to values that achieve a balance between receiving up to date information whilst preventing excessive load.

The SNMP agent persistent configuration file

On running the SNMP agent for the first time, the `persist` directory will be created. This contains configuration files that are maintained by the SNMP agent. This directory will be created in the

following location:

```
/opt/nfast/etc/snmp/persist
```

Modifications should only be made to the persist folder's `snmp.conf` file in order to create users. The files within this directory should otherwise only be managed by the SNMP agent itself.

User creation can be performed with the `createUser` directive. See [USM users on page 291](#). On initialisation of the agent the information is read from the file and the lines are removed (eliminating the storage of the master password for that user) and replaced with the key that is derived from it. This key is a localised key, so that unlike the password, if it is stolen it can not be used to access other agents.

Note: Do not modify the persistent `snmpd.conf` file while the agent is running. The file is only read on initialisation of the agent and it is overwritten when the SNMP agent terminates. Any changes made to this file while the SNMP agent directives is running will be lost. The SNMP agent should be stopped prior to adding `createUser` directories to the configuration file.

Agent Behaviour

There are a small number of directives that control the behaviour of the SNMP Agent when considering it as a daemon providing a network service.

agentaddress directive

The listening address(es) that the SNMP Agent will use are defined by the `agentaddress` directive. It takes a comma separated list of address specifiers where an address specifier consists of one or more of :

- a transport specifier `udp:` or `tcp`
- a hostname or IPv4 address
- a port number (e.g. `:161` or `:1161`).

The default behaviour is to listen on UDP port 161 on all IPv4 interfaces (i.e. equivalent to `udp:161`).

```
agentaddress localhost : 161,tcp:1161
```

`agentaddress` will listen on UDP port 161, but only on the loopback interface (the port specification `:161` is not strictly necessary as this is the default port) . It will also listen on TCP port 1161 on all IPv4 interfaces.

agentgroup and agentuser directives

The user and group that the SNMP Agent changes to after opening the listening port(s) are defined using the `agentgroup` and `agentuser` directives. The following must be used:

```
agentgroup ncsnmpd
agentuser ncsnmpd
```

System information

Most of the scalar objects in the `.iso.org.dod.internet.mgmt.mib-2.system` sub-tree can be configured.

```
sysLocation STRING
sysContact STRING
sysName STRING
```

The three directives above set the system location, contact or name for the SNMP Agent respectively. Ordinarily these objects are writable via a suitably authorised SNMP SET request, however, specifying one of these directives in the configuration file makes the corresponding object read-only.

```
sysServices INTEGER
```

Sets the value of the `sysService.0` object. RFC 1213 defines how the integer value is calculated.

```
sysDescr STRING
sysObjectID OID
```

The two directives above set the system description and object ID for the agent. These objects are not SNMP-writable, but these directives can be used by a network administrator to configure suitable values for them.

USM users

The SNMPv3 protocol supports a User based Security Model as defined in RFC-3414.

USM provides authentication and privacy (encryption) functions and operates at the message level allowing for the following security level to be used with SNMPv3:

- Communication without authentication and privacy (**noauth**)
- Communication with authentication and without privacy (**auth**)
- Communication with authentication and privacy (**priv**).

Within this document the three possible security levels are referred to as **noauth**, **auth** and **priv**. However, other forms are sometimes used within the NET-SNMP and the equivalents are:

Security level	Equivalents
noauth	noauthnopriv

Security level	Equivalents
auth	authnopriv
priv	authpriv

Users can be added to the SNMP configuration with the `createUser` directive, defining the security mechanisms to be used.

```
createUser [-e ENGINEID] username [SHA authpassphrase] [AES privpassphrase]
```

It would not normally be necessary to specify the engine ID, but if it is specified, **ENGINEID** is defined as a hexadecimal string of octets starting with the 0x prefix. The encoding of the engine ID is defined in the description of `SnmpEngineID` from RFC3411.

The following recommendations should be followed when defining the security parameters for SNMPv3:

- Select a 'Security Level' of Priv, (**authpriv**) or auth (**authNoPriv**).
 - **priv** is the preferred 'Security Level', since this will provide both data source authentication and confidentially protection for the SNMP messages.
 - **auth** is the minimum 'Security Level' that should be selected, since this will ensure that SNMP data sent/received has not been tampered with and has been sent from an authorised entity.
- Define separate **authpassphrase** and **privpassphrase**.
 - It is good security practice to have key separation.
- Use randomly generated passphrases which contain upper and lower case characters, numbers and symbols (e.g. ASCII characters 0x20 - 0x7E).
 - This should give an entropy per character of 6.57bits,
- Use either 15 char for 96 bits of security strength keys and 20 char for 128 bits security strength keys.
 - The minimum length of both **auth** and **priv** passphrases is eight characters.
 - If a random passphrase is not used, consult NIST SP800-63-2 - Appendix A to determine the security strength of the password and the resultant keys. See: <http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-63-2.pdf>.

Note: MD5 and DES are not supported or enabled in the nShield distribution of SNMP. Only SHA may be used for authentication, and only AES may be used for privacy (encryption).

It is strongly recommended that `createUser` directives be added to the `persist/snmpd.conf` file, so that the passphrases are not available after the SNMP agent is installed. See [USM users on page 291](#). The user can then be referenced in access control directives(s) after which it can be used.

Traditional access control

Most simple access control requirements can be specified using the directives `rouser/rwuser` (for SNMPv3) or `rocommunity/rwcommunity` (for SNMPv1 or SNMPv2c).

```
rouser [-s usm] USERNAME [noauth | auth | priv [OID | -V VIEW [CONTEXT]]]
rwuser [-s usm] USERNAME [noauth | auth | priv [OID | -V VIEW [CONTEXT]]]
```

These directives specify that an SNMPv3 user (USERNAME) will be allowed read-only or read-write access respectively. The default (unspecified) security level is **auth**, which is the recommended minimum security level (see above). It is not recommended to use the usm security level **noauth**, where all SNMP messages are unauthenticated and any tampering of the message cannot be detected. Using **noauth** will reduce the security of the SNMP messages to the level of SNMPv1 or SNMPv2c.

OID restricts access for that user to the subtree rooted at the given OID.

VIEW restricts access for that user to the specified View-based Access Control Model (VACM) view name. An optional context can also be specified, or **context** to denote a context prefix. If no context field is specified (or the token * is used), the directive will match all possible contexts. (Contexts are a mechanism within SNMPv3 whereby an agent can support parallel versions of the same MIB objects, referring to different underlying data sets.)

A security model can be specified with "-s **SECMODEL**" however the default security model "usm" is the only security model which is supported in the nShield distribution of SNMP.

Example:

- Read-only user with access to the full OID tree requiring authentication as a minimum:

```
rouser user1
```

Or

```
rouser -s usm user1 auth .1
```

- Read-only user with access to the nCipher MIB allowing unauthenticated requests:

```
rouser user2 noauth .1.3.6.1.4.1.7682
```

- Read-write user with access to the full OID tree requiring authentication as a minimum:

```
rwuser user3
```

Or

```
rwuser user3 auth .iso
```

- Read-write user with access to the snmpVacmMIB subtree requiring authentication and encryption:

```
rwuser user4 priv snmpVacmMIB
```

Or

```
rwuser user4 priv .1.3.6.1.6.3.16
```

```
rocommunity COMMUNITY [SOURCE [ OID | -V VIEW [CONTEXT]]
rwcommunity COMMUNITY [SOURCE [ OID | -V VIEW [CONTEXT]]
```

Specifies an SNMPv1 or SNMPv2c community that will be allowed read-only (**GET** and **GETNEXT**) or read-write (**GET**, **GETNEXT** and **SET**) access respectively. By default, this will provide access to the full OID tree for such requests, regardless of where they were sent from. **SOURCE** allows access either from a particular range of source addresses, or globally ("**default**"). A restricted source can either be a specific hostname or address (e.g. **localhost** or 127.0.01), or a subnet - represented as IP/MASK (e.g. 10.10.10.0/255.255.255.0), or IP/BITS (e.g. 10.10.10.0/24).

OID **VIEW** and **CONTEXT** are as defined for **rouser** and **rwuser**.

Example:

- Setting up a read-only community named **public** that can be accessed by any user with the community name:

```
rocommunity public
```

- Setting up a read/write community named **private** that can only be accessed from the machine on which the agent is running:

```
rocommunity private localhost
```

In each case, only one directive should be specified for a given SNMPv3 user, or community string. It is not appropriate to specify both **rouser** and **rwuser** directives referring to the same SNMPv3 user (or equivalent community settings). The **rwuser** directive provides all the access of **rouser** (as well as allowing **SET** support). The same applies to **rwcommunity** and **rocommunity**.

More complex access requirements (such as access to two or more distinct OID subtrees, or different views for **GET** and **SET** requests) should use VACM configuration directives.

VACM configuration

The full flexibility of the VACM, for example allowing access to two or more distinct OID subtrees, or different access requirements for reading and writing, is available using four configuration directives - `com2sec`, `group`, `view` and `access`. The directives essentially define who has access and what they have access to using four directives. The first two directives (`com2sec` and `group`) define the who, while the last two (`view` and `access`) define the what.

```
Com2sec [-Cn CONTEXT] SECNAME SOURCE COMMUNITY
```

Maps an SNMPv1 or SNMPv2c community string to a security name. As it defines the community and maps it to a security name, `rocommunity`/`rwcommunity` directives are not required when using the directive.

SECNAME is the security name to be defined.

SOURCE is as defined for the `rocommunity`/`rwcommunity` directives above.

COMMUNITY defines the community name to be mapped to the security name. The same community string can be specified in several separate directives with different source tokens, and the first source/community combination that matches the incoming request will be selected. Various source/community combinations can also map to the same security name.

CONTEXT if defined (using `-Cn`), means that the community string will be mapped to a security name in the named SNMPv3 context. Otherwise the default context ("") will be used.

Example:

Creating three SNMPv1/v2c community names (`private`, `public` and `ltd`), where `private` and `ltd` only allow requests from the machine on which the SNMP Agent is running (note lines beginning with a `#` in `snmpd.conf` are treated as comments):

```
#          [-Cn CONTEXT] SECNAME      SOURCE      COMMUNITY
com2sec    ""              sec_private localhost private
com2sec    ""              sec_public  default  public
com2sec    ""              sec_limited localhost ltd
```

```
group GROUP v1 | v2c | usm SECNAME
```

Maps a security name (in the specified security model) into a named group. Several group directives can specify the same group name, allowing a single access setting to apply to several users and /or community strings. Note that groups must be set up for the two community-based models separately - a single `com2sec` directive will typically be accompanied by two `group` directives.

GROUP is the group name being defined/added to `v1`, `v2c` or `usm` defines the security model to which the definition relates **SECNAME** is the security (USM user name or security name defined by `com2sec` to be added to the group.

Example:

Creating three groups (**grp_private**, **grp_public**, **grp_limited**) for three USM users (**user1**, **user2** and **user3**) and the three communities shown in the **com2sec** example above:

```
# GROUP v1|v2c|usm SECNAME
group grp_private v1 sec_private
group grp_private v2c sec_private
group grp_private usm user1

group grp_public v1 sec_public
group grp_public v2c sec_public
group grp_public usm user2

group grp_limited v1 sec_limited
group grp_limited v2c sec_limited
group grp_limited usm user3
```

```
view VNAME included | excluded OID [MASK]
```

Defines a named **view** - a subset of the overall OID tree. This is most commonly a single subtree, but several **view** directives can be given with the same view name (**VNAME**), to build up a more complex collection of OIDs. An optional mask can also be specified, providing a means of indicating which parts of the OID must be matched.

VNAME is the view being modified.

included | **excluded** allows you to define whether the view includes or excludes the subtree, allowing the definition of a more complex view (e.g. by excluding certain sensitive objects from an otherwise accessible subtree).

MASK is an optional list of hex octets (separated by '.' or ':') whose bits indicate which OID sub-identifiers to match against. So for example if we assume we have an OID with 11 sub-identifiers (**.1.3.6.1.x.y.z.table.entry.column.1**) where the last four relate to a table, an entry, a column and index 1, specifying a **MASK** value of "**FF.A0**" (i.e. 1111111110100000) maps to this OID as follows:

```
1.3.6.1.x.y.z.table.entry.column.1
1 1 1 1 1 1 1 1 1 0 1
```

i.e. this mask means all parts of the OID except the column must match, therefore defining a view to any column of the first row of the table.

By including and excluding various aspects of the full OID tree, it is possible to define fine grained visibility within a view's definition.

Example:

Creating five views where **vw_sysContact** only has access to the **system.sysContact.0** OID, **vw_ncipher** only has access to the MIB, **vw_global1** has access to the full OID tree, **vw_ncipher_stats** only

has access to `nCipher.nC-series.statistics` and `vw_nCipher_admin` only has access to `nCipher.nC-series.administration`:

#	VNAME	included excluded	OID	[MASK]
view	<code>vw_sysContact</code>	excluded	.1	
view	<code>vw_sysContact</code>	included	<code>system.sysContact.0</code>	FF.80
view	<code>vw_nCipher</code>	excluded	.iso	
view	<code>vw_nCipher</code>	included	<code>.1.3.6.1.4.1.7682</code>	
view	<code>vw_global</code>	included	.1	
view	<code>vw_nCipher_stats</code>	excluded	.1	
view	<code>vw_nCipher_stats</code>	included	<code>enterprises.nCipher.nC-series.statistics</code>	
view	<code>vw_nCipher_admin</code>	excluded	.1	
view	<code>vw_nCipher_admin</code>	included	<code>enterprises.nCipher.nC-series.administration</code>	

```
access GROUP CONTEXT any | v1 | v2c | usm noauth | auth | priv exact | prefix READ WRITE NOTIFY
```

Maps from a group of users/communities (with a particular security model and minimum security level, and specific context) to one of three views, depending on the request being processed.

GROUP is a group name defined by the group directive and specifies the group that access is being defined for.

CONTEXT specifies the context for the access (the default context is the empty string ""). The context of incoming requests must match against the context either exactly or by prefix, as specified by the choice of **exact** | **prefix** made in this directive.

any, **v1**, **v2c**, or **usm** define the security model to which this definition relates.

noauth | **auth** | **priv** define the security level to which this definition relates. For **v1** or **v2c** access, this will need to be **noauth** as these protocols do not support authentication.

exact | **prefix** specify how **CONTEXT** should be matched against the context of the incoming request, either an exact match to **CONTEXT**, or prefixed by **CONTEXT**.

READ, **WRITE** and **NOTIFY** specifies the view to be used for **GET***, **SET** and **TRAP/INFORM** requests (although the **NOTIFY** view is not currently used). The keyword **none** is used if there is to be no access for that type of request.

Example:

Specifying that:

- SNMPv1 requests using the public community only have read access to the `enterprises.nCipher.nC-series.statistics` subtree,
- SNMPv2c requests using the public community only have read access to the `enterprises.nCipher.nC-series.administration.subtree`,
- SNMPv3 requests using the user2 USM user, must as a minimum be authenticated, and have read, notify access to the nCipher MIB (i.e. `enterprises nCipher`)

- SNMPv3 requests using the user1 USM user, must as a minimum be authenticated and encrypted, and have read, write and notify access to the full OID tree. Note that since requests must be authenticated and encrypted as a minimum, SNMPv1 and v2c requests using the private community cannot be made even though the community is included in grp_private.
- SNMPv1 and SNMPv2 requests using the ltd community and SNMPv3 requests using the user3 USM user, do not require to be authenticated or encrypted, and have read, write access to the system.sysContact.0 OID.

#	GROUP	CONTEXT	SECMODEL	LEVEL	PREFIX	READ	WRITE	NOTIFY
access	grp_public	""	v1	noauth	exact	vw_nCipher_stats	none	none
access	grp_public	""	v2c	noauth	exact	vw_nCipher_admin	none	none
access	grp_public	""	usm	auth	exact	vw_nCipher	none	vw_
nCipher								
access	grp_private	""	any	priv	exact	vw_global	vw_global	vw_
global								
access	grp_limited	""	any	noauth	exact	vw_sysContact	vw_sysContact	none

Trap Configuration

The distribution of SNMP supports SNMPv1, SNMPv2 and SNMPv3 traps. Control over these traps is defined with a number of directives:

SNMPv1 and SNMPv2 traps

```
trapcommunity COMMUNITY
```

Defines the default community to be used when sending SNMPv1 or SNMPv2 traps. Note that this directive must be used prior to a `trapsink` or `trap2sink` directive that wishes to use this community.

COMMUNITY the community name to be used.

Example:

```
trapcommunity traps
```

```
trapsink HOST [COMMUNITY [PORT]]
trap2sink HOST [COMMUNITY [PORT]]
```

Defines a destination for SNMPv1 or SNMPv2 traps generated by the agent.

HOST is an address specifier defining the network target that traps will be sent to. It consists of an optional transport specifier (`udp` (default if not specified) or `tcp`) followed by a hostname or IPv4

address followed by an optional port number, delimited by colons ":". (e.g. `localhost` or `tcp:192.168.137.2:163`).

COMMUNITY if specified will be the community name used for the traps. If it is not specified, the most recently specified `trapcommunity` will be used.

PORT allows for port-number to be defined if it is not present as part of the **HOST** specification. If no port is defined, the default port number of 162 will be used.

When a TCP transport specifier is used the SNMP agent establishes the TCP connection with the trap manager at start-up. Therefore the trap manager must be started before the SNMP agent otherwise an error is reported for the line in the `snmpd.conf` file which defines the trap manager.

Likewise when the TCP connection between the SNMP agent and the trap manager is dropped, traps are lost. Therefore it is inadvisable to use TCP instead of UDP for the transport specifier of trap managers.

If TCP is used for the connection between the SNMP agent and the trap manager and the connection is lost, to re-establish the connection the SNMP agent must be restarted, with the trap manager running and able to accept a TCP connection from the SNMP agent.

For issues with the trap manager accepting TCP connections from a SNMP agent refer to trap manager documentation.

Example:

```
trap2sink udp:192.168.137.220:162 traps
```

SNMPv3 traps

```
trapsess [SNMPCMD_ARGS] HOST
```

Defines the configuration for a trap. This is the only way to define SNMPv3 traps and it is an alternative method for defining SNMPv1 and SNMPv2 traps.

SNMPCMD_ARGS are arguments that would be used for an equivalent `snmptrap` command. So for example to send an SNMPv3 trap as USM user `user1` with authentication and encryption, the value "`-v3 -u user1 -1 priv`" would be used.

HOST see host definition for `trap2sink` above.

Example:

```
trapsess -v3 -u user1 -1 priv udp:192.168.137.220:162
trapsess -v2c -c public 192.168.137.221:162
```

Using the SNMP agent with a manager application

Note: The nShield SNMP monitoring agent provides MIB files that can be added to your (third-party) SNMP manager application.

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 creates the following:

- 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 SNMP 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 SNMP agent is designed to monitor all current nShield modules, working with all supported versions of nShield firmware (contact Support for details of supported firmware).

MIB module overview

A large proportion of the 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 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 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 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 can be found in the following location:

```
%NFAST_HOME%\etc\snmp\mibs\ncipher-mib.txt
```

MIB functionality

The 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 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 three items that it would be useful to check at-a-glance:

Node name	R/W	Type	Remarks
hardserverFailed	R	TruthValue	True 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	True if any modules have failed.
load	R	Unsigned32	Percentage of total available capacity currently utilized.

Traps

The traps sub-tree (`enterprises.ncipher.nc-series.nc-traps`) contains traps that the SNMP agent sends when certain events occur. For details on configuring traps, see *USM users on page 291*.

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.
memoryUsageHighAlert	This trap is sent when the HSM memory usage high threshold has been reached or exceeded by an HSM. See section on Memory usage monitoring below for more details.
memoryUsageOkAlert	This trap is sent when the memory usage for an HSM falls below the HSM memory usage ok threshold. See section on Memory usage monitoring below for more details.

Note: Some traps can take up to five minutes to be received.

Note: Other generic Net-SNMP traps may also be received. These include the two below, see Net-SNMP project website for more details.

Net-SNMP trap name	Description
SNMPv2-MIB::coldStart	This trap is sent when the SNMP agent is started
NET-SNMP-AGENT-MIB::nsNotifyShutdown	This trap is sent when the SNMP agent is stopped

Memory usage monitoring

The HSM memory usage thresholds and memory usage traps provide a mechanism to monitor HSM memory usage for HSMs in which the SNMP agent's client computer are enrolled.

With memory usage monitoring enabled, there will be a `memoryUsageHighAlert` trap sent each time the currently in-use `memoryUsageHighThreshold` is reached or exceeded by an HSM.

With memory usage monitoring enabled, a `memoryUsageHighAlert` trap is also sent:

- If the SNMP agent starts up and recognises that there are HSMs in a high memory usage state or,
- If HSMs in a high memory usage state are enrolled or,
- If the SNMP agent loses and then re-gains contact with the local hardserver which is connected to HSMs in a high memory usage state or,
- If failed HSMs in a high memory usage state then recover.

For each of the four scenarios above, one `memoryUsageHighAlert` trap will be sent for each HSM in a high memory usage state.

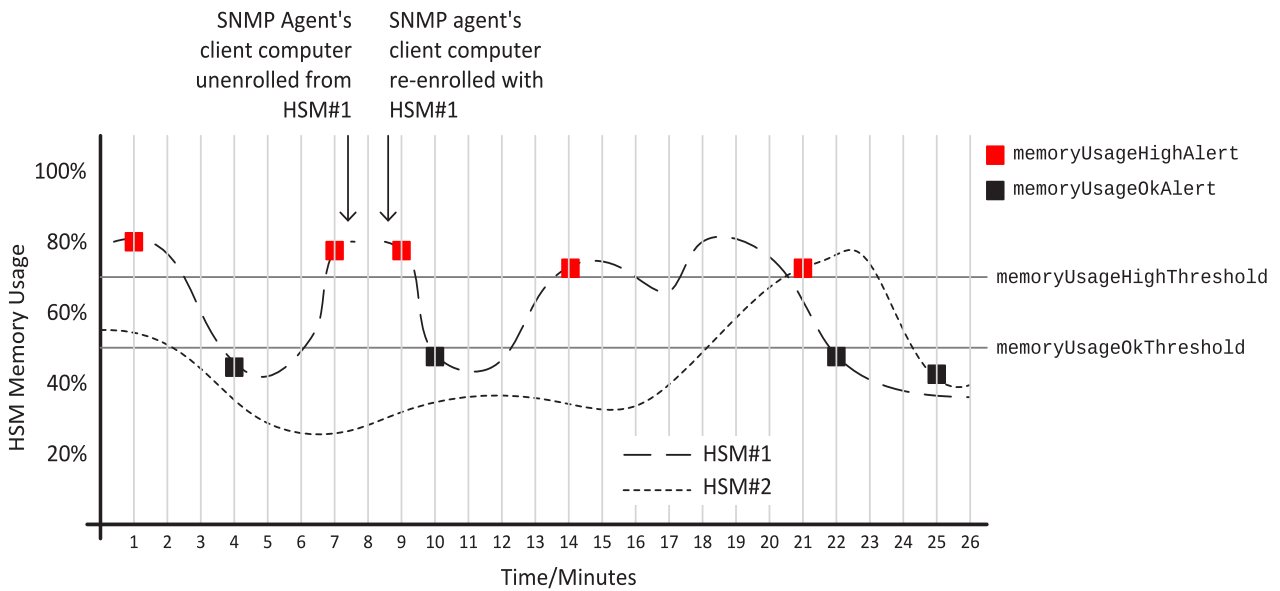
With memory usage monitoring enabled, there will be a `memoryUsageOkAlert` trap sent each time the memory usage for an HSM falls below the currently in-use `memoryUsageOkThreshold`.

The value for `memoryUsageOkThreshold` is read from the `snmpd.conf` file on starting the SNMP agent and is used provided it contains an integer value in the range 0 to 100 (inclusive); otherwise, the default value of 0 is used. The value for `memoryUsageHighThreshold` is processed in the same way.

Memory usage monitoring is enabled unless the in-use values for `memoryUsageOkThreshold` and `memoryUsageHighThreshold` are both 0 or the in-use values are such that `memoryUsageOkThreshold > memoryUsageHighThreshold`.

For example, in `snmpd.conf`, if `memoryUsageOkThreshold` is assigned an invalid value and `memoryUsageHighThreshold` is assigned a valid value of say 75%, then memory usage monitoring will be enabled and the values 0% and 75% will be used respectively.

An example of memory usage monitoring by an SNMP agent on a client computer enrolled with 2 HSMs is given below:



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
<code>hardserverRunning</code>	R	Enum 1: Running 2: NotRunning	This variable reflects the current state of the hardserver (Running or NotRunning).
<code>noOfModules</code>	R	Gauge32	Number of nC-series modules.
<code>hsVersion</code>	R	DisplayString	Hardserver version string.
<code>globalSpeedIndex</code>	R	Gauge32	Number of 1024-bit signatures each second.
<code>globalminQ</code>	R	Gauge32	Minimum recommended queue.
<code>globalmaxQ</code>	R	Gauge32	Maximum recommended queue.
<code>SecurityWorld</code>	R	TruthValue	True if a Security World is installed and operational.
<code>swState</code>	R	DisplayString	Security World display flags, as reported by <code>nfkminfo</code> .

Node name	R/W	Type	Remarks
listKeys	R/W	Integer	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.
		1: none	
		2: all	
		3: query	
4: resetquery			
serverFlags	R	DisplayString	Supported hardserver facilities (the NewEnquiry level 4 flags).
remoteServerPort	R	Gauge32	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** file (see [Basic configuration on page 288](#)).

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	Gauge32	The default quorum of Administrator cards.
adminQuorumN	R	Gauge32	The total number of cards in the ACS.
adminQuorumM	R	Gauge32	The quorum required for module reprogramming.
adminQuorumR	R	Gauge32	The quorum required to transfer keys for OCS replacement.
adminQuorumP	R	Gauge32	The quorum required to recover the pass phrase for an Operator card.
adminQuorumNV	R	Gauge32	The quorum required to access nonvolatile memory (NVRAM).
adminQuorumRTC	R	Gauge32	The quorum required to update the Real Time Clock.
adminQuorumDSEE	R	Gauge32	The quorum required to view full SEE debug information.
adminQuorumFTO	R	Gauge32	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	Gauge32	Module number of this row in the table.

Node name	R/W	Type	Remarks
mode	R	Integer 1: Operational 2: Pre-init 3: Init 4: Pre-maint 5: Maint 6: AccelOnly 7: Failed 8: Unknown	Current module state.
fwVersion	R	DisplayString	Firmware version string.
speedIndex	R	Gauge32	Speed index (approximate number of 1024-bit modulo exponentiation operations possible per second) of module
minQ	R	Gauge32	Module minimum recommended queue length
maxQ	R	Gauge32	Module maximum recommended queue length
serialNumber	R	DisplayString	Module Electronic Serial Number (ESN).
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	Integer 1: unknown 2: usable 3: maintmode 4: uninitialized 5: factory 6: foreign 7: accelonly 8: failed 9: unchecked 10: initmode 11: preinitmode 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 serverFlags , but for each module.
versionSerial	R	Gauge32	Firmware Version Security Number (VSN); see <i>Version Security Number (VSN)</i> on page 281.
hashKNETI	R	MHash	K_{NETI} hash, if present.
longQ	R	Gauge32	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
<code>slotAdminModuleIndex</code>	R	Integer32	Module number of the module containing the slot.
<code>slotAdminSlotIndex</code>	R	Integer32	Slot number (1-based, unlike <code>nCore</code> which is 0-based).
<code>slotType</code>	R	Integer 1: Datakey 2: Smart card 3: Emulated 4: Soft token 5: Unconnected 6: Out of range 7: Unknown	Slot type.
<code>slotFlags</code>	R	DisplayString	Flags referring to the contents of the slot (from <code>slotinfo</code>).
<code>slotState</code>	R	Integer 1: Unused 2: Empty 3: Blank 4: Administrator 5: Operator 6: Unidentified 7: Read error 8: Partial 9: Out of range	partial refers to cards in a partially-created card set.
<code>slotListFlags</code>	R	DisplayString	Flags referring to attributes of the slot (from <code>getslotlist</code>).
<code>slotShareNumber</code>	R	Gauge32	Share number of card currently in slot, if present.
<code>slotSharesPresent</code>	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	Gauge32	Required number of cards in the card set.
cardsetN	R	Gauge32	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	Gauge32	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	
keyRecovery	R	Integer 1: Enabled 2: Disabled 3: No key 4: Unknown 5: Invalid 6: Unset	The value <code>unset</code> is never returned by the key table. If you set the value <code>unset</code> , the keys are not filtered based on any of the attributes.

Node name	R/W	Type	Remarks
		Integer	
		1: Module	
		2: Cardset	
keyProtection	R	3: No key	The value unset is never returned by the key table. If you set the value unset , the keys are not filtered based on any of the attributes.
		4: Unknown	
		5: Invalid	
		6: Unset	
keyCardsetHash	R	MHash	Hash of the card set protecting the key, if applicable.
keyFlags	R	DisplayString	Certificate and public key flags.
keyExtraEntries	R	Gauge32	Number of extra key attributes.
keySEEInteg	R	DisplayString	SEE integrity key, if present.
keyGeneratingESN	R	DisplayString	ESN of the module that generated the key, if present.
keyTimeLimit	R	Gauge32	Time limit for the key, if set.
keyPerAuthUseLimit	R	Gauge32	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	

Node name	R/W	Type	Remarks
keyQueryRecovery	R/W	Integer	The value <code>unset</code> is never returned by the key table. If you set the value <code>unset</code> , the keys are not filtered based on any of the attributes.
		1: Enabled	
		2: Disabled	
		3: No key	
		4: Unknown	
		5: Invalid	
keyQueryProtection	R/W	Integer	The value <code>unset</code> is never returned by the key table. If you set the value <code>unset</code> , the keys are not filtered based on any of the attributes.
		1: Module	
		2: Cardset	
		3: No key	
		4: Unknown	
		5: Invalid	
keyQueryCardsetHash	R/W	DisplayString	Hash of the card set protecting the key, if applicable.
keyQueryFlags	R/W	DisplayString	Certificate and public key flags.
keyQueryExtraEntries	R/W	Gauge32	Number of extra key attributes.
keyQuerySEESInteg	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	Gauge32	Time limit for the key, if set (0 for no limit).
keyQueryPerAuthUseLimit	R/W	Gauge32	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 configuration file: `snmp.conf` on page 289*.

Statistics sub-tree

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
<code>moduleStatsIndex</code>	R	Integer	Module number of this row (for <code>moduleStatsTable</code>).
<code>hsuptime</code>	R	Counter32	Uptime of the hardserver.
<code>cmdCountAll</code>	R	Counter32	Returned aggregated for all modules and all commands.
<code>cmdBytesAll</code>	R	Counter32	
<code>cmdErrorsAll</code>	R	Counter32	Returned as for <code>cmdCount</code> , returned value is combined module errors added to hardserver marshalling/unmarshalling errors.
<code>replyCountAll</code>	R	Counter32	
<code>replyBytesAll</code>	R	Counter32	
<code>replyErrorsAll</code>	R	Counter32	See notes above for <code>cmdErrors</code> .
<code>clientCount</code>	R	Gauge32	
<code>maxClients</code>	R	Counter32	
<code>deviceFails</code>	R	Counter32	
<code>deviceRestarts</code>	R	Counter32	
<code>outstandingCmds</code>	R	Counter32	Total number of outstanding commands over all modules.
<code>load[All]</code>	R	Counter32	

Module statistics table

The following table gives details of the nodes in the module statistics table (`enterprises.ncipher.nc-series.statistics.moduleStatsTable`):

Node name	R/W	Type	Remarks
<code>moduleStatsIndex</code>	R	Integer	Module number of this row (for <code>moduleStatsTable</code>).
<code>uptime</code>	R	Counter32	Uptime of the module.
<code>cmdCount</code>	R	Counter32	Returned aggregated for all commands.
<code>cmdBytes</code>	R	Counter32	
<code>cmdErrors</code>	R	Counter32	Returned as for <code>cmdCount</code> all the different error states aggregated into one counter.

Node name	R/W	Type	Remarks
replyCount	R	Counter32	
replyBytes	R	Counter32	
replyErrors	R	Counter32	See notes above for cmdErrors.
loadModule	R	Counter32	
loadModule	R	Counter32	
objectCount	R	Gauge32	
clock	R	DisplayString	Depending on the module settings, this can require K_{NSO} 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). Only available on non-XC variants.
maxTemp	R	DisplayString	Maximum temperature the module has ever reached. Only available on non-XC variants.
nvRAMInUse	R	Gauge32	
volatileRAMInUse	R	Gauge32	
tempSP	R	DisplayString	Only available on XC variants.
currentCPUTemp1	R	DisplayString	Only available on XC variants.
currentCPUTemp2	R	DisplayString	Only available on XC variants.
currentFanSpeed	R	DisplayString	Only available on XC variants.
currentFanDuty	R	DisplayString	Only available on XC variants.
CPUVoltage1	R	DisplayString	Only available on XC variants.
CPUVoltage2	R	DisplayString	Only available on XC variants.
CPUVoltage3	R	DisplayString	Only available on XC variants.
CPUVoltage4	R	DisplayString	Only available on XC variants.
CPUVoltage5	R	DisplayString	Only available on XC variants.
CPUVoltage6	R	DisplayString	Only available on XC variants.
CPUVoltage7	R	DisplayString	Only available on XC variants.
CPUVoltage8	R	DisplayString	
CPUVoltage9	R	DisplayString	
CPUVoltage10	R	DisplayString	
CPUVoltage11	R	DisplayString	

nShield Connect statistics table

The following table gives details of the nodes in the nShield Connect statistics table (`enterprises.ncipher.nc-series.statistics.netHSMStatsTable`):

Node name	R/W	Type	Remarks
<code>netHSMStatsIndex</code>	R	Integer	Table index (not module number).
<code>netHSMUptime</code>	R	Counter32	Host system uptime.
<code>netHSMCPUUsage</code>	R	Gauge32	CPU usage of unit host processor.
<code>netHSMUserMem</code>	R	Gauge32	Total user memory of unit.
<code>netHSMKernelMem</code>	R	Gauge32	Total kernel memory of unit.
<code>netHSMCurrentTemp</code>	R	DisplayString	Internal unit temperature (sensor 1).
<code>netHSMMaxTemp</code>	R	DisplayString	Maximum recorded temperature (sensor 1).
<code>netHSMCurrentTemp2</code>	R	DisplayString	Internal unit temperature (sensor 2).
<code>netHSMMaxTemp2</code>	R	DisplayString	Maximum recorded temperature (sensor 2).
<code>netHSMVoltage5V</code>	R	DisplayString	unit 5V power reading.
<code>netHSMVoltage12V</code>	R	DisplayString	unit 12V power reading.
<code>netHSMFan1Speed</code>	R	Gauge32	Fan 1 speed (RPM).
<code>netHSMFan2Speed</code>	R	Gauge32	Fan 2 speed (RPM).
<code>netHSMFan3Speed</code>	R	Gauge32	Fan 3 speed (RPM).
<code>netHSMIPAddress</code>	R	IpAddress	IP address of unit.
<code>netHSMDescription</code>	R	DisplayString	Textual description of module (for example, <code>Local module #3</code>).
<code>netHSMFan4Speed</code>	R	Gauge32	Fan 4 speed (RPM).
<code>netHSMVoltage3p3V</code>	R	DisplayString	3.3V Supply Rail Voltage
<code>netHSMCurrent3p3V</code>	R	DisplayString	3.3V Supply Rail Current
<code>netHSMCurrent5V</code>	R	DisplayString	5V Supply Rail Current
<code>netHSMCurrent12V</code>	R	DisplayString	12V Supply Rail Current
<code>netHSMVoltage5VSB</code>	R	DisplayString	5V Supply Rail Voltage (Standby)
<code>netHSMCurrent5VSB</code>	R	DisplayString	5V Supply Rail Current (Standby)
<code>netHSMTamperBattery1</code>	R	DisplayString	Voltage of Tamper Battery 1
<code>netHSMTamperBattery2</code>	R	DisplayString	Voltage of Tamper Battery 2
<code>netHSMPSUfailure</code>	R	TruthValue	Power Supply failure status

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
<code>connStatsIndex</code>	R	Integer32	Index of this entry.
<code>connNumber</code>	R	Integer32	Hardserver connection number.
<code>connUptime</code>	R	Counter32	Uptime of the connection.
<code>connCmdCount</code>	R	Counter32	Number of commands submitted through this connection.
<code>connCmdBytes</code>	R	Counter32	Number of bytes submitted through this connection.
<code>connCmdErrors</code>	R	Counter32	Number of marshalling errors on commands through this connection.
<code>connReplyCount</code>	R	Counter32	Number of replies received by this connection.
<code>connReplyBytes</code>	R	Counter32	Number of bytes received by this connection.
<code>connReplyErrors</code>	R	Counter32	Number of marshalling errors on replies through this connection.
<code>connDevOutstanding</code>	R	Gauge32	Number of commands outstanding on this connection.
<code>connQOutstanding</code>	R	Gauge32	Number of commands outstanding in the hardserver queue.
<code>connLongOutstanding</code>	R	Gauge32	Number of long jobs outstanding for this connection.
<code>connRemoteIPAddress</code>	R	IpAddress	IP Address of connection client.
<code>connProcessID</code>	R	Integer32	Process identifier reported by connection client.
<code>connProcessName</code>	R	DisplayString	Process name reported by connection client.
<code>connObjectTotal</code>	R	Gauge32	The total object count for a connection

Module/connection statistics table

The following table gives details of the nodes in the per module, per connection statistics table (`enterprises.nCipher.nC-series.statistics.connModuleStatsTable`).

Node name	R/W	Type	Remarks
<code>connModuleStatsConnId</code>	R	Integer	Identity of this connection

Node name	R/W	Type	Remarks
<code>connModuleStatsModuleIndex</code>	R	Integer	Index of the module entry
<code>connModuleStatsObjectCount</code>	R	Gauge32	The object count on this module for this connection

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
<code>netHSMModuleIndex</code>	R	Integer32	Module number
<code>netHSMFanIndex</code>	R	Integer32	Fan number
<code>netHSMFanSpeed</code>	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
<code>compIndex</code>	R	Integer	Table index.
<code>compName</code>	R	DisplayString	Component name.
<code>compOutput</code>	R	Component output name	Component name.
<code>compMajorVersion</code>	R	Gauge	
<code>compMinorVersion</code>	R	Gauge	
<code>compPatchVersion</code>	R	Gauge	
<code>compRepository</code>	R	DisplayString	Repository name.
<code>compBuildNumber</code>	R	Gauge	

SNMP agent command-line

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 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
-h	This option displays a usage message.
-H	This option displays the configuration file directives that the agent understands.
-v	This option displays version information.
-f	This option specifies not forking from the calling shell.
-a	This option specifies logging addresses.
-A	This option specifies that warnings and messages should be appended to the log file rather than truncating it.
-d	This option specifies the dumping of sent and received UDP SNMP packets.
-V	This option specifies verbose display.
-P	PIDFILE This option specifies the use of a file (PIDFILE) to store the process ID.
-q	This option specifies that information be printed in a more easily parsed format (quick print).
-D	This option turns on debugging output.
-p	NUM This option specifies running on port NUM instead of the default: 161.
-c	CONFFILE This option specifies reading CONFFILE as a configuration file.
-C	This option specifies that the default configuration files not be read.
-L	This option prints warnings and messages to stdout and err.
-s	This option logs warnings/messages to syslog.
-r	This option specifies not exiting if root-only accessible files cannot be opened.
-I	[-]INITLIST This option specifies a list of MIB modules to initialize (or not). Run <code>snmpd</code> with the <code>-Dmib_init</code> option for a list.
-l	LOGFILE This option prints warnings/messages to a file LOGFILE (by default, LOGFILE=log/snmpd.log).

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 SNMP agent. These utilities support the `-h` (display a usage message) as described in the table above.

Utility	Description
<code>snmpstat</code>	This utility monitors and manages SNMP information.
<code>snmpget</code>	This utility runs a single GET request to query for SNMP information on a network entity.
<code>snmpset</code>	This utility runs a single SET request to set SNMP information on a network entity.
<code>snmpgetnext</code>	This utility runs a single GET NEXT request to query for SNMP information on a network entity.
<code>snmpwalk</code>	This utility obtains and prints an SNMP table.
<code>snmptranslate</code>	This utility translates SNMP object specifications into human-readable descriptions.
<code>snmpbulkwalk</code>	This utility communicates with a network entity using repeated GET NEXT requests.
<code>snmpbulkwalk</code>	This utility communicates with a network entity using BULK 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 K: Morse code error messages

If a Hardware Security Module (HSM) encounters an unrecoverable error, it enters the error state. In the error state, the module does not respond to commands and does not write data to the bus.

The blue Status LED flashes the Morse distress code (SOS: three short pulses, followed by three long pulses, followed by three short pulses). The Morse distress code is followed by one of the error codes listed in the tables shown in this guide.

For internal security modules running firmware 2.61.2 and above, the error code listed in this chapter is also reported by the `enquiry` utility in the `hardware status` field of the `Module` and under `hardware errors` in the hardserver log.

Errors are a rare occurrence. If any module goes into the error state, except as a result of you issuing the `Fail` command, contact Support, and give full details of your set up and the error code.

Contact Support even if you successfully recover from the error by taking the recommended action. For troubleshooting information, see the relevant *Installation Guide* for your module type.

Reading Morse code

The following guidelines are useful when reading Morse code messages from the module:

- The duration of a dash (-) is 3 times the duration of a dot (.).
- The gap between components of a letter has the same duration as a dot.
- The gap between letters has the same duration as a dash.
- The duration of the gap between repeated series of letters (a Morse code word gap) is 7 times the duration of a dot.

The following table shows the error codes corresponding to numerals.

Numeral	Morse
1	. - - - -
2	. . - - -
3	. . . - -
4 -
5
6	-
7	- - . . .
8	- - - . .
9	- - - - .
0	- - - - -

Runtime library errors

Memory failures can occur if the module is exposed to excessive heat. If you experience these errors, check the ventilation around the module. The module generates considerable heat and, if not well ventilated, may be operating at too high a temperature, even if the rest of your server room is at an appropriate temperature.

The runtime library error codes could be caused by firmware bugs or by faulty hardware. If any of these errors is indicated, reset the module.

Code	Meaning
OLC ---	SIGABRT: assertion failure and/or <code>abort()</code> called.
OLD ---	Interrupt occurred when disabled.
OLE ---	SIGSEGV: access violation.
OLI ---	SIGSTAK: out of stack space.
OLJ ---	SIGFPE: unsupported arithmetic exception (such as division by 0).
OLK ---	SIGOSERROR: runtime library internal error.
OLN ---	SIGFATALPANIC: error in error handling code.

Codes **OLD**, and **OLE** are more likely to indicate a hardware problem than a firmware problem.

To reset a unit that is in an error state, turn off the unit and then turn it on again.

Hardware driver errors

In general, the hardware driver error codes described in the following table indicate that some form of automatic hardware detection has failed. As well as indicating simple hardware failure, one of these error codes could indicate that there is a bug in the firmware or that the wrong firmware has been loaded.

Note: In the following table, the symbol “#” stands for a given numeral’s Morse code representation.

If any of these errors is indicated, contact support.

Code	Meaning
HL	Dorris M48T37 NVRAM (or battery) failed
HB	Debug serial port initialization failed.
HC	Processing thread initialization failed.
HCP	Card poll thread initialisation failed.
HD	Failure reading unique serial number.
HE	EEPROM failed on initialization.
HF	Starting up crypto offload.

Code	Meaning
HI	Interrupt controller initialization failed.
HM	System hardware initialization failed.
HO	Token interface initialization failed.
	Random number generator failed.
HR	Note: This code may also be generated if an attempt is made to downgrade firmware on a JackEC or Jack Lite to version 2.50.x or older.
HRS	RNG startup failed.
HRT	RNG selftest failed.
H RTP	Periodic (scheduled daily) RNG selftest failed.
HRM	RNG data matched.
HRZ	Impossible RNG failure (match after PRNG).
HS	Unexpected error from SCSI controller or host interface initialization failed.
HV	Environment sensors failed (for example, temperature sensor)
HCV	CPLD wrong version for PCI policing firmware.
H P P	PCI Interface Policing failure.
HST	Speed test failed.
H H R	RTC hardware detection failed or random number generator detection failed.
H R H	RNG hardware failed during operation
KR	RSA selftest failed.
H M n	DSP <i>n</i> failed self-test at start up.
H C n C A	CPU <i>n</i> failed self-test; no memory for cached RAM test.
H C n C C	CPU <i>n</i> failed self-test; CPU ID check failed.
H C n C F	CPU <i>n</i> failed self-test; freeing memory for cached RAM test.
H C n C G	CPU <i>n</i> failed self-test; setting up cached RAM test.
H C n C R	CPU <i>n</i> failed self-test; read error during cached RAM test.
H C n C V	CPLD version number incorrect (Dorris).

Code	Meaning
HC n C W	CPU <i>n</i> failed self-test; write error during cached RAM test.
HC n K A	CPU <i>n</i> failed selftest - AES known-answer test.
HC n K B	CPU <i>n</i> failed selftest - AES CMAC known-answer test.
HC n K C	CPU <i>n</i> failed selftest - ECDSA known-answer test
HC n K E	CPU <i>n</i> failed self-test; DES known-answer test.
HC n K F	CPU <i>n</i> failed self-test; Triple-DES known-answer test.
HC n K H	CPU <i>n</i> failed self-test; SHA-1 known-answer test.
HC n KI	CPU <i>n</i> failed selftest - HMAC-SHA512 known-answer test.
HC n KJ	CPU <i>n</i> failed selftest - HMAC-SHA256 known-answer test.
HC n K M	CPU <i>n</i> failed self-test; HMAC-SHA1 known-answer test.
HC n K N	CPU <i>n</i> failed selftest - HMAC-SHA224 known-answer test.
HC n K P	CPU <i>n</i> failed selftest - HMAC-SHA384 known-answer test.
HC n K R	CPU <i>n</i> failed selftest - RSA known-answer test
HC n K S	CPU <i>n</i> failed self-test; DSA known-answer test.
HC n L C	CPU <i>n</i> failed self-test; locking check.
HC n P S	CPU <i>n</i> failed self-test; test terminated at start.
HC n RT	CPU <i>n</i> failed selftest - RTC check.
HC n S A	CPU <i>n</i> failed self-test; no memory for uncached RAM test.
HC n S F	CPU <i>n</i> failed self-test; freeing memory for uncached RAM test.

Code	Meaning
HC n S R #	CPU <i>n</i> failed self-test; read error during uncached RAM test.
HC n S W #	CPU <i>n</i> failed self-test; write error during uncached RAM test.
HC n T S # -	CPU <i>n</i> failed self-test; could not start test.

Maintenance mode errors

The following error codes indicate faults encountered when a module is in the maintenance mode.

Code	Meaning	Action
I D	Copies of metadata do not match when trying to run image.	Contact Support.
I H	Bad metadata: hash mismatch.	Repeat firmware upgrade.
I I	Execution image does not match metadata.	Contact Support.
I L	Bad metadata: either bad length or bad metadata when running loadboot application.	Repeat firmware upgrade.
I M	Bad metadata: malformed ImageMetaData.	Repeat firmware upgrade.
I P	Bad metadata: bad padding.	Repeat firmware upgrade.
I R	Bad metadata: extra bytes at end.	Repeat firmware upgrade.
I S	Image entry point not found.	Contact Support.
I U	Bad metadata: ROM blank.	Repeat firmware upgrade.
I X	Bad metadata: malformed header.	Repeat firmware upgrade.
J H	Both copies of metadata invalid.	Contact Support.
H Z E	Monitor checksum failed.	Contact Support.
K F E	Flash sector erase failed.	Repeat firmware upgrade.
K F P	Flash sector program failed.	Repeat firmware upgrade.
M M D	No memory for download buffer.	Contact Support.

Note: For instructions on upgrading module firmware, see the appendix in the User Guide for your module type.

Operational mode errors

The following runtime library error codes could be caused by either bugs in the firmware or faulty hardware.

Code	Meaning	Action
D - . .	Fail command received.	Reset module by turning it off and then on again.
T -	Temperature of the module has exceeded the maximum allowable.	Restart your host computer, and improve module cooling.
G G G 	Failure when performing ClearUnit or Fail command.	Contact Support.

Note: To improve the cooling of your PCIe module, increase the distance between PCIe cards, and increase the airflow through your host computer.

Appendix L: Application Performance Tuning

Job Count

To achieve the best throughput of cryptographic jobs (such as Sign or Decrypt) in your application, arrange for multiple jobs to be on the go at the same time, rather than doing them one at a time. This is true even when using only a single HSM in your system.

When using a Solo, Solo+, Connect or Connect+, around 40 outstanding jobs per HSM is a good target when using an application that is coded directly against the nCore API. When using higher-level APIs such as PKCS#11 or CNG, your application may benefit from increasing the job count further, e.g. to 60 or more outstanding jobs per HSM.

The `ncperftest` utility supports performance measurements of a range of cryptographic operations with different job counts and client thread counts. You may find this useful to inform tuning of your application. Run `ncperftest --help` to see the available options.

Client Configuration

If your application is coded directly against nCore, you have a choice of sending multiple jobs asynchronously from a single client connection to the hardserver, or having multiple threads each with their own client connection to the hardserver with a single job sent synchronously in each. You can use the `--threads` parameter to the `ncperftest` utility to experiment with the performance impact of having more threads/connections with fewer jobs outstanding in each, or having fewer or just one thread/connection with more jobs outstanding in that connection.

When using higher-level APIs such as PKCS#11 or CNG, all cryptographic operations are synchronous, so larger numbers of threads must be used to increase the job count and make full use of HSM resources. These APIs automatically create a hardserver connection for each thread. If many HSMs are being used, a great many threads may be required to achieve best throughput. You can adjust the thread counts in the performance test tools for these APIs (e.g. `cksigtest` for PKCS#11 and `cngsoak` for CNG) to gauge how much concurrency is required for best throughput in your application.

Highly Multi-threaded Client Applications

If your application is highly multi-threaded, operating system defaults may not be optimal for best performance:

You may benefit from using a scalable memory allocator that is designed to be efficient in multi-threaded applications, examples include:

- `tcmalloc` - Windows, Linux
- `mtmalloc` - Solaris

- **MALLOCOPTIONS** environment variable - AIX
- **Ma11ocNextGen** - HP-UX.

On all systems except HP-UX, the **hardserver** is configured to use a scalable memory allocator by default, so only the client application requires special configuration. On HP-UX, the **hardserver** will automatically use the HP **Ma11ocNextGen** memory allocator if it is installed. **Ma11ocNextGen** is an optional package which can be downloaded free of charge from the HP website:

<https://h20392.www2.hp.com/portal/swdepot/displayProductInfo.do?productNumber=MallocNextGen>

It is recommended that **Ma11ocNextGen** be installed for best performance on HP-UX.

On some systems the default operating system scheduling algorithm is also not optimized for highly multi-threaded applications. A real-time scheduling algorithm such as the POSIX round-robin scheduler may yield noticeable performance improvements for your application. This has been observed to be especially the case on HP-UX, where you can execute your application using the **rtsched** utility to change the scheduling algorithm from the default.

File Descriptor Limits

On Linux/UNIX systems, large numbers of threads each with their own **hardserver** connection will require your application to make use of large numbers of file descriptors. It may be necessary to increase the file descriptor limit for your application. This can be done using **ulimit -n NewLimit** on most systems, but you may need to increase system-wide hard limits first.

Appendix M: Product returns

If you need to return your nShield Connect product, contact Support for instructions:

<https://help.ncipher.com>

Glossary

Authorized Card List

Controls the use of Remote Administration cards. If the serial number of a card does not appear in the Authorized Card List, it is not recognized by the system and cannot be used. The list only applies to Remote Administration cards.

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.

The Administrator Cards containing share in the logical tokens that protect the Security World keys, including K_{NSO} , the key-recovery key, and the recovery authorization keys. Each card contains one share from each token. The ACS is created using the well-known module key so that it can be loaded onto any nShield module.

See also *Security World, 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) on page 329*

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).

client identifier: R_{SC}

This notation represents an arbitrary number used to identify a client. In the nCore API, all client identifiers are 20 bytes long.

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 on page 334](#), [Advanced Encryption Standard \(AES\) on page 328](#)

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 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 \(DSA\) on page 329](#)

ECDH

A variant of the Diffie-Hellman anonymous key agreement protocol which uses elliptic curve cryptography.

See also [Diffie-Hellman on page 329](#).

ECDSA

Elliptic Curve DSA: a variant of the Digital Signature Algorithm (DSA) which uses elliptic curve cryptography.

See also [Digital Signature Algorithm \(DSA\) on page 329](#), [Diffie-Hellman on page 329](#).

encryption: $\{A\}_B$

This notation indicates the result of **A** encrypted with key **B**.

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 the internal hardware security module and applications on the network. The module hardserver is configured using the front panel on the module or by means of uploaded configuration data. Configuration data is stored on the module and in files in a specially configured file system on each client computer.

hardware security module (HSM)

A hardware security module (commonly referred to as an HSM) is a hardware device used to hold cryptographic keys and software securely.

hash: $H(X)$

This notation indicates a fixed length result that can be obtained from a variable length input and that can be used to identify the input without revealing any other information about it. The nShield module uses the Secure Hash Algorithm (SHA-1) for its internal security.

identifier hash: $H_{ID}(X)$

An identifier hash is a hash that uniquely identifies a given object (for example, a key) without revealing the data within that object. The module calculates the identity hash of an object by hashing together the object type and the key material. The identity hash has the following properties:

H_{ID} is not modified by any operations on the key (for example, altering the ACL, the application data field, or other modes and flags)

H_{ID} is the same for both public and private halves of a key pair.

Unique data is added to the hash so that a H_{ID} is most unlikely to be the same as any other hash value that might be derived from the key material.

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)*.

key object: K_A

This is a key object to be kept securely by the module. A key object may be a private key, a public counterpart to a private key, a key for a symmetric cipher (MAC or some other symmetric algorithm), or an arbitrary block of data. Applications can use this last type to allow the module to protect any other data items in the same way that it protects cryptographic keys. Each key object is stored with an ACL and a 20-byte data block that the application can use to hold any relevant information.

KeyID: ID_{KA}

When a key object KA is loaded within the module's RAM, it is given a short identifier or handle that is notated as ID_{KA} . This is a transient identifier, not to be confused with the key hash **HID(KA)**.

logical token: K_T

A logical token is a key used to protect key blobs. A logical token is generated on the nShield module and never revealed, except as shares.

MAC: MAC_{KC}

This notation indicates a MAC (Message Authentication Code) created using key **KC**.

module

See *hardware security module (HSM)*.

module key: K_M

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.

All modules include two module keys:

- module key zero K_{M0} , a module key generated when the module is initialized and never revealed outside the module.
- **null**, or well-known module key K_{MWK} .

You can program extra module keys into a module.

See also: *Security World, hardware security module (HSM)*.

module signing key: K_{ML}

The module signing key is the module's public key. It is used to issue certificates signed by the module. Each module generates its own unique K_{ML} and K_{ML}^{-1} values when it is initialized. The private half of this key pair, K_{ML}^{-1} , is never revealed outside the module.

nShield master feature enable key K_{SA}

Certain features of the module firmware are available as options. These features must be purchased separately from nCipher. To use a feature on a specific module, you require a certificate from nCipher

signed by K_{SA} . These certificates include the electronic serial number for the module.

nShield Remote Administration Card

Smart cards that are capable of negotiating cryptographically secure connections with an HSM, using warrants as the root of trust. nShield Remote Administration Cards can also be used in the local slot of an HSM if required. You must use nShield Remote Administration Cards with Remote Administration.

nShield Security Officer's key: K_{NSO}^{-1}

The notation K_{NSO}^{-1} indicates the Security Officer's signing key. This key is usually a key to a public-key signature algorithm.

nShield Trusted Verification Device

A smart card reader that allows the card holder to securely confirm the Electronic Serial Number (ESN) of the HSM to which they want to connect, using the display of the device. nCipher supplies the nShield Trusted Verification Device and recommends its use with Remote Administration.

null module key: K_{MWK}

The null module key is used to create tokens that can be loaded onto any module. Such tokens are required for recovery schemes. The null module key is a Triple DES key of a value 01010101. As this value is well known, this module key does not have any security. Key blobs cannot be made directly under the null module key. To make a blob under a token protected by the null module key, the key must have the ACL entry **AllowNullKMToken**.

Operator Card Set (OCS)

Part of the Security World architecture, an Operator Card Set (OCS) is a set of smart cards containing shares of the logical tokens 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, Administrator Card Set (ACS)*.

Recovery key: K_{RA}

The recovery key is the public key of the key recovery agent.

Remote access solution

The remote access solution, such as SSH or a remote desktop application, which is used as standard by your organization. Enables you to carry out Security World administrative tasks from a different location to that of an nShield Connect or nShield Solo.

For example, the remote access solution is used to run security world utilities remotely and to enter passphrases.

Note: nCipher does not provide this software.

Remote Administration

An optional Security World feature that enables Remote Administration card holders to present their cards to an HSM located elsewhere. For example, the card holder may be in an office, while the HSM is in a data center. Remote Administration supports the ACS, as well as persistent and non-persistent OCS cards, and allows all smart card operations to be carried out, apart from loading feature certificates.

nShield Remote Administration Client

A GUI or command-line interface that enables you to select an HSM located elsewhere from a list provided by the Remote Administration Service, and associate a card reader attached to your computer with the HSM. Resides on your local Windows or Linux-based computer.

Remote Administration Service

Enables secure communications between an nShield Remote Administration Card and the hardserver that is connected to the appropriate HSM. Listens for incoming connection requests from nShield Remote Administration Clients. Supplies a list of available HSMs to the nShield Remote Administration Client and maintains an association between the relevant card reader and the HSM.

Dynamic Slot

Virtual card slots that can be associated with a card reader connected to a remote computer. Remote Administration Slots are in addition to the local slot of an HSM and any soft card slot that may be available. HSMs have to be configured to support between zero (default) and 16 Remote Administration Slots.

Rijndael

See [Advanced Encryption Standard \(AES\) on page 328](#)

salt: X

The random value, or salt, is used in some commands to discourage brute force searching for keys.

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 \(ACS\)](#), [Operator Card Set \(OCS\)](#).

Security World key: K_{MSW}

The Security World key is the module key that is present on all modules in a Security World. Each Security World has a unique Security World key. This key is generated randomly when the Security World is created, and it is stored as a key blob protected by the ACS.

share: K_{Ti}

The notation K_{Ti} indicates a share of a logical token. Shares can be stored on smart cards or software tokens. Each share is encrypted under a separate share key.

share key: K_{Si}

A share key is a key used to protect an individual share in a token. Share keys are created from a Security World key, a pass phrase, and a salt value.

Standard nShield Cards

Smart cards used in the local slot of an HSM. Standard nShield cards are not supported for use with Remote Administration.

Standard card reader

A smart card reader for ISO/IEC 7816 compliant smart cards. nCipher recommends that standard smart card readers are only used with the nShield Remote Administration Client command-line utility, not the GUI.

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 \(DES\) on page 329](#), [Advanced Encryption Standard \(AES\) on page 328](#).