

SafeNet USB HSM 6.3

Administration Guide

Document Information

Product Version	6.3
Document Part Number	007-011302-015
Release Date	23 September 2022

Revision History

Revision	Date	Reason
A	23 September 2022	Initial release.

Trademarks, Copyrights, and Third-Party Software

Copyright 2001-2022 Gemalto. All rights reserved. Gemalto and the Gemalto logo are trademarks and service marks of Gemalto and/or its subsidiaries and are registered in certain countries. All other trademarks and service marks, whether registered or not in specific countries, are the property of their respective owners.

Acknowledgements

This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit.
(<http://www.openssl.org>)

This product includes cryptographic software written by Eric Young (eay@cryptsoft.com). This product includes software written by Tim Hudson (tjh@cryptsoft.com).

This product includes software developed by the University of California, Berkeley and its contributors.

This product uses Brian Gladman's AES implementation.

Refer to the End User License Agreement for more information.

Disclaimer

All information herein is either public information or is the property of and owned solely by Gemalto and/or its subsidiaries who shall have and keep the sole right to file patent applications or any other kind of intellectual property protection in connection with such information.

Nothing herein shall be construed as implying or granting to you any rights, by license, grant or otherwise, under any intellectual and/or industrial property rights of or concerning any of Gemalto's information.

This document can be used for informational, non-commercial, internal, and personal use only provided that:

- The copyright notice, the confidentiality and proprietary legend and this full warning notice appear in all copies.
- This document shall not be posted on any publicly accessible network computer or broadcast in any media, and no modification of any part of this document shall be made.

Use for any other purpose is expressly prohibited and may result in severe civil and criminal liabilities.

The information contained in this document is provided "AS IS" without any warranty of any kind. Unless otherwise expressly agreed in writing, Gemalto makes no warranty as to the value or accuracy of information contained herein.

The document could include technical inaccuracies or typographical errors. Changes are periodically added to the information herein. Furthermore, Gemalto reserves the right to make any change or improvement in the specifications data, information, and the like described herein, at any time.

Gemalto hereby disclaims all warranties and conditions with regard to the information contained herein, including all implied warranties of merchantability, fitness for a particular purpose, title and non-infringement. In no event shall Gemalto be liable, whether in contract, tort or otherwise, for any indirect, special or consequential damages or any damages whatsoever including but not limited to damages resulting from loss of use, data, profits, revenues, or customers, arising out of or in connection with the use or performance of information contained in this document.

Gemalto does not and shall not warrant that this product will be resistant to all possible attacks and shall not incur, and disclaims, any liability in this respect. Even if each product is compliant with current security standards in force on the date of their design, security mechanisms' resistance necessarily evolves according to the state of the art in security and notably under the emergence of new attacks. Under no circumstances, shall Gemalto be held liable for any third party actions and in particular in case of any successful attack against systems or equipment incorporating Gemalto products. Gemalto disclaims any liability with respect to security for direct, indirect, incidental or consequential damages that result from any use of its products. It is further stressed that independent testing and verification by the person using the product is particularly encouraged, especially in any application in which defective, incorrect or insecure functioning could result in damage to persons or property, denial of service, or loss of privacy.

All intellectual property is protected by copyright. All trademarks and product names used or referred to are the copyright of their respective owners. No part of this document may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, chemical, photocopy, recording or otherwise without the prior written permission of Gemalto

Regulatory Compliance

This product complies with the following regulatory regulations. To ensure compliancy, ensure that you install the products as specified in the installation instructions and use only Gemalto-supplied or approved accessories.

USA, FCC

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

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



Note: This equipment has been tested and found to comply with the limits for a “Class B” digital device, pursuant to part 15 of the FCC rules. These limits are designed to provide reasonable protection against harmful interference in a residential installation. This equipment generates, uses and can radiate radio frequency energy and, if not installed and used in accordance with the instructions, may cause harmful interference to radio communications. However, there is no guarantee that interference will not occur in a particular installation.

If this equipment does cause harmful interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one or more of the following measures:

- Reorient or relocate the receiving antenna
- Increase the separation between the equipment and receiver

- Connect the equipment into an outlet on a circuit different from that to which the receiver is connected
- Consult the dealer or an experienced radio/TV technician for help

Changes or modifications not expressly approved by Gemalto could void the user's authority to operate the equipment.

Canada

This class B digital apparatus meets all requirements of the Canadian interference- causing equipment regulations.

Europe

This product is in conformity with the protection requirements of EC Council Directive 2004/108/EC. Conformity is declared to the following applicable standards for electro-magnetic compatibility immunity and susceptibility; CISPR22 and IEC801. This product satisfies the CLASS B limits of EN 55022.

CONTENTS

PREFACE	About the Administration Guide	15
Customer Release Notes		16
Gemalto Rebranding		16
Audience		16
Document Conventions		17
Notes		17
Cautions		17
Warnings		17
Command syntax and typeface conventions		17
Support Contacts		18
1	Audit Logging	20
Audit Logging Overview		20
Audit Role on HSM		21
Audit Role Available Commands		21
Audit Logging Features		22
Audit Log Secret		23
Audit Log Records		23
Audit Log Message Format		24
Synchronizing Time between HSM and Host		25
Log Secret and Log Verification		25
Capacity		25
Time Reported in Log		26
Configuration Persists		26
Audit Logging Stops Working if the Current Log File is Deleted		26
Configuring and Using Audit Logging		26
Configure Audit Logging for SafeNet PCIe HSM or SafeNet USB HSM		27
Audit Log Operational Activities		28
Deciphering the audit log records		29
Additional Considerations		29
Audit Logging General Advice and Recommendations		29
Disk Full		31
Audit Log Categories and HSM Events		31
HSM Access		31
Log External		32
HSM Management		32
Key Management		33
Key Usage and Key First Usage		35
Audit Log Management		35
Verifying the Log Entries for Another HSM		36
2	Backup and Restore	37
Backup and Restore in General		37

Backup options	37
HSM or Partition	37
Other Info	38
Backup Your HSM Contents	38
Cloning Backup to another HSM	38
Additional Notes	39
Backup (Clone) Your HSM Partition	39
3 Capabilities and Policies	40
HSM Capabilities and Policies	40
Partition Capabilities and Policies	44
4 Configuration File Summary	50
5 Domains	61
Single Domain Policy	61
Legacy Domains and Migration	62
6 Error Codes and Troubleshooting	66
General Troubleshooting Tips	66
Remote PED	66
System Operational and Error Messages	67
Why do I often see extra slots that say "token not present"?	67
Error: 'hsm update firmware' failed. (10A0B : LUNA_RET_OPERATION_RESTRICTED) when attempting to perform hsm update firmware?	67
KR_ECC_POINT_INVALID Error when decrypting a file encrypted from BSAFE through ECIES using ECC key with any of the curves from the x9_t2 section.	67
Slow/interrupted response from the HSM, and the "hsm show" command shows LUNA_RET_SM_SESSION_REALLOC_ERROR	68
Keycard and Token Return Codes	68
Library Codes	83
Vendor-defined Return Codes	88
7 High-Availability (HA) Configuration and Operation	93
High Availability (HA) Overview	93
How HA is Implemented	94
Example: Database Encryption	94
Conclusion	95
Load Balancing	96
Key Replication	97
Manual Synchronization	98
Failover	98
Reaction to Failures	99
Recovery and Reconnection	101
Recovery after failure of a member	101
HA Automatic add/remove feature	106
Solaris (and other Unix)	107
HA Auto-Reconnect	107

Active Automatic recovery (the way it was)	107
Preserving HA context (the HA Auto Reconnect way)	108
What do the HA Auto Reconnect options mean?	108
What about timeouts and retries?	108
When to use HA Auto Reconnect	109
Steps to use HA Auto Reconnect	109
Performance	110
Maximizing Performance	110
Standby Members	112
Planning Your Deployment	116
HA Group Members	116
High Availability Group Sizing	117
Network Requirements	118
Upgrading and Redundancy and Rotation	118
Configuring HA	119
Create the HA Group	119
Verification Steps	125
HA Standby Mode [optional]	125
Using HA With Your Applications	125
HAOnly	125
Authentication	126
Key Generation	126
Application Object Handles	127
HA and FindObjects	127
Managing and Troubleshooting Your HA Groups	128
Slot Enumeration	128
Determining Which Device is in Use	129
Determining Which Devices are Active	129
Duplicate Objects	129
Adding, Removing, Replacing, or Reconnecting HA Group Members	130
Adding or Removing an HA Group Member	130
Reconnecting an Off-line Unit	130
Replacing a Failed SafeNet Network HSM	132
Replace a SafeNet Network HSM Using the same IP	134
Summary	135
Client-side - Reconfigure HA if a SafeNet Network HSM Must Be Replaced	136
Replacing the Secondary HA Group Member	139
Frequently Asked Questions	140
8 HSM Initialization	141
Initialization Overview for Password-Authenticated HSMs	141
Hard Initialization	141
Initializing	142
Logging In, Once You Have Initialized	142
Soft Initialization	142
Why choose Hard Init or Soft Init?	143
Initialization Overview for PED-authenticated HSMs	143
Hard Initialization	144
Initializing	144

Logging In, Once You Have Initialized	145
Soft Initialization	146
Why choose Hard Init or Soft Init?	146
HSM Initialization and Zeroization	146
Additional Notes	147
Re-initialize an HSM	147
Initialize an HSM With Existing Domain and Shared PED Keys	147
9 HSM Partitions	149
HSM Partitions	149
Partition Creation with Policy Template Using LunaCM	150
Process for a New Template	151
Modify a partition template, then apply the modified partition template	156
Delete a partition policy template	161
Separation of HSM Workspaces	162
Legacy Application Partitions	163
PPSO Application Partitions	163
Operation	163
Key Management Commands	164
Normal Usage Commands	164
Unauthenticated Commands	165
Commands That are Valid Only in a Session, But Require Special Handling	166
Configured and Registered Client Using an HSM Partition	166
Simple Troubleshooting	167
About Activation and Auto-Activation	167
Activation	168
AutoActivation	169
De-Activate a Partition	169
Removing Partitions	170
Security of Your Partition Challenge	170
How Secure Is the Challenge Secret or Password?	171
Frequently Asked Questions	172
Why do I get an error when I attempt to set the partition policies for activation (22) and auto-activation (23) on my password authenticated SafeNet Network HSM?	172
So, what is the difference in security, once Activation and Auto-activation are started?	172
10 HSM Status Values	174
11 Key Migration	177
Key Migration Procedures	177
Migrating Key Material from Older (2U) to New (1U) Appliances	177
Frequently Asked Questions	177
We want to generate keys on one HSM and copy them to other HSMs.Can they have the same object handles?	177
We want to migrate from a Microsoft Certificate Authority to a Linux CA while keeping the same private key. Does the SafeNet HSM offer any barriers to doing this?	178
We want to migrate from a Microsoft IIS with existing keys stored in software to CSP with keys stored in the SafeNet HSM. We have followed the steps to import using ms2luna utility, but it appears that IIS is still using the keys in software.	178

12 PED Authentication 179

About the SafeNet PED	179
PED Features	179
Using the PED	180
Interaction with Other Operations	181
Versions	182
Authentication	183
Local and Remote	183
When Do I Need A PED?	184
What Do I Do?	184
Standalone or local or off-line PED operations	185
EXCEPTION: Secure Recovery	186
EXCEPTION: Remote PED	186
About PED Keys	186
Why do you need PED Keys?	187
Types of PED Key	187
What is a Set of PED Keys?	191
Physical Identification of PED Keys	193
Using PED Keys	194
Compare Password and PED Authentication	194
What is a PED PIN?	195
But what is it?	196
How to invoke/require a PED PIN with an HSM	198
Must I Use a PED PIN?	202
Should I Use a PED PIN?	202
What If I Change My Mind?	202
Does that apply to the other PED Key colors?	203
What is a Shared or Group PED Key?	203
What else do I need to know?	203
Best Practice	204
How to Use a SafeNet PED	204
SafeNet PED Keypad Functions	205
Interaction between the HSM and the PED	206
How it was - versus - how it is today	210
Restating the "obvious"?	212
Duplicating PED keys	212
Lost PED Keys or PED PINs, or passwords	214
Help! I have lost my blue/black/red/orange/purple/white PED Key or I have forgotten the password!	214
But I don't have keys or secrets in secure on-site or off-site storage! What do I do?	214
I have my PED Key, but I forgot my PED PIN! What can I do?	216
I have my PED Keys and my PED PINS, but I can't remember which one goes with which HSM (or partition)!	217

13 PED Key Management 218

PED Key Management Overview	218
"Possible" Does Not Mean "Necessary"	218
PED Keys and Operational Roles	221
Actions That Require a PED Key	223
Shared or Group PED Keys	226

How does it work?	226
The Exception	227
Domain PED Keys	228
The "New Domain" Question	228
To What Does a Domain Apply?	230
What about Legacy HSMs and Partitions?	230
Summary	230
Duplicating PED Keys	230
Considerations for Duplicate PED Keys	231
How Many PED Keys Do I Need?	232
Basic amount for operation	232
One PED Key for many HSMs - grouping or reuse	233
Many PED Keys for one HSM - MofN	236
Combining MofN with Grouped/Unique Authentication	238
Calculate PED Key requirements for some or all HSM authentication secrets	241
Maintaining Your PED Keys	242
Conclusion	242
Using MofN	243
Typical Practice	243
Common MofN Usage	244
MofN and PED PINs	244
Revoking Means Re-initializing	245
How to Add an MofN Requirement Where There Was No MofN Before	245
Implementation Suggestions for backup MofN sets	246
Make one big MofN set, instead of three small sets	247
Complexity When Managing PED Keys	247
General Advice on PED Key Handling	248
Keep a Log	248
Apply Meaningful Labels	248
Keys	249
Updating PED Keys – Example	249
Risk of Losing access	249
PIN-change Procedure for Multiple HSMs	250
Updating PED Key for a Backup Token	253
Frequently Asked Questions	254
How should SafeNet PED Keys(*) be stored? (*Model iKey 1000 for use with SafeNet PED2)	254
So I shouldn't keep all the PED Keys for all my SafeNet HSMs in one box in a desk drawer?	254
I've lost my purple PED Key. Or, I forgot my PED PIN for my purple PED Key.	254
Do we really need to include a PED PIN with each PED Key?	255
14 Performance	256
Performance Overview	256
HA Performance	257
HSM Information Monitor	257
Notes about Monitor/Counter Behavior	257
Performance and the PE1746Enabled Setting	258
Effect on HA	258
Resetting the Internal SafeNet Network HSM PE1746Enabled Setting Following an Upgrade	258
Frequently Asked Questions	258

Can I buy a SafeNet Network HSM 1700 and later upgrade it to SafeNet Network HSM 7000?	259
Can you highlight the relative performance figures?	259
How can I achieve the kinds of performance numbers you quote?	259
Do you have any additional advice on how to interpret performance numbers? We're trying to match against a set of performance requirements that are stated as "signings per millisecond".	260
We expect to generate millions of keys per year. What is the expected number of read/write operations that your HSM memory can perform before the solid-state memory begins to fail?	260
In the "key factory" scenario, we need to generate approximately 30 ECC P224 keys/second. How many SafeNet Enterprise HSMs will we require?	260
15 Public Key Infrastructure (PKI) and Removable HSMs	261
PKI with SafeNet Network HSM	261
What to Do	261
HA	263
Using SafeNet USB HSM or Token-format HSM with SafeNet Network HSM Appliance	263
Constraints	264
PKI and HA	265
Card Reader (SafeNet DOCK 2) and Token-style HSMs	266
Frequently Asked Questions	269
We operate a Managed PKI and must satisfy our auditors that the root and intermediate keys and certs are protected according to an accepted standard, including when cloned/backed-up.	269
16 Remote PED	270
About Remote PED	270
Why do I want it?	271
How does it work?	271
One-to-One Remote PED Connections	273
Legacy vs Peer-to-peer Remote PED Connections	273
Constraints	274
Certificate exchange and port	275
Security	275
Configuration file	275
Priority and Lockout	276
Remote PED Timeout	276
Ports	277
Windows 7	277
Limitations	279
Compatibility	279
Security of Remote PED	279
Remote PED and pedclient and pedserver	279
Security of Remote PED	280
Multiple HSMs and Remote PED	280
Configuring Remote PED	280
You will need:	281
Configuring the PEDClient and PEDServer	281
Relinquishing Remote PED	286
Troubleshooting	287
Using the Remote PED Feature	289
Prepare a Remote PED Vector	290

Using Client-initiated Remote PED Connection	294
Using Server-initiated (Peer-to-Peer) Remote PED Connection	298
Constraints	299
Configuration Prerequisite	299
Physical setup for server-initiated Remote PED connection	299
Setting up the server-initiated Remote PED connection	300
pedServer Configuration File	302
Troubleshooting Remote PED	303
Ped connect can fail if IP is not accessible	303
VPN	304
Timeout	305
Pedserver fails to start with "LOGGER_init failed"	305
Remote PED Architecture	306
17 Removing/Destroying Content for Safe Disposal	308
Tamper/Declassify/Decommission the Thales Luna USB HSM	308
End of service and disposal	309
Needs Can Differ	309
SafeNet HSM Protects Your Keys and Objects	309
Comparison of Destruction/Denial Actions	309
RMA and Shipping Back to SafeNet	312
What Does Zeroized Mean?	313
18 User and Password Administration	314
About Changing HSM and Partition Passwords	314
HSM Passwords	315
Partition Passwords	315
Failed Logins and Forgotten Passwords	316
Appliance	316
Failed Logins	316
HSM Response When You Reach the Bad-attempt Threshold	317
Control the Outcome	318
Resetting Passwords	318
HSM	318
Partition	320
Default Challenge Password	321
19 Security Effects of Administrative Actions	322
Overt Security Actions	322
Actions with Security- and Content-affecting Outcomes	322
Elsewhere	322
Summary of Outcomes of Security-affecting Actions	323
Factory Reset HSM With Firmware <6.22.0	323
Factory Reset HSM With Firmware ≥6.22.0	323
Zeroize HSM With Firmware ≥6.22.0	324
Change Destructive HSM Policy	324
Apply Destructive CUF Update	324
HSM Initialize When Admin Not Initialized	325
HSM Initialize When Admin Initialized	325

Non-Admin Partition Initialize When the Partition is Not Initialized	326
Non-Admin Partition Initialize When the Partition is Initialized	326
20 Secure Transport Mode	327
MTK and SRK	327
Tamper and Recover with Purple Key NOT Enabled	327
Tamper and Recover with Purple PED Key Enabled	328
Behavior with Purple PED Key Enabled but MISSING or DAMAGED	328
Secure Transport Mode	329
Make a New Purple PED Key (SRK external split)	329
Master Key must be present	330
What if the purple SRK has been lost?	330
Disabling SRK	330
Compare and Contrast Some "Denial" and Destructive Scenarios	330
Secure Transport Mode [Local]	331
Backup	331
Recovery	332
No Re-split?	332
Additional Notes	333
Re-Split Required	334
Security	334
21 Slot Numbering and Behavior	335
Order of Occurrence for Different SafeNet HSMs	335
Settings Affecting Slot Order	336
Effects of Settings on Slot List	336
Effects of New Firmware on Slot Login State	337
22 SNMP Monitoring	338
Overview and Installation	338
MIB	338
SafeNet SNMP Subagent	339
Configuration Options In the luna-snmp.conf File	340
The SafeNet Chrysalis-UTSP MIB	340
The SAFENET HSM MIB	341
SNMP Table Updates	341
hsmTable	342
hsmLicenseTable	344
hsmPolicyTable	344
hsmPartitionPolicyTable	344
hsmClientRegistrationTable	344
hsmClientPartitionAssignmentTable	345
SNMP output compared to SafeNet tools output	345
Frequently Asked Questions	349
We want to use SNMP to remotely monitor and manage our installation – why do you not support such standard SNMP traps as CPU and Memory exhaustion?	349
23 Software Maintenance and Updates	350

About Updating SafeNet HSM	350
How Firmware Updates Affect Agency Validation	350
Updating the Client Software	351
Advanced Configuration Upgrades	352
ECIES Acceleration	355
Rollback Behavior	355
Apply a Capability Upgrade/Update to HSM	357
Preparing to Upgrade	357
Installing the Upgrade Package	357
Firmware Rollback	359
Serial Number Handling	361
24 Standards and Validations	363
About FIPS Validation	363
What does this mean to me?	364
About HSM NOT in FIPS140-2 Approved Mode	364
The FIPS-Approved Algorithms	364
What Does This Mean For Your Application?	365
What Are the Implications of Changing This Policy Setting?	365
Migrating from Non-FIPS HSM to FIPS HSM	365
SafeNet Network HSM or SafeNet PCIe HSM application partition (f/w 6.22.0 or newer) in LunaCM	369
NIST SP 800-131A: Changes to FIPS-Supported Algorithms	377
Summary 2014	378
Affected Algorithms	378
Impact on your operations	379
Mechanisms Affected	380
Summary 2016 (Triple DES)	382
Other Effects	383
Modification to DES3 Algorithm for NIST Compliance (2015)	384
UPDATE Affecting Triple DES (SP 800-67)	384
Common Criteria	384
Background	385
Trade-offs	385
So, What Are the Options?	386

PREFACE

About the Administration Guide

This document describes the operational and administrative tasks you can perform to maintain the functionality and efficiency of your HSMs. It contains the following chapters:

- ["Audit Logging" on page 20](#)
- ["Backup and Restore" on page 37](#)
- ["Capabilities and Policies" on page 40](#)
- ["Configuration File Summary" on page 50](#)
- ["Domains" on page 61](#)
- ["Error Codes and Troubleshooting" on page 66](#)
- ["High-Availability \(HA\) Configuration and Operation" on page 93](#)
- ["Host Trust Link Client Authentication" on page 1](#)
- ["HSM Initialization" on page 141](#)
- ["HSM Partitions" on page 149](#)
- ["HSM Status Values" on page 174](#)
- ["Key Migration" on page 177](#)
- ["PED Authentication" on page 179](#)
- ["PED Key Management" on page 218](#)
- ["Performance" on page 256](#)
- ["Public Key Infrastructure \(PKI\) and Removable HSMs" on page 261](#)
- ["Remote PED" on page 270](#)
- ["Removing/Destroying Content for Safe Disposal" on page 308](#)
- ["User and Password Administration" on page 314](#)
- ["Security Effects of Administrative Actions" on page 322](#)
- ["Secure Transport Mode" on page 327](#)
- ["Slot Numbering and Behavior" on page 335](#)
- ["SNMP Monitoring" on page 338](#)
- ["Software Maintenance and Updates" on page 350](#)
- ["Standards and Validations" on page 363](#)

This preface also includes the following information about this document:

- ["Customer Release Notes" on the next page](#)
- ["Gemalto Rebranding" on the next page](#)

- ["Audience" below](#)
- ["Document Conventions" on the next page](#)
- ["Support Contacts" on page 18](#)

For information regarding the document status and revision history, see ["Document Information" on page 2](#).

Customer Release Notes

The customer release notes (CRN) provide important information about this release that is not included in the customer documentation. Read the CRN to fully understand the capabilities, limitations, and known issues for this release. You can view or download the latest version of the CRN for this release at the following location:

- http://www.securedby safenet.com/releasenotes/luna/crn_luna_hsm_6-3.pdf

Gemalto Rebranding

In early 2015, Gemalto completed its acquisition of SafeNet, Inc. As part of the process of rationalizing the product portfolios between the two organizations, the Luna name has been removed from the SafeNet HSM product line, with the SafeNet name being retained. As a result, the product names for SafeNet HSMs have changed as follows:

Old product name	New product name
Luna SA HSM	SafeNet Network HSM
Luna PCI-E HSM	SafeNet PCIe HSM
Luna G5 HSM	SafeNet USB HSM
Luna PED	SafeNet PED
Luna Client	SafeNet HSM Client
Luna Dock	SafeNet Dock
Luna Backup HSM	SafeNet Backup HSM
Luna CSP	SafeNet CSP
Luna JSP	SafeNet JSP
Luna KSP	SafeNet KSP



Note: These branding changes apply to the documentation only. The SafeNet HSM software and utilities continue to use the old names.

Audience

This document is intended for personnel responsible for maintaining your organization's security infrastructure. This includes SafeNet HSM users and security officers, key manager administrators, and network administrators.

All products manufactured and distributed by Gemalto are designed to be installed, operated, and maintained by personnel who have the knowledge, training, and qualifications required to safely perform the tasks assigned to them. The information, processes, and procedures contained in this document are intended for use by trained and qualified personnel only.

It is assumed that the users of this document are proficient with security concepts.

Document Conventions

This document uses standard conventions for describing the user interface and for alerting you to important information.

Notes

Notes are used to alert you to important or helpful information. They use the following format:



Note: Take note. Contains important or helpful information.

Cautions

Cautions are used to alert you to important information that may help prevent unexpected results or data loss. They use the following format:



CAUTION: Exercise caution. Contains important information that may help prevent unexpected results or data loss.

Warnings

Warnings are used to alert you to the potential for catastrophic data loss or personal injury. They use the following format:



WARNING! Be extremely careful and obey all safety and security measures. In this situation you might do something that could result in catastrophic data loss or personal injury.

Command syntax and typeface conventions

Format	Convention
bold	<p>The bold attribute is used to indicate the following:</p> <ul style="list-style-type: none"> • Command-line commands and options (Type <code>dir /p</code>.) • Button names (Click Save As.) • Check box and radio button names (Select the Print Duplex check box.) • Dialog box titles (On the Protect Document dialog box, click Yes.) • Field names (User Name: Enter the name of the user.)

Format	Convention
	<ul style="list-style-type: none"> Menu names (On the File menu, click Save.) (Click Menu > Go To > Folders.) User input (In the Date box, type April 1.)
<i>italics</i>	In type, the italic attribute is used for emphasis or to indicate a related document. (See the <i>Installation Guide</i> for more information.)
<variable>	In command descriptions, angle brackets represent variables. You must substitute a value for command line arguments that are enclosed in angle brackets.
[optional] [<optional>]	Represent optional keywords or <variables> in a command line description. Optionally enter the keyword or <variable> that is enclosed in square brackets, if it is necessary or desirable to complete the task.
{a b c} {<a> <c>}	Represent required alternate keywords or <variables> in a command line description. You must choose one command line argument enclosed within the braces. Choices are separated by vertical (OR) bars.
[a b c] [<a> <c>]	Represent optional alternate keywords or variables in a command line description. Choose one command line argument enclosed within the braces, if desired. Choices are separated by vertical (OR) bars.

Support Contacts

Contact method	Contact	
Phone (Subject to change. An up-to-date list is maintained on the Technical Support Customer Portal)	Global	+1 410-931-7520
	Australia	1800.020.183
	India	000.800.100.4290
	Netherlands	0800.022.2996
	New Zealand	0800.440.359
	Portugal	800.863.499
	Singapore	800.1302.029
	Spain	900.938.717
	Sweden	020.791.028
	Switzerland	0800.564.849
	United Kingdom	0800.056.3158
	United States	(800) 545-6608
Web	https://safenet.gemalto.com	

Contact method	Contact
Technical Support Customer Portal	https://supportportal.gemalto.com Existing customers with a Technical Support Customer Portal account can log in to manage incidents, get the latest software upgrades, and access the Knowledge Base. To create a new account, click the Register link at the top of the page. You will need your Customer Identifier number.

Audit Logging

This chapter describes how to use audit logging to provide security audits of HSM activity. It contains the following sections:

- ["Audit Logging Overview" below](#)
- ["Configuring and Using Audit Logging" on page 26](#)
- ["Audit Logging General Advice and Recommendations" on page 29](#)
- ["Audit Log Categories and HSM Events" on page 31](#)
- ["Verifying the Log Entries for Another HSM" on page 36](#)
- ["Remote Audit Logging " on page 1](#)

Audit Logging Overview

The HSM can log its events with varying levels of verbosity (like critical events only, failures, failures and successes, key usage, all). By default, only critical events are logged, which imposes virtually no load on the HSM.

In addition, you have the option to create the Audit user on the HSM and specify that log entries are secured for writing off the HSM. As above, the level/frequency of that logging can be specified by setting the logging threshold.

Or, you can just specify that entries are written to the SafeNet Network Appliance file system without verification security.

Beginning with release 5.2, SafeNet HSMs consolidate and enhance auditing of HSM operations in the Standard Audit Logging feature.

- This method is the most secure and verifiable, to satisfy the strictest auditing requirements.
- This method places the highest Audit-related load on the HSM.

Beginning with release 6.2.1 in general release (and 5.3.10 in limited release), SafeNet Network HSM offers direct-to-file, or appliance-side logging of HSM activity.

- This method preserves the HSM activity audit records and offloads them to the file system for further processing or off-board storage, without securing or validating the resulting records or files.
- This method imposes substantially less processing load on the HSM, compared with the fully secured logging method (above).

The audit logging feature works only with hardware and software update levels at, or newer than, the versions that introduced the feature:

Hardware

- SafeNet USB HSM

Software

- Client Software 5.2.0 or newer

- HSM Firmware 6.10.x or newer
- SafeNet PED 2.5.0-2 or newer (if PED-authenticated HSM)

Audit Role on HSM

A SafeNet HSM Audit role allows complete separation of Audit responsibilities from the Security Officer (SO or HSM Admin), the Partition User (or Owner), and other HSM roles. If the Audit role is initialized, the HSM and Partition administrators are prevented from working with the log files, and auditors are unable to perform administrative tasks on the HSM. As a general rule, the Audit role should be created before the HSM Security Officer role, to ensure that all important HSM operations (including those that occur during initialization), are captured.

Password-authenticated HSMs

For SafeNet HSMs with Password Authentication, the auditor role logs into the HSM to perform their activities using a password.

PED-authenticated HSMs

For SafeNet HSMs with PED Authentication, the auditor role logs into the HSM to perform their activities using the Audit (white) PED Key. The Audit feature works only with SafeNet PED version 2.5.0-3 or newer. Older versions of PED firmware are not aware of the Audit role and Audit Key.

Role Initialization

Creating the Audit role (and imprinting the white PED Key for PED authenticated HSMs) does not require the presence or cooperation of the HSM SO.

Audit Role Available Commands

In Lunacm, all commands are visible to the person who launches the utility, and some can be used without specific authentication to the HSM, such as view/show/list commands, which might be classified as "monitoring" functions. Attempts to run operational or administrative commands that need role-specific authentication, without that authentication, result in an error message. The Audit role has a limited set of operations available to it, on the HSM, which constitutes any of the generally accessible monitoring commands, plus everything under the "audit" heading.

```
lunacm:>audit
```

The following sub commands are available:

Command	Short	Description
init	i	Initialize the Audit role
changePwd	ch	Change Audit User Password or PED Key
login	logi	Login as the Audit user
logout	logo	Logout the Audit user
config	co	Set Audit Parameters
sync	sy	Synchronize HSM Time to Host Time
show	sh	Display the Audit logging info
log	l	> Manage Audit Log Files
secret	se	> Export/Import Audit Logging Secret
remotehost	r	> Configure Audit Logging Remote Hosts

Syntax: `audit <sub command>`

Command Result : No Error

lunacm:>

Anyone accessing the computer and running lunacm can see the above commands, but cannot run them if they do not have the "audit" role authentication (password or PED Key, as appropriate).

What is important is not the role you can access on the computer (a named user, admin, root), but the role you can access within the HSM.

Audit Logging Features

The following list summarizes the functionality of the audit secure logging feature:

- Log entries originate from the SafeNet HSM - the feature is implemented via HSM firmware (rather than in the library) for maximum security
- Log origin is assured
- Logs and individual records can be validated by any SafeNet HSM that is a member of the same domain
- The audit logging feature is applicable to password-authenticated (FIPS 140-2 level 2) and to PED-authenticated (FIPS 140-2 level 3) product configurations (but not between the two - see the "same domain" requirement, above)
- Each entry includes the following:
 - when the event occurred
 - who initiated the event (the authenticated entity)
 - what the event was
 - the result of the logging event (success, error, etc.)
- Multiple categories of audit logging are supported, configured by the audit role
- Audit management is a separate role - the role creation does not require the presence or co-operation of the SafeNet HSM SO
- The category of audit logging is configurable by (and only by) the audit role
- Audit log integrity is ensured against the following:
 - Truncation - erasing part of a log record
 - Modification - modifying a log record
 - Deletion - erasing of the entire log record
 - Addition - writing of a fake log record
- Log origin is assured
- The following critical events are logged unconditionally, regardless of the state of the audit role (initialized or not):
 - Tamper
 - Zeroization
 - SO creation

- Audit role creation

Audit Log Secret

The HSM creates a log secret unique to the HSM, computed during the first initialization after manufacture. The log secret resides in flash memory (permanent, non-volatile memory), and is used to create log records that are sent to a log file. Later, the log secret is used to prove that a log record originated from a legitimate HSM and has not been tampered with.

Audit Log Records

A log record consists of two fields – the log message and the HMAC for the previous record. When the HSM creates a log record, it uses the log secret to compute the SHA256-HMAC of all data contained in that log message, plus the HMAC of the previous log entry. The HMAC is stored in HSM flash memory. The log message is then transmitted, along with the HMAC of the previous record, to the host. The host has a logging daemon to receive and store the log data on the host hard drive.

For the first log message ever returned from the HSM to the host there is no previous record and, therefore, no HMAC in flash. In this case, the previous HMAC is set to zero and the first HMAC is computed over the first log message concatenated with 32 zero-bytes. The first record in the log file then consists of the first log message plus 32 zero-bytes. The second record consists of the second message plus HMAC1 = HMAC (message1 || 0x0000). This results in the organization shown below.

MSG 1	HMAC 0
	...
MSG n-1	HMAC n-2
MSG n	HMAC n-1
...	
MSG n+m	HMAC n+m-1
MSG n+m+1	HMAC n+m
...	
MSG end	HMAC n+m-1
Recent HMAC in NVRAM	HMAC end

To verify a sequence of m log records which is a subset of the complete log, starting at index n , the host must submit the data illustrated above. The HSM calculates the HMAC for each record the same way as it did when the record was originally generated, and compares this HMAC to the value it received. If all of the calculated HMACs match the received HMACs, then the entire sequence verifies. If an HMAC doesn't match, then the associated record and all following records can be considered suspect. Because the HMAC of each message depends on the HMAC of the previous one, inserting or altering messages would cause the calculated HMAC to be invalid.

- The “who” is lunash session “session 1 Access 2147483651:22621” (identified by the lunash access ID major = 2147483651, minor = 22621).
- The “what” is “LUNA_CREATE_CONTAINER”.
- The operation status is “LUNA_RET_SM_UNKNOWN_TOSM_STATE(0x00300014)”.



Note: Log Rotation Categories, Rotation Intervals, and other Configurable Factors are covered here in the Administration & Maintenance Manual. Command syntax is in the Reference Manual.

Synchronizing Time between HSM and Host

The HSM has an internal real-time clock (RTC). The RTC does not have a relevant time value until it is synchronized with the HOST system time. Because the HSM and the host time could drift apart over time, periodic re-synchronization is necessary. Only an authenticated audit officer is allowed to synchronize the time.

Log Secret and Log Verification

The 256-bit log secret which is used to compute the HMACs is stored in the parameter area on the HSM. It is set the first time an event is logged. It can be exported from one HSM to another so that a particular sequence of log messages can be verified by the other HSM. Conversely, it can be imported from other HSMs for verification purpose.

To accomplish cross-HSM verification, the HSM generates a key-cloning vector (KCV, a.k.a the Domain key) for the audit role when it is initialized. The KCV can then be used to encrypt the log secret for export to the HOST.

To verify a log that was generated on another HSM, assuming it is in the same domain, we simply import the wrapped secret, which the HSM subsequently decrypts; any records that are submitted to the host for verification will use this secret thereafter.

When the HSM exports the secret, it calculates a 32-bit checksum which is appended to the secret before it is encrypted with the KCV.

When the HSM imports the wrapped secret, it is decrypted, and the 32-bit checksum is calculated over the decrypted secret. If this doesn't match the decrypted checksum, then the secret that the HSM is trying to import comes from a system on a different domain, and an error is returned.

To verify a log generated on another HSM, in the same domain, the host passes to the target HSM the wrapped secret, which the target HSM subsequently decrypts; any records submitted to the target HSM for verification use this secret thereafter.

Importing a log secret from another HSM does not overwrite the target log secret because the operation writes the foreign log secret only to a separate parameter area for the wrapped log secret.



CAUTION: Once an HSM has imported a wrapped log secret from another HSM, it must export and then re-import its own log secret in order to verify its own logs again.

Capacity

The log capacity of SafeNet HSMs varies depending upon the physical memory available on the device. The SafeNet PCIe HSM and the HSM contained in the SafeNet Network HSM appliance are the SafeNet K6 HSM card. The HSM

inside both the SafeNet USB HSM and the SafeNet Remote Backup HSM is the SafeNet G5 HSM module.

The K6 HSM has approximately 16 MB available for Audit logging (or more than 200,000 records, depending on the size/content of each record).

The G5 HSM has approximately 4 MB available for Audit logging (or more than 50,000 records, depending on the size/content of each record).

In both cases, the normal function of Audit Logging is to export log entries constantly to the file system. Short-term, within-the-HSM log storage capacity becomes important only in the rare situations where the HSM remains functioning but the file system is unreachable from the HSM. This would be a rare or unlikely event for an HSM connected to a server or workstation, and almost unheard-of in the closed and hardened environment of a SafeNet Network HSM appliance.

Time Reported in Log

When you perform audit time get you might see a variance of a few seconds between the reported HSM time and the Host time. Any difference up to five seconds should be considered normal, as the HSM reads new values from its internal clock on a five-second interval. So, typically, Host time would show as slightly ahead.

Configuration Persists

Audit Logging configuration is not removed or reset upon HSM re-initialization. It survives tamper and factory reset. Logs must be cleared by specific command. Therefore, if your security regime requires at end-of-life, or prior to shipping an HSM, then explicit clearing of HSM logs should be part of that procedure.

This is by design, as part of separation of roles in the HSM. When the Audit role exists, the SO cannot modify the logging configuration, and therefore cannot hide any activity from auditors.

Audit Logging Stops Working if the Current Log File is Deleted

As a general rule, you should not delete a file while it is open and in use by an application. In most systems, deletion of a file is deletion of an inode, but the actual file itself, while now invisible, remains on the file system until the space is cleaned up or overwritten. If a file is in use by an application - such as audit logging, in this case - the application can continue using and updating that file, unaware that it is now in deleted status.

If you delete the current audit log file, the audit logging feature does not detect that and does not create a new file, so you might lose log entries.

The workaround is to restart the pedClient daemon, which creates a new log file.

Configuring and Using Audit Logging

The overall sequence when initializing an HSM that will use Security Audit Logging is as follows:

1. **Configure the SafeNet Network HSM appliance or the SafeNet PCIe HSM/SafeNet USB HSM host workstation to use the network time protocol (NTP).**

Configure access to at least two geographically separated NTP servers for redundancy. Select at least one NTP server that is known to have a high degree of accuracy and reliability (servers associated with national standards bodies are good candidates) as one of the configured servers.

2. **Initialize the Audit Officer role.**

This enables logging for all subsequent actions performed by the SO and partition User(s).

3. Execute the 'audit sync' command.

This ensures that the HSM's clock is synchronized with the host time (which should also be synchronized with the NTP server) and that all subsequent log records will have a valid and accurate timestamp.

4. Configure the audit category and level of audit

You can specify the level of audit appropriate for needs of the organization's policy and the nature of the application(s) using the HSM. Security audits can generate a very large amount of data, which consumes HSM processing resources, host storage resources, and makes the job of the Audit Officer quite difficult when it comes time to review the logs. For this reason, ensure that you configure audit logging such that you capture only relevant data, and no more.

For example, the 'First Key Usage Only' category is intended to assist Audit Officers to capture the relevant data in a space-efficient manner for high processing volume applications. On the other hand, a top-level Certificate Authority would likely be required, by policy, to capture all operations performed on the HSM but, since it is typically not an application that would see high volumes, configuring the HSM to audit all events would not impose a significant space and/or performance premium in that situation.

5. Configure log rotation and remote logging server(s) as necessary.

The settings for these configuration elements will often be dictated by the organization's Audit and/or IT policies and procedures. As with configuring the audit category, the Audit Officer should be prudent in making these configuration settings. It is recommended that the default setting of 'Rotate Log Daily' be maintained until the typical/average logging rate can be determined. The use of redundant remote log servers, accessible only by the members of the audit team, is strongly recommended.

6. Initialize the HSM and create partitions as necessary.

At this point, the HSM is ready to be turned over to the SO to initialize it and begin creating the partitions needed to serve the processing applications.

Configure Audit Logging for SafeNet PCIe HSM or SafeNet USB HSM

This section describes how to prepare and use audit logging with your SafeNet HSM.

Required SafeNet HSM Client version is 5.2 or later; HSM firmware version is 6.10.9 or later.

In summary, the steps are:

- Initialize, to create the role on the HSM.
- Configure the various logging parameters.
- Begin collecting and verifying logs of HSM activities.

To configure audit logging

1. Set the slot focus to the HSM administrative partition of the desired HSM:

```
lunacm:>slot set slot <number>.
```

2. Initialize the **Auditor** role:

```
lunash:>role init -name Auditor
```

- On password-authenticated HSMs, you are prompted for a domain string and password
- On PED-authenticated HSMs, you are referred to SafeNet PED, which prompts for a white PED Key.

3. Now that the **Auditor** role exists on the HSM, the auditing function must be configured. However, before you can configure you must log in as the **Auditor** user:

```
lunacm:>role login -name Auditor
```

- On password-authenticated HSMs, you are prompted to enter the password for the **Auditor** user.
- On PED-authenticated HSMs, you are referred to SafeNet PED, which prompts for the white PED Key for the **Auditor** user.

4. Configure audit logging:

```
lunash:> audit config
```



Note: The first time you configure audit logging, we suggest using only the "?" option, in order to see all the available options in the configuration process.

Log Entries

Log entries are made within the HSM, and are written to the currently active log file on the appliance file system. When a log file reaches the rotation trigger, it is closed, and a new file gets the next log entry. The number log files on the appliance grows according to the logging settings and the rotation schedule that you configured (above). At any time, you can copy files to a remote computer and then clear the originals from the HSM, if you wish to free the space. For USB-connected external (SafeNet USB HSM) and PCI-bus internal HSMs (SafeNet PCIe HSM), the lunacm tool includes options to set the log file size and the log file path (which are then written to the config file).

Audit Log Operational Activities

```
lunacm:> audit export
```

5. Exit LunaCM and browse to see the filename of the wrapped log secret:

```
/user/safenet/lunaclient/bin :>cd ../../lunalog
/user/safenet/lunalog :>ls
123456 7001347 k6secret.bin LogSecret_130115210057_123456.lws
```

6. On the computer where the HSM is attached, that you will use to verify the downloaded audit log file, run:

```
/usr/safenet/lunaclient/bin :>scp audit@mylunasa1:151170.lws .
```

Substitute the actual file name of the exported secret in the above example command) and provide the audit user's credentials when prompted. This copies the identified file from the remote SafeNet Network HSM's file system (in the "audit" account) and stores the copy on your local computer file system in the directory from which you issued the command.

7. Launch LunaCM,

```
/usr/safenet/lunaclient/bin :>./lunacm
```

8. For this example, we will assume that you have initialized the HSM Audit User role, using the same domain/secret as is associated with the source SafeNet Network HSM.

9. Import the Audit Logging secret into the locally attached HSM:

```
lunacm:>audit import file 151170.lws
```

10. Verify the file:

```
lunacm:>audit verify file mylunsa1_audit_2014-02-28.tgz
```

You might need to provide the full path to the file, depending upon your current environment settings.

Deciphering the audit log records

In general, the audit logs are self-explanatory. Due to limitations in the firmware, however, some audit log records required further explanation, as detailed in the following sections:

Determining the serial number of a created partition from the audit log

An audit log entry similar to the following is generated when a partition is created on the HSM:

```
5,12/12/17 16:14:14,S/N 150718 session 1 Access 2147483651:2669 SO container operation LUNA_CREATE_CONTAINER returned RC_OK(0x00000000) container=20 (using PIN (entry=LUNA_ENTRY_DATA_AREA))
```

It is not obvious from this entry what the serial number is for the created partition. This information, however, can be derived from the log entry, since the partition serial number is simply a concatenation of the HSM serial number and the partition container number, which are specified in the log entry, as highlighted below:

```
5,12/12/17 16:14:14,S/N 150718 session 1 Access 2147483651:2669 SO container operation LUNA_CREATE_CONTAINER returned RC_OK(0x00000000) container=20 (using PIN (entry=LUNA_ENTRY_DATA_AREA))
```

In the example above, the HSM serial number is 150718 and the partition container number is 20. Note that the partition container number is a three-digit number with leading zeros suppressed, so that the actual partition container number is 020. To determine the partition serial number concatenate the two numbers as follows:

```
150718020
```

Use this number to identify the partition in subsequent audit log entries.

Additional Considerations

- The audit role PED key or password is a critical property to manage the audit logs. If that authentication secret is lost, the HSM must be factory reset (that is, zeroize the HSM) in order to initialize the audit role again. This is equivalent to the same situation for the HSM's Security Officer (SO).
- Multiple bad logins produce different results for the SO and for the audit role, as follows:
 - After 3 bad SO logins, the LUNA_RET_SO_LOGIN_FAILURE_THRESHOLD error is returned and the HSM is zeroized.
 - After 3 bad audit logins, the LUNA_RET_AUDIT_LOGIN_FAILURE_THRESHOLD error is returned, but the HSM is unaffected. If subsequent login attempt is executed within 30 seconds, the LUNA_RET_AUDIT_LOGIN_TIMEOUT_IN_PROGRESS error is returned. If you wait for more than 30 seconds and try login again with the correct password, the login is successful.

Audit Logging General Advice and Recommendations

The Secure Audit Logging feature can produce a significant volume of data. It is expected, however, that Audit Officers will configure it properly for their specific operating environments. The data produced when the feature has been properly configured might be used for a number of reasons, such as:

- maintaining an audit trail that can later be used to reconstruct a particular action or set of actions (i.e., forensics);
- maintaining an audit trail that can later be used to trace the actions of an application or individual user (i.e., accounting); and
- maintaining an audit trail that can later be used to hold a specific individual accountable for his/her actions (i.e., non-repudiation)

That last bullet point represents the ultimate conclusion of any audit trail – to establish an irrefutable record of the chain of events leading up to a particular incident for the purpose of identifying and holding accountable the individual responsible. Not every organization will want to use security audit to meet the strict requirements of establishing such a chain of events. However, all security audit users will want to have an accurate representation of a particular sequence of events. To ensure that the audit log does contain an accurate representation of events and that it can be readily interpreted when it is reviewed, these basic guidelines should be followed after the audit logging feature has been properly configured:

- Use a shell script to execute the audit sync command at least once every 24 hours, provided the host has maintained its connection(s) to its configured NTP server(s).
- Do not allow synchronization with the host's clock if the host has lost connectivity to NTP. This ensures that the HSM's internal clock is not set to a less accurate time than it has maintained internally. In general, the HSM's RTC will drift much less than the host's RTC and will, therefore, be significantly more accurate than the host in the absence of NTP.
- Review logs at least daily and adjust configuration settings if necessary. It is important that any anomalies be identified as soon as possible and that the logging configuration that has been set is effective. If possible, use the remote logging feature to transmit log data to a Security Information and Event Management (SIEM) system to automatically analyze log data and identify anomalous events.
- In the case of SafeNet Network HSM, execute the audit log tarlogs lush command regularly to archive the audit logs and transfer them to a separate machine for long term storage. Also, execute the 'audit log clear' lush command regularly to free up the audit log disk space on SafeNet Network HSM.
- Consider installing and configuring an HSM (for example, a SafeNet USB or PCIe HSM) connected to the remote log server to act as a "verification engine" for the remote log server. Ensure that the log secret for the operational HSM(s) has been shared with the log server verification HSM.

Note: This is not always possible, unless you are physically copying the logs over from the .tgz archive.



Because log records do not necessarily appear on the remote log server immediately, the HMAC might be incorrect. Also, if more than one SafeNet Network HSM is posting log records to a remote server, this could interfere with record counts.

- The audit log records are comma-delimited. We recommend that full use be made of the CSV formatting to import records into a database system or spreadsheet tool for analysis, if an SIEM system is not available.
- The ASCII hex data representing the command and returned values and error code should be examined if an anomaly is detected in log review/analysis. It may be possible to match this data to the HSM's dual-port data. The dual-port, if it is available, will contain additional data that could be helpful in establishing the context surrounding the anomalous event. For example, if an unexpected error occurs it could be possible to identify the trace through the firmware subsystems associated with the error condition. This information would be needed to help in determining if the error was unexpected but legitimate or if it was forced in an attempt to exploit a potential weakness (e.g., searching for buffer overflows).

An important element of the security audit logging feature is the 'Log External' function. See the SDK for more information "[Audit Logging](#)" on page 1. For applications that cannot add this function call, it is possible to use the lunacm command-line function 'audit log external' within a startup script to insert a text record at the time the application is started.

Disk Full

In the event that all the audit disk space is used up, audit logs are written to the HSM's small persistent memory. When the HSM's persistent memory is full, normal crypto commands will fail with "disk full" error.

To resolve that situation, the audit user must:

- archive the audit logs on the host side,
- move them to some other location for safe storage,
- clear the audit log directory, and
- restart the logger daemon.

To prevent the "disk full" situation, we recommend that the audit user should routinely archive the audit logs and clear the audit log directory.

Audit Log Categories and HSM Events

This section provides a summary of the audit log categories and their associated HSM events.

HSM Access

HSM Event	Description
LUNA_LOGIN	C_Login. This event must be allowed to proceed even if the result should be logged but cannot (for example, due to a log full condition).
LUNA_LOGOUT	C_Logout. This event must be allowed to proceed even if the result should be logged but cannot (for example, due to a log full condition).
LUNA_MODIFY_OBJECT	C_SetAttributeValue
LUNA_OPEN_SESSION	C_OpenSession. This event must be allowed to proceed even if the result should be logged but cannot (for example, due to a log full condition).
LUNA_CLOSE_ALL_SESSIONS	C_CloseAllSessions
LUNA_CLOSE_SESSION	C_CloseSession This event must be allowed to proceed even if the result should be logged but cannot (for example, due to a log full condition).
LUNA_OPEN_ACCESS	CA_OpenApplicationID
LUNA_CLEAN_ACCESS	CA_Restart, CA_RestartForContainer
LUNA_CLOSE_ACCESS	CA_CloseApplicationID
LUNA_LOAD_CUSTOM_	CA_LoadModule

HSM Event	Description
MODULE	
LUNA_LOAD_ENCRYPTED_CUSTOM_MODULE	CA_LoadEncryptedModule
LUNA_UNLOAD_CUSTOM_MODULE	CA_UnloadModule
LUNA_EXECUTE_CUSTOM_COMMAND	CA_PerformModuleCall
LUNA_HA_LOGIN	CA_HAGetLoginChallenge, CA_HAAnswerLoginChallenge, CA_HALogin, CA_HAAnswerMofNChallenge, HAActivateMofN

Log External

HSM Event	Description
LUNA_LOG_EXTERNAL	CA_LogExternal

HSM Management

HSM Event	Description
LUNA_ZEROIZE	CA_FactoryReset This event is logged unconditionally.
LUNA_INIT_TOKEN	C_InitToken This event is logged unconditionally.
LUNA_SET_PIN	C_SetPIN
LUNA_INIT_PIN	C_InitPIN
LUNA_CREATE_CONTAINER	CA_CreateContainer
LUNA_DELETE_CONTAINER	CA_DeleteContainer, CA_DeleteContainerWithHandle
LUNA_SEED_RANDOM	C_SeedRandom
LUNA_EXTRACT_CONTEXTS	C_GetOperationState
LUNA_INSERT_CONTEXTS	C_SetOperationState
LUNA_SELF_TEST	C_PerformSelfTest

HSM Event	Description
LUNA_LOAD_CERT	CA_SetTokenCertificateSignature
LUNA_HA_INIT	CA_HAInit
LUNA_SET_HSM_POLICY	CA_SetHSMPolicy
LUNA_SET_DESTRUCTIVE_HSM_POLICY	CA_SetDestructiveHSMPolicy
LUNA_SET_CONTAINER_POLICY	CA_SetContainerPolicy
LUNA_SET_CAPABILITY	Internal, for capability update
LUNA_CREATE_LOGIN_CHALLENGE	CA_CreateLoginChallenge
LUNA_REQUEST_CHALLENGE	CA_SIMInsert, CA_SIMMultiSign
LUNA_PED_INIT_RPV	CA_InitializeRemotePEDVector
LUNA_PED_DELETE_RPV	CA_DeleteRemotePEDVector
LUNA_MTK_LOCK	Internal, for manufacturing
LUNA_MTK_UNLOCK_CHALLENGE	Internal, for manufacturing
LUNA_MTK_UNLOCK_RESPONSE	Internal, for manufacturing
LUNA_MTK_RESTORE	CA_MTKRestore
LUNA_MTK_RESPLIT	CA_MTKResplit
LUNA_MTK_ZEROIZE	CA_MTKZeroize
LUNA_FW_UPGRADE_INIT	CA_FirmwareUpdate
LUNA_FW_UPGRADE_UPDATE	CA_FirmwareUpdate
LUNA_FW_UPGRADE_FINAL	CA_FirmwareUpdate
LUNA_FW_ROLLBACK	CA_FirmwareRollback
LUNA_MTK_SET_STORAGE	CA_MTKSetStorage
LUNA_SET_CONTAINER_SIZE	CA_SetContainerSize

Key Management

HSM Event	Description
LUNA_CREATE_OBJECT	C_CreateObject
LUNA_COPY_OBJECT	C_CopyObject
LUNA_DESTROY_OBJECT	C_DestroyObject

HSM Event	Description
LUNA_DESTROY_MULTIPLE_OBJECTS	CA_DestroyMultipleObjects
LUNA_GENERATE_KEY	C_GenerateKey
LUNA_GENERATE_KEY_PAIR	C_GenerateKeyPair
LUNA_WRAP_KEY	C_WrapKey
LUNA_UNWRAP_KEY	C_UnwrapKey
LUNA_DERIVE_KEY	C_DeriveKey
LUNA_GET_RANDOM	C_GenerateRandom
LUNA_CLONE_AS_SOURCE, LUNA_REPLICATE_AS_SOURCE	CA_CloneAsSource
LUNA_CLONE_AS_TARGET_INIT, LUNA_REPLICATE_AS_TARGET_INIT	CA_CloneAsTargetInit
LUNA_CLONE_AS_TARGET, LUNA_REPLICATE_AS_TARGET	CA_CloneAsTarget
LUNA_GEN_TKN_KEYS	CA_GenerateTokenKeys
LUNA_GEN_KCV	CA_ManualKCV, C_InitPIN, C_InitToken, CA_InitAudit
LUNA_SET_LKCV	CA_SetLKCV
LUNA_M_OF_N_GENERATE	CA_GenerateMofN_Common, CA_GenerateMofN
LUNA_M_OF_N_ACTIVATE	CA_ActivateMofN
LUNA_M_OF_N_MODIFY	CA_ActivateMofN
LUNA_EXTRACT	CA_Extract
LUNA_INSERT	CA_Insert
LUNA_LKM_COMMAND	CA_LKMInitiatorChallenge, CA_LKMReceiverResponse, CA_LKMInitiatorComplete, CA_LKMReceiverComplete.
LUNA_MODIFY_USAGE_COUNT	CA_ModifyUsageCount

Key Usage and Key First Usage

HSM Event	Description
LUNA_ENCRYPT_INIT	C_EncryptInit
LUNA_ENCRYPT	C_Encrypt
LUNA_ENCRYPT_END	C_EncryptFinal
LUNA_DECRYPT_INIT	C_DecryptInit
LUNA_DECRYPT	C_Decrypt
LUNA_DECRYPT_END	C_DecryptFinal
LUNA_DIGEST_INIT	C_DigestInit
LUNA_DIGEST	C_Digest
LUNA_DIGEST_KEY	C_DigestKey
LUNA_DIGEST_END	C_DigestFinal
LUNA_SIGN_INIT	C_SignInit
LUNA_SIGN	C_Sign
LUNA_SIGN_END	C_SignFinal
LUNA_VERIFY_INIT	C_VerifyInit
LUNA_VERIFY	C_Verify
LUNA_VERIFY_END	C_VerifyFinal
LUNA_SIGN_SINGLEPART	C_Sign
LUNA_VERIFY_SINGLEPART	C_Verify
LUNA_WRAP_CSP	CA_CloneMofN_Common
LUNA_M_OF_N_DUPLICATE	CA_DuplicateMofN
LUNA_ENCRYPT_SINGLEPART	C_Encrypt
LUNA_DECRYPT_SINGLEPART	C_Decrypt
LUNA_PE1746_COMMAND	Used when PE1746 is enabled

Audit Log Management

HSM Event	Description
LUNA_LOG_SET_TIME	CA_TimeSync

HSM Event	Description
LUNA_LOG_GET_TIME	CA_GetTime
LUNA_LOG_SET_CONFIG	CA_LogSetConfig This event must be allowed to proceed even if the result should be logged but cannot (for example, due to a log full condition).
LUNA_LOG_GET_CONFIG	CA_LogGetConfig This event must be allowed to proceed even if the result should be logged but cannot (for example, due to a log full condition).
LUNA_LOG_VERIFY	CA_LogVerify
LUNA_CREATE_AUDIT_CONTAINER **	CA_InitAudit The event is logged unconditionally.
LUNA_LOG_IMPORT_SECRET	CA_LogImportSecret
LUNA_LOG_EXPORT_SECRET	CA_LogExportSecret

Verifying the Log Entries for Another HSM

You can use one HSM to verify the audit log files/entries that were created by another HSM. You can only verify the logs that are stored in the **ready_for_archive** folder; you cannot verify log files that are currently being written.

To verify the log entries for another HSM

1. Export the secret on SafeNet HSM1 (`audit secret export`)
2. Tar logs on SafeNet HSM1 host (`audit log tar`)
3. Transfer the secret to SafeNet HSM2 (`scp`)
4. Transfer the archive to SafeNet HSM2 (`scp`)
5. Import the secret onto SafeNet HSM2 (`audit secret import -f <SafeNetHSM1_SN>.lws -serialtarget <SafeNetHSM2_SN> -serialsource <SafeNetHSM1_SN>`)



Note: If you are verifying logs on a different HSM, you must provide the **serialsource** argument, as the SafeNet HSM will not look for other SafeNet HSM log files without it.

6. Untar logs on SafeNet HSM2 (`audit log untarlogs -f audit-<SafeNetHSM1_SN>.tgz`)
7. Verify log file. (`audit log verify -f <LOG_FILENAME>.log -serialtarget <SafeNetHSM2_SN> -serialsource <SafeNetHSM1_SN>`)



Note: You cannot pass in the full path to the log file on SafeNetSA, as the command does not parse the slashes, but it will look in all the subfolders under the HSM serial number that you specified with `serialsource`.

Backup and Restore

This chapter describes how to backup and restore the contents of your HSMs. It contains the following sections:

- ["Backup and Restore in General" below](#)
- ["Backup Your HSM Contents " on the next page](#)
- ["Backup \(Clone\) Your HSM Partition" on page 39](#)

Backup and Restore in General

Backup options

You have options when you need to back up the contents of your SafeNet USB HSM.

Direct SafeNet USB HSM to SafeNet USB HSM Cloning

If you have a small number of SafeNet USB HSMs in service, the recommended backup procedure is to simply clone to another SafeNet USB HSM that sits on a shelf until you need it. If the working unit fails, you simply plug in the clone and it works immediately, with no delay, while you work with Technical Support to fix the problem or to send the defective unit back for repair.

Cloning "backup" of SafeNet USB HSMs uses the cloning feature and the `hsm clone` or `partition clone` commands. Cloning takes place from hardware to hardware (from one SafeNet USB HSM connected to your computer to another SafeNet USB HSM connected to the same computer) in secure fashion.

Backup to SafeNet Backup HSM

If you have a larger number of SafeNet USB HSMs in service, then it begins to become uneconomical to have a separate SafeNet USB HSM as backup for each in-service unit. In that situation, it makes more sense to have just a reasonable number of spare "blank" HSMs in storage, and to use SafeNet Backup HSM to backup the contents of the production units, since one SafeNet Backup HSM can have many partitions. In the event that an in-service HSM fails, you can restore its backup onto one of the blank units and have the replacement in operation.

When backing up the contents of an HSM or a partition on an HSM, the source and target HSMs must share the same cloning domain (red PED Key for PED Authenticated HSMs), if cloning is to take place. The domain is set at initialization time, and cannot be altered without initializing the HSM again. During the transfer, all data is encrypted with the domain secret. See `saw`.

HSM or Partition

Roughly equivalent backup and restore options exist for the HSM Administrator / SO space of the HSM, and for the User Partition. These are handled separately.

For the HSM level, if you clone to a second HSM you wipe that target (initialize it) and fill it with the HSM Administrator / SO objects from the source HSM. In most cases there would not be any HSM Administrator / SO-only objects - you would be cloning the structure (authentication, etc.) Similarly, if you restore to an HSM, you initialize it in the process. This means that you cannot incrementally or selectively restore HSM SO-owned objects via cloning, so you cannot keep any changes that you made in the original HSM after the backup (clone) was made.

Normally, this is not an issue, because there is usually little need to backup HSM Administrator / SO-owned objects at the HSM level. The HSM Administrator / SO level is usually only administrative on the HSM.

The more usual requirement is to backup the working contents of a User Partition, which is the level where the real work of your Client applications takes place, and the working keys and certificates and other objects are stored.

For partitions, the cloning can include all partition objects or a subset that you indicate by a (comma-separated) list of object handles.

Your cloning backup and restore operations are "lunacm partition clone", in either direction (**to** the target HSM's partition for the backup operation, or **from** it for the restore operation).

Other Info

The authentication for the Backup Tokens can match that of the HSM or Partition, or it can be different. This is a decision that should be referred to your organization's security policies. However, the HSM and the backup token must share the same domain.

Backup Your HSM Contents

You can backup the non-User (non-partition) objects on your HSM - which would be either public objects or objects owned by the HSM Administrator / SO - with the "clone" feature.

Cloning Backup to another HSM

HSM cloning securely clones HSM objects (not including objects that are contained within HSM Partitions), from the HSM to another HSM in your computer. To backup the HSM, have ready a blank HSM, or one that is acceptable to re-initialize (initializing, or re-initializing an HSM destroys any material that was on the HSM).

To backup your HSM

1. Have two HSMs connected to your computer.
2. Start the lunacm utility.
3. Login to the primary/source SafeNet HSM as HSM Administrator / SO.
4. At the lunacm prompt, type :

```
hsm clone -objects <handles> -slot <slot number> [-password <password>]
```

 (the '-password <password>' is needed only if your HSMs are Password Authenticated
 the **source HSM is the current slot** while the target HSM is the slot that you indicate in the command).
5. Secure the receiving/target HSM. Best practice for important keys and objects is to have a backup HSM in onsite secure lockup, for quick resumption of service in case of damage or loss of the primary HSM, and another backup HSM in secure off-site storage for disaster recovery.

To later restore the HSM Administrator / SO's token objects, perform a cloning operation from the backup token to the HSM that needs the objects.

See "[Backup \(Clone\) Your HSM Partition](#)" on the next page for separate handling of partition objects.

Additional Notes

Backing-up/cloning the HSM Administrator / SO-space to a target requires that the target HSM be initialized as part of the process.

If there are no HSM Administrator / SO objects to clone (a common situation), then the process halts.

Backup (Clone) Your HSM Partition

As described elsewhere, you can have a SafeNet HSM:

- with Cloning capability (direct, secure transfer from one HSM to another)

"**Partition clone**" securely clones partition objects (not including HSM Administrator / SO objects that are contained on the HSM, but not within an HSM Partition) from the HSM Partition to an encrypted file on your computer. The two HSMs must share the same domain (that is, they must have been initialized with the same red PED Key (for PED authenticated version) or the same text Domain value (Password authenticated versions)).

To backup your HSM partition

1. Start the lunacm utility.
2. Login to the partition as User.
3. At the lunacm prompt, type

```
partition clone -objects <handles> -slot <target slot> [-password <secret>] [-force]
```

 (the '-password <password>' is needed only if your HSMs are Password Authenticated
 the **source HSM is the current slot** while the target HSM is the slot that you indicate in the command).
4. Secure the receiving/target HSM. Best practice for important keys and objects is to have a backup HSM in onsite secure lockup, for quick resumption of service in case of damage or loss of the primary HSM, and another backup HSM in secure off-site storage for disaster recovery.

Capabilities and Policies

HSM capabilities describe the SafeNet USB HSM's configuration, and are displayed . They are set at manufacture according to the model you selected at time of purchase. Capabilities can only be modified by purchase and application of capability updates.

HSM policies correspond to a subset of capabilities that allow you to customize the HSM functions. Policies can be modified to provide greater security based on your specific needs. For example, you can restrict the HSM to use only FIPS-approved algorithms by setting HSM policy **12**.

Partitions inherit the capabilities and policy settings of the HSM. Partitions also have policies that can be set to customize the partition functions. Partition policies can never be modified to be less secure than the corresponding HSM capability/policy. For example, if the HSM's cloning policy is disallowed (see HSM policy **7**), partition policies **0** and **4**, which allow cloning of private or secret keys, cannot be set.

The following sections list and describe the HSM and partition capabilities and policies:

- ["HSM Capabilities and Policies" below](#)
- ["Partition Capabilities and Policies" on page 44](#)

HSM Capabilities and Policies

HSM capabilities describe the SafeNet USB HSM's configuration. They are set a manufacture according to the model you selected at time of purchase. Capabilities can only be modified by purchase and application of capability updates.

HSM policies correspond to a subset of capabilities that allow you to modify the HSM functions. Policies can be modified to provide greater security based on your specific needs. They can never be modified to be less secure than the corresponding capability.

To view the HSM capability and policy settings, .

To modify HSM policies, login as HSM SO and use the **-policy <policy#> -value <0/1>**.

See for command syntax.

To zeroize the HSM and reset the policies to their default values, use **hsm factoryreset**.

Destructiveness

In some cases, changing an HSM policy zeroizes all application partitions or the entire HSM as a security measure. These policies are listed as **destructive**.

HSM Capability and Policy Descriptions

The table below summarizes the relationships and provides a brief description of the purpose and operation of each capability and policy.

#	HSM Capability	HSM Policy	Description
0	Enable PIN-based authentication		If allowed, the HSM authenticates all users with keyboard-entered passwords.
1	Enable PED-based authentication		If allowed, the HSM authenticates users with secrets stored on physical PED keys, read by a SafeNet Luna PED. The Crypto Officer and Crypto User roles may also be configured with a secondary, keyboard-entered challenge secret.
2	Performance level		Numerical value indicates the performance level of this HSM: <ul style="list-style-type: none"> • 4: Standard performance ~1700 1024-bit RSA signature/second • 15: Maximum performance ~7000 1024-bit RSA signatures/second
4	Enable domestic mechanisms & key sizes		Always allowed. All current SafeNet Luna HSMs are capable of full-strength cryptography with no US export restrictions.
6	Enable masking	Allow masking Destructive	If enabled, the SafeNet USB HSM is capable of SIM, and this feature can be turned on or off by the HSM SO. If disabled, the SafeNet USB HSM is not capable of SIM, and there is no way for the HSM SO to change this.
7	Enable cloning	Allow cloning Destructive	If allowed, the HSM is capable of cloning cryptographic objects from one partition to another. This policy must be enabled to backup partitions over a network or create HA groups. Partition Security Officers may then enable/disable cloning on individual partitions.
8	Enable special cloning certificate		Always disallowed on current SafeNet USB HSM. No vendor-specific cloning certificates can be loaded onto the HSM.
9	Enable full (non-backup) functionality		If allowed, the HSM is capable of full cryptographic functions. This capability is only disallowed on SafeNet Luna Backup HSMs.
12	Enable non-FIPS algorithms	Allow non-FIPS algorithms Destructive	If allowed, the HSM can use all available cryptographic algorithms. If disallowed, only algorithms sanctioned by the FIPS 140-2 standard are permitted. The following is displayed in the output from :
15	Enable SO reset of partition PIN	SO can reset partition PIN	If allowed, a Partition SO can reset the password or PED secret of a Crypto Officer who has been locked out after

#	HSM Capability	HSM Policy	Description
		Destructive	too many bad login attempts. If disallowed, the lockout is permanent and the partition contents are no longer accessible. The partition must be re-initialized, and key material restored from a backup device. See "Failed Logins" on page 316 for more information.
16	Enable network replication	Allow network replication	If allowed, cryptographic object cloning is permitted over a network. This is required for HA groups, and for partition backup to a remote or client-connected SafeNet Luna Backup HSM. If disallowed, cloning over a network is not permitted. Partition backup is possible to a locally-connected SafeNet Luna Backup HSM only. Setting this policy to 0 means that only the HSM SO can backup partitions. This capability is allowed only on cloning HSM versions, and is disallowed on Key-Export HSM versions.
17	Enable Korean Algorithms	Allow Korean algorithms	If allowed, the SafeNet USB HSM can use the Korean algorithm set. This capability may be purchased as an upgrade. See "Software Maintenance and Updates" on page 350 .
18	FIPS evaluated		Always disallowed - deprecated policy. All SafeNet USB HSMs are capable of operating in FIPS Mode.
19	Manufacturing Token		N/A (SafeNet internal use only)
20	Enable Remote Authentication	Allow Remote Authentication Destructive	Deprecated policy - Remote Authentication is no longer supported. The feature is replaced by Remote PED.
21	Enable forcing user PIN change	Force user PIN change after set/reset	If allowed, when a Partition SO initializes the Crypto Officer role (or resets the password/PED secret), the CO must change the credential with role changepw before any other actions are permitted. The same is true when the CO initializes/resets the Crypto User role. This policy is intended to enforce the separation of roles on the partition. If disallowed, the CO/CU may continue to use the credential assigned by the Partition SO.
22	Enable portable masking key	Allow off-board storage Destructive	Allows or disallows the use of the portable SIM key.
23	Enable partition		Always disallowed - deprecated policy.

#	HSM Capability	HSM Policy	Description
	groups		
25	Enable Remote PED usage	Allow Remote PED usage	Always enabled on PED-authenticated SafeNet USB HSMs. All PED-authenticated HSMs are capable of connecting to a local PED or a remotely-located PED server. The HSM SO may turn this feature on or off.
26	Enable external storage of MTK split		Allows one of the splits of the MTK, the Secure Recovery Vector, to be stored outside the HSM on a purple Secure Recovery PED Key. Used for Secure Transport Mode, and for controlled/supervised recovery from tamper events. The policy associated with this capability is set automatically when the ILunaSH command hsm srk enable is run. If that command is never run, both MTK splits remain inside the HSM and recovery from tamper is automatic after restart. Not applicable to password-authenticated SafeNet USB HSMs.
27	HSM non-volatile storage space		Displays the non-volatile maximum storage space (in bytes) on the HSM. This is determined by the model of SafeNet USB HSM you selected at time of purchase.
29	Enable Acceleration	Allow Acceleration Destructive	If allowed, provides best performance for key generation (RSA, DSA, KCDSA) and HMAC operations.
30	Enable Unmasking	Allow unmasking	If allowed, cryptographic material can be migrated from legacy SafeNet appliances that used SIM.
31	Enable FW5 compatibility mode		Not applicable to SafeNet USB HSMs.
33	Maximum number of partitions		Displays the maximum number of application partitions that can be created on the HSM. The number of allowable partitions can be upgraded with a separate purchase. See " Software Maintenance and Updates " on page 350 for more information.
34	Enable ECIES support	Allow ECIES	Elliptic Curve Integrated Encryption Scheme is enabled by a purchased Capability Update. When the CUF is applied, a Policy setting becomes available to switch ECIES off and on. This is a non-FIPS algorithm. If Allow non-FIPS algorithms is set to ON , that setting overrides this one.
35	Enable Single Domain		Not applicable to SafeNet USB HSMs.
36	Enable Unified PED		Not applicable to SafeNet USB HSMs.

#	HSM Capability	HSM Policy	Description
	Key		
37	Enable MofN	Allow MofN	If allowed on PED-authenticated SafeNet USB HSMs, this policy enables you to split a PED secret among multiple PED keys (see "Using MofN" on page 243). If disallowed, users will no longer be asked to split a PED secret (M and N automatically set to 1). Always disallowed on password-authenticated HSMs.
38	Enable small form factor backup/restore		Enabled by a purchased capability update - backup the contents of an HSM partition to a SafeNet eToken 7300, by means of a SafeNet PED. Requires that Masking be enabled and allowed.
40	Enable decommission on tamper		Not applicable to SafeNet USB HSMs.
41	Enable Per-Partition SO		If allowed, the HSM SO can create PPSO partitions. Each PPSO partition has its own Security Officer with administrative control, allowing full separation of roles across the HSM.
42	Enable partition re-initialize	Allow partition re-initialize	Not applicable to SafeNet USB HSMs.

Partition Capabilities and Policies

Partitions inherit the capabilities and policy settings of the HSM. Partitions also have policies that can be set to customize the partition functions. Partition policies can never be modified to be less secure than the corresponding HSM capability/policy. For example, if the HSM's cloning policy is disallowed (see HSM policy 7), partition policies 0 and 4, which allow cloning of private or secret keys, cannot be set.



Note: If you are running more than one LunaCM session against the same partition, and change a partition policy in one LunaCM session, the policy change will be reflected in that session only. You must exit and restart the other LunaCM sessions to display the changed policy settings.

To view the partition capabilities and policy settings, use the LunaCM command **partition showpolicies**.

To modify partition policies, login as Partition SO and use the LunaCM command **partition changepolicy -policy <policy#> -value <0/1/value>**.

See ["partition changepolicy" on page 1](#) in the *LunaCM Command Reference Guide* for command syntax.

Destructiveness

In some cases, changing a partition policy forces deletion of all cryptographic objects on the partition as a security measure. These policies are listed as **destructive**. Destructive policies are typically those that change the security level of the objects stored in the partition.

Use the LunaCM command **partition showpolicies -verbose** to check whether the policy you want to enable/disable is destructive.

Partition Capabilities and Policies List

The table below summarizes the relationships and provides a brief description of the purpose and operation of each capability and policy.

#	Partition Capability	Partition Policy	Description
0	Enable private key cloning	Allow private key cloning Destructive: ON	If enabled, the partition is capable of cloning cryptographic objects to another partition. This policy must be enabled to backup partitions or create HA groups.
1	Enable private key wrapping	Allow private key wrapping Destructive: ON	Always disabled for all partitions on a SafeNet USB HSM. Private keys on the partition may not be wrapped off. The Partition SO cannot change this policy.
2	Enable private key unwrapping	Allow private key unwrapping	If enabled, private keys may be unwrapped onto the partition. The Partition SO can turn this feature on or off. If disabled, private key unwrapping is not available, and the Partition SO cannot change this.
3	Enable private key masking	Allow private key masking Destructive: ON	Always disabled. SIM has been deprecated on all current SafeNet USB HSMs. The Partition SO cannot change this policy.
4	Enable secret key cloning	Allow secret key cloning Destructive: ON	If enabled, secret keys on the partition can be backed up. The Partition SO can turn this feature on or off. The Partition SO may wish to turn this feature on immediately before a scheduled backup, and then turn it off again to prevent unauthorized backup. If disabled, secret keys cannot be backed up, and the Partition SO cannot change this.. Partition backup or partition network replication is allowed for the SafeNet high availability feature.
5	Enable secret key wrapping	Allow secret key wrapping Destructive: ON	If enabled, secret keys can be wrapped off the partition. The Partition SO can turn this feature on or off. The Partition SO may wish not to allow secret key wrapping, in which case he/she would turn off this policy. If disabled, the partition does not support secret key wrapping, and the Partition SO cannot change this.

#	Partition Capability	Partition Policy	Description
6	Enable secret key unwrapping	Allow secret key unwrapping	If enabled, secret keys can be unwrapped onto the partition. The Partition SO can turn this feature on or off. If disabled, the partition does not support secret key unwrapping, and the Partition SO cannot change this.
7	Enable secret key masking	Allow secret key masking Destructive: ON	Always disabled. SIM has been deprecated on all current SafeNet USB HSMs. The Partition SO cannot change this policy.
10	Enable multipurpose keys	Allow multipurpose keys Destructive: ON	If enabled, keys for multiple purposes, such as signing and decrypting, may be created on the partition. The Partition SO can turn this feature on or off. If disabled, keys created on (or unwrapped onto) the partition must specify only a single function in the attribute template.
11	Enable changing key attributes	Allow changing key attributes Destructive: ON	If enabled, non-sensitive attributes of the keys on the partition are modifiable (the user can change the functions that the key can use). If disabled, keys created on the partition cannot be modified. This policy affects the following "key function attributes": CKA_ENCRYPT CKA_DECRYPT CKA_WRAP CKA_UNWRAP CKA_SIGN CKA_SIGN_RECOVER CKA_VERIFY CKA_VERIFY_RECOVER CKA_DERIVE CKA_EXTRACTABLE
15	Allow failed challenge responses	Ignore failed challenge responses Destructive: ON	This policy applies to PED-authenticated SafeNet USB HSMs only. The Partition SO can turn the feature on or off. If enabled, failed challenge secret login attempts on an activated partition are not counted towards a partition lockout. Only failed PED key authentication attempts will increment the counter. If disabled, failed login attempts using either a PED key or a challenge secret will count towards a partition lockout. See "About Activation and Auto-Activation" on page 167 and "Failed Logins" on page 316 for more information.
16	Enable operation without RSA	Operate without RSA blinding	If enabled, the partition may run in a mode that does not

#	Partition Capability	Partition Policy	Description
	blinding	Destructive: ON	<p>use RSA blinding (a technique that introduces random elements into the signature process to prevent timing attacks on the RSA private key. Use of this technique may be required by certain security policies, but it does reduce performance). The Partition SO can turn this feature on or off.</p> <p>If disabled, the partition will always run in RSA blinding mode; performance will be affected.</p> <p>If the policy is on (set to 1), RSA blinding is not used.</p>
17	Enable signing with non-local keys	Allow signing with non-local keys	<p>If a key was generated on an HSM, CKA_LOCAL is set to 1. With this policy turned off, only keys with CKA_LOCAL=1 can be used to sign data on the HSM.</p> <p>Keys that are imported (unwrapped) to the HSM have CKA_LOCAL explicitly set to 0, so they may not be used for signing. Cloning and SIM maintain the value of CKA_LOCAL.</p> <p>With this policy turned on, keys that did not originate on the HSM (CKA_LOCAL=0) may be used for signing, and their trust history is not assured.</p>
18	Enable raw RSA operations	Allow raw RSA operations Destructive: ON	<p>If enabled, the partition may allow raw RSA operations (mechanism CKM_RSA_X_509). This allows weak signatures and weak encryption. The Partition SO can turn this feature on or off.</p> <p>If disabled, the partition will not support raw RSA operations.</p>
20	Max failed user logins allowed	Max failed user logins allowed	<p>Displays the maximum number of failed partition login attempts before the partition is locked out (see "Failed Logins" on page 316).</p> <p>The Partition SO can change the number of failed logins to a value lower than the maximum if desired.</p>
21	Enable high availability recovery	Allow high availability recovery	<p>If enabled, partitions in the same HA group may be used to restore the login state of this partition after power outage or other deactivation. RecoveryLogin must be configured in advance (see "role recoveryinit" on page 1 and "role recoverylogin" on page 1 in the <i>LunaCM Command Reference Guide</i> for details). The Partition SO can turn this feature on or off.</p>
22	Enable activation	Allow activation	<p>Applies only to PED-authenticated HSMs.</p> <p>If enabled, the black and/or gray PED key secrets may be cached, so that the CO or CU only needs the challenge secret to login. The Partition SO can turn this feature on or off.</p>

#	Partition Capability	Partition Policy	Description
			<p>If disabled (or the policy is turned off), PED keys must be presented at each login, whether the call is local or from a client application.</p> <p>This policy setting is overridden and activation is disabled if a tamper event occurs, or if an uncleared tamper event is detected on reboot. See "SafeNet HSM Tamper Detection" on page 1, and "About Activation and Auto-Activation" on page 167 for more information.</p>
23	Enable auto-activation	Allow auto-activation	<p>See Capability 22 above for a description of activation.</p> <p>If enabled, the black or gray PED key secrets may be encrypted and semi-permanently cached to hard disk, so that the partition's activation status can be maintained after a power loss of up to two hours. The Partition SO can turn this feature on or off.</p> <p>If disabled, this partition does not support auto-activation.</p> <p>This policy setting is overridden and auto-activation is disabled if a tamper event occurs, or if an uncleared tamper event is detected on reboot. See "SafeNet HSM Tamper Detection" on page 1, and "About Activation and Auto-Activation" on page 167 for more information.</p>
25	Minimum PIN length (inverted: 255 - min)	Minimum PIN length (inverted: 255 - min)	<p>The absolute minimum length for a partition login PIN is 8 characters. This is displayed as a value subtracted from 256. The policy value is determined as follows:</p> <p>Subtract the desired minimum PIN length from 256 (the absolute maximum length), and set policy 25 to that value.</p> <p>256 - (min PIN) = (policy value)</p> <p>For example, to set the minimum PIN length to 10 characters, the Partition SO should set the value of this policy to 246:</p> <p>256 - 10 = 246</p> <p>The reason for this inversion is that a policy can only be set to a value equal to or lower than the value set by its capability. If the absolute minimum PIN length was set to 8, the Partition SO would be able to set the preferred minimum to 2, a less-secure policy. The Partition SO may only change the minimum PIN length to increase security by forcing stronger passwords.</p>
26	Maximum PIN length	Maximum PIN length	<p>The absolute maximum length for a partition login PIN is 255 characters. The effective maximum may be changed by the Partition SO, and must always be greater than the value of the minimum PIN length, determined by the formula in the description of policy 25 (above).</p>

#	Partition Capability	Partition Policy	Description
28	Enable Key Management Functions	Allow Key Management Functions Destructive: ON	The Partition SO can disable access to any key management functions by the user - all users become Crypto Users (the restricted-capability user) even if logged in as Crypto Officer.
29	Enable RSA signing without confirmation	Perform RSA signing without confirmation Destructive: ON	The HSM can perform an internal verification (confirmation) of a signing operation to validate the signature. This confirmation is disabled by default because it has a performance impact on signature operations.
30	Enable Remote Authentication	Allow Remote Authentication	Deprecated policy - Remote Authentication is no longer supported. The feature is replaced by Remote PED.
31	Enable private key unmasking	Allow private key unmasking	Remove encryption with AES 256-bit key from private key
32	Enable secret key unmasking	Allow secret key unmasking	Remove encryption with AES 256-bit key from secret key
33	Enable RSA PKCS mechanism	Allow RSA PKCS mechanism Destructive: ON	
34	Enable CBC-PAD (un)wrap keys of any size	Allow CBC-PAD (un)wrap keys of any size Destructive: ON	
35	Enable private key SFF backup/restore	Allow private key SFF backup/restore	Not available in this release.
36	Enable secret key SFF backup/restore	Allow secret key SFF backup/restore	Not available in this release.

Configuration File Summary

Many aspects of SafeNet HSM configuration and operation are controlled or adjusted by the Chrystoki.conf file (Linux/UNIX) or Crystoki.ini file (Windows).

The configuration file is organized into named sections, under which related configuration-affecting entries might appear. A basic configuration file is always present in the SafeNet Client folder, installed by the SafeNet Client installer, with default values assigned to the populated entries. In addition to the most basic sections and entries, some additional sections and entries can be included at installation time, if you select more than the minimal installation options for your HSM model(s).

In addition, new entries can be added, or existing entries can be adjusted by actions that you perform in SafeNet tools such as LunaCM and vtl.

Finally, some sections or entries can be added or adjusted by manual editing of the Chrystoki.conf / Crystoki.ini file.

If you install SafeNet Client where a previous version was installed, then the existing configuration file is saved and the new file adds to the existing content if appropriate. That is, if you have a SafeNet HSM setup, already configured and tweaked to your satisfaction, those settings are preserved when you update to newer SafeNet Client.

The following table lists sections and settings that you are likely to encounter in normal use of SafeNet products. Not all are applicable to every SafeNet HSM. Each setting is named, with default values, allowed range of values, description of the item/setting, and remarks about any interactions between the current setting and others that you might configure.

Where the range is a file path, <luna_client_dir> specifies the path to your SafeNet HSM client installation, for example <luna_client_dir> on Windows.

Setting	Range (Default)	Description
[Chrystoki2]		
LibNT=	(<luna_client_dir>\cryptoki.dll)	Path to the Chrystoki2 library
[Luna]		
PEDTimeout1=	(100000)	Specifies the PED timeout time 1 - defines how long the HSM tries to detect if it can talk to the PED before starting the actual communication with it. If the PED is unreachable the HSM returns to the host a

Setting	Range (Default)	Description
		<p>result code for the respective HSM command. The result code indicates that the PED is not connected. This timeout is intended to be small so that the user is informed quickly that the PED is not connected.</p>
PEDTimeout2=	(200000)	<p>Specifies the PED timeout time 2 - defines how long the firmware waits for the local PED to respond to PED commands. PED commands should not be confused with PED-related HSM commands. An HSM sends PED commands to the PED when processing PED-related HSM commands, such as LOGIN or PED_CONNECT. One PED-related HSM command can involve many PED commands being sent by the HSM to the PED (for example, the MofN related commands). If a local PED does not respond to the PED commands within the span of PEDTimeout2 the HSM returns an appropriate result code (such as PED_TIMEOUT) for the respective PED-related HSM command.</p> <p>NOTE: The (default) value of 200000 is necessary to support Small Form-Factor Backup.</p>
PEDTimeout3=	(10000)	<p>Specifies the PED timeout time 3 - defines additional time the firmware must wait for the remote PED to</p>

Setting	Range (Default)	Description
		respond to PED commands. That is, the actual time the firmware waits for a remote PED to respond is PEDTimeout2 + PEDTimeout3.
DefaultTimeOut=	(500000)	Sets the default timeout interval - defines how long the HSM driver in the host system waits for HSM commands to return a result code. If the result code is not returned in that time, the driver assumes that the HSM is stuck and halts it, with the DEVICE_ERROR returned to all applications that use the HSM. Most HSM commands use this timeout. Very few exceptions exist, when a command's timeout is hard-coded in the Cryptoki library, or separate timeouts are specified in the Chrystoki.conf for certain classes of HSM commands.
CommandTimeoutPedSet=	(720000)	This is such an exception to DefaultTimeout (above). It defines timeout for all PED-related HSM commands. This class of PED-related commands can take more time than the ordinary commands that subscribe to the DefaultTimeOut value. As a rule of thumb, CommandTimeOutPedSet = DefaultTimeOut + PEDTimeout1 + PEDTimeout2 + PEDTimeout3. NOTE: The (default) value

Setting	Range (Default)	Description
		of 720000 is necessary to support Small Form-Factor Backup.
KeypairGenTimeout=	(2700000)	The amount of time the library allows for a Keypair generate operation to return a value. Due to the random component, large key sizes can take an arbitrarily long time to generate, and this setting keeps the attempts within reasonable bounds. The default is calculated as the best balance between the inconvenience of occasional very long waits and the inconvenience of restarting a keygen operation. You can change it to suit your situation.
CloningCommandTimeout=	(300000)	
[CardReader]		
RemoteCommand=	0 = false (1 = true)	This setting was used when debugging older SafeNet products. For modern products it is ignored.
LunaG5Slots=	(3)	Number of SafeNet USB HSM slots reserved so that the library will check for connected devices. Can be set to zero if you have no SafeNet USB HSMs and wish to get rid of the reserved spaces in your slot list. Can be set to any number, but is effectively limited by the number of external USB devices your host can support.

Setting	Range (Default)	Description
[RBS]		
HostName=	Any hostname or IP address (0.0.0.0)	
HostPort=	Any unassigned port (1792)	
ClientAuthFile=	(<luna_client_dir>\config\clientauth.dat)	
ServerCertFile=	(<luna_client_dir>\cert\server\server.pem)	
ServerPrivKeyFile=	(<luna_client_dir>\cert\server\serverkey.pem)	
ServerSSLConfigFile=	(<luna_client_dir>\openssl.cnf)	
CmdProcessor=	(<luna_client_dir>\rbs_processor2.dll)	
NetServer=	0 = false (1 = true)	
[LunaSA Client]		
HtlDir=	(<luna_client_dir>\htl\)	Location of HTL-related files - dhparams certificate, htl_client, and the logs directory
SSLConfigFile=	(<luna_client_dir>\openssl.cnf)	Location of the OpenSSL configuration file.
ReceiveTimeout=	in milliseconds (20000)	Number of milliseconds before a receive timeout
TCPKeepAlive=	0 = false (1 = true)	TCPKeepAlive TCPKeepAlive is a TCP stack option, available at the LunaClient, and at the SafeNet Network HSM appliance. For SafeNet purposes, it is controlled via an entry in the Chrystoki.conf /crystoki.ini file on the LunaClient, and in an equivalent file on

Setting	Range (Default)	Description
		<p>SafeNet Network HSM. For SafeNet HSM 6.1 and newer, a fresh client software installation includes an entry "TCPKeepAlive=1" in the "LunaSA Client" section of the configuration file Chrystoki.conf (Linux/UNIX) or crystoki.ini (Windows). Config files and certificates are normally preserved through an uninstall, unless you explicitly delete them.</p> <p>As such, if you update (install) LunaClient software where you previously had an older LunaClient that did not have a TCPKeepAlive entry, one is added and set to "1" (enabled), by default. In the case of update, if TCPKeepAlive is already defined in the configuration file, then your existing setting (enabled or disabled) is preserved.</p> <p>On the SafeNet Network HSM appliance, where you do not have direct access to the file system, the TCPKeepAlive= setting is controlled by the lunash:> ntls TCPKeepAlive set command.</p> <p>The settings at the appliance and the client are independent. This allows a level of assurance, in case (for example) a firewall setting blocks in one direction.</p>
NetClient=	0 = false (1 = true)	If true library will search for network slots

Setting	Range (Default)	Description
ServerCAFile=	(<luna_client_dir>\cert\server\CAFile.pem)	Location, on the client, of the server certificate file (set by vtl)
ClientCertFile=	(<luna_client_dir>\cert\client\ClientNameCert.pem)	Location of the Client certificate file that is uploaded to SafeNet Network HSM for NTLS. (set by vtl)
ClientPrivKeyFile=	(<luna_client_dir>\cert\client\ClientNameKey.pem)	Location of the Client private key file. (set by vtl)
ServerName00=192.20.17.200 ServerPort00=1792 ServerHtl00=0 ServerName01= ServerPort01= ServerHtl01=		Entries embedded by VTL utility, when you run "vtl addServer" command. Identifies the NTLS-linked SafeNet Network HSM servers, and determines the order in which they are polled to create a slot list.

[Presentation]

ShowUserSlots=<slot>(<serialnumber>)	Comma-delimited list of <slotnumber> (<serialnumber>), like ShowUserSlots=1(351970018022),2(351970018021),3(351970018020),....	Sets the starting slot for the identified partition (affects only PPSO partitions). If one PPSO slot on an HSM is specified, then any that are not listed from that HSM are not displayed.
ShowAdminTokens=	yes/(no)	Admin partitions of local HSMs are visible/(not visible) in a slot listing
ShowEmptySlots=	(0)/1	When the number of partitions on an HSM is not at the limit, unused slots are shown/(not shown).
OneBaseSlotId=	(0)/1	Causes basic slot list to start at slot number 1 instead of (0).

[HAConfiguration]

Setting	Range (Default)	Description
HAOnly=	(0)/1	When set to 1, shows only the HA virtual slot to the client, and hides the physical partitions/slots that are members of the virtual slot. Setting HAOnly helps prevent synchronization problems among member partitions, by forcing all client actions to be directed against the virtual slot, and dealing with synch transparently. HAOnly also prevents the shifting of slot numbers in the slot list that could occur if a visible physical partition were to drop out, which could disrupt an application that identifies its client partitions by slot numbers.
reconnAtt=	(10)	Specifies how many reconnection attempts will be made, when a member drops from the group. A value of "-1" is infinite retries.
AutoReconnectInterval=	(60) seconds	Specifies the interval at which the library will attempt to reconnect with a missing member, until "reconnAtt" is reached, and attempts cease. The default value of 60 seconds is the lowest that is accepted.
[Misc]		
ToolsDir=	(<luna_client_dir>\)	
PE1746Enabled=	0 = false (1 = true)	Specifies the performance target for symmetric operations based on packet sizes. For small packets, turn off this setting.

Setting	Range (Default)	Description
RSAKeyGenMechRemap=	(0)/1	<p>Controls what happens on newer firmware, when calls are made to specific older mechanisms that are now discouraged due to weakness.</p> <p>When this item is set to 0, no re-mapping is performed.</p> <p>When the value is set to 1, the following re-mapping occurs if the HSM firmware permits:</p> <ul style="list-style-type: none"> • PKCS Key Gen -> 186-3 Prime key gen • X9.31 Key Gen -> 186-3 Aux Prime key gen <p>(see "Mechanism Remap for FIPS Compliance " on page 1)</p>
RSAPre1863KeyGenMechRemap=	(0)/1	<p>Controls what happens on older firmware, when specific newer mechanisms are called, that are not supported on the older firmware.</p> <p>When this item is set to 0, no re-mapping is performed.</p> <p>When the value is set to 1, the following re-mapping occurs if the HSM firmware permits:</p> <ul style="list-style-type: none"> • 186-3 Prime key gen -> PKCS Key Gen • 186-3 Aux Prime key gen -> X9.31 Key Gen <p>Intended for evaluation purposes, such as with existing integrations that require newer mechanisms, before you update to firmware that actually supports the more secure</p>

Setting	Range (Default)	Description
		mechanisms. Be careful with this setting, which makes it appear you are getting a new, secure mechanism, when really you are getting an outdated, insecure mechanism. (see " Mechanism Remap for FIPS Compliance " on page 1)
ProtectedAuthenticationPathFlagStatus=	(0)/1/2	This flag specifies which role to check for challenge request status. Possible values include: <ul style="list-style-type: none"> • 0 (default): no challenge request • 1: check for Crypto Officer challenge request • 2: check for Crypto User challenge request Edited using the configurator tool.
[Secure Trusted Channel]		
ClientIdentitiesDir=	<luna_client_dir>\data\client_ identities	Specifies the directory used to store the STC client identity.
PartitionIdentitiesDir=	<luna_client_dir>\data\partition_ identities	Specifies the directory used to store the STC partition identities exported using the LunaCM stconfig partitionid export command.
ClientTokenLib=	For soft token: <ul style="list-style-type: none"> • <luna_client_dir>\softtoken.dll • <luna_client_dir>\win32\softtoken.dll (32-bit Windows) For hard token: <ul style="list-style-type: none"> • C:\Windows\System32\etoken.dll 	Specifies the location of the token library. This value must be correct in order to use a client token. By default, ClientTokenLib points to the location of the soft token library. If you are

Setting	Range (Default)	Description
	(Windows) <ul style="list-style-type: none"> • /usr/lib/libetoken.so (32-bit Linux/UNIX) • /usr/lib64/libetoken.so (64-bit Linux/UNIX) 	using a hard token, you must manually change this value to point to the hard token library for your operating system.
SoftTokenDir=	<luna_client_dir>\softtoken	Specifies the location where the STC client soft token (token.db) is stored. Each client soft token is stored in its own numbered subdirectory. Note: In this release there is only one client token, which is stored in the 001 subdirectory.

This chapter describes cloning domains. It contains the following sections:

- ["Single Domain Policy" below](#)
- ["Legacy Domains and Migration" on the next page](#)

Single Domain Policy

The HSM is able to support multiple partitions [See Note * below], each with its own cloning domain, if desired, as well as partition authentication for administrative users (black User PED Key for PED-authenticated HSMs, etc.) and for clients/applications (the partition challenge secret). It is possible to force all partitions on the HSM to use the same cloning domain as the administrative partition (the SO space), by setting the "Force Single Domain" policy to "Yes". This would normally be decided before any user partitions have been created on the HSM, because it is a destructive policy change, meaning that any existing HSM contents and partitions are destroyed when this policy changes. This is a convenience feature. It does not affect other authentication secrets that apply to individual partitions on the HSM.

- If the policy is set to "No" - not in force - then whenever a new partition is created, the SO is prompted to create a new cloning domain for that partition, or to imprint the partition with an existing domain. By re-using existing domain secrets, you can cause partitions to share domains, if desired, but that is optional and not forced while the policy is set to "No".
- If the policy is set to "Yes" - in force - then that prompt is skipped and each new partition is automatically assigned the cloning domain that is already in use for the HSM SO / administrative partition.
- If the policy is set to yes, then the Domain PED Key cannot have a PED PIN

Changing policies marked "destructive" will zeroize (erase completely) the entire HSM.

Description	Value	Code	Destructive
=====	=====	=====	=====
Allow masking	On	6	Yes
Allow cloning	On	7	Yes
Allow non-FIPS algorithms	On	12	Yes
SO can reset partition PIN	On	15	Yes
Allow network replication	On	16	No
Allow Remote Authentication	On	20	Yes
Allow offboard storage	On	22	Yes
Allow partition groups	On	23	No
Allow remote PED usage	On	25	No
Allow Acceleration	On	29	Yes
Allow unmasking	On	30	Yes
Allow FW5 compatibility mode	Off	31	No
Force Single Domain	On	35	Yes
Allow Unified PED Key	On	36	No

The HSM is NOT in FIPS 140-2 approved operation mode.

```
Command Result : 0 (Success)
[local_host] lush:>
```



Note: For SafeNet USB HSM and SafeNet PCIe HSM, two partitions can exist, the HSM Security Officer/administrative partition (as long as the HSM has been initialized), and a single User/Application partition (once that has been created).

For SafeNet Network HSM, up to 101 partitions can exist, the HSM Security Officer/administrative partition (as long as the HSM has been initialized), and up to 100 User/Application partitions depending on purchased-or-upgraded configuration (once those are created).

Legacy Domains and Migration

The "Legacy Cloning Domain" for Password authenticated HSM partitions is the text string that was used as a cloning domain on the legacy token HSM whose contents are to be migrated to the SafeNet Network HSM partition.

The "Legacy Cloning Domain" for PED authenticated HSM partitions is the cloning domain secret on the red PED key for the legacy PED authenticated token HSM whose contents are to be migrated to the SafeNet Network HSM partition.

Your **target** SafeNet Network HSM partition has, and retains, whatever modern partition cloning domain was imprinted (on a red PED Key) when the partition was created. The "partition setLegacyDomain" command takes the domain value from your legacy HSM's red PED Key and associates that with the modern-format domain of the partition, to allow the partition to be the cloning (restore...) recipient of objects from the legacy (token) HSM.

You can repeat the "partition setLegacyDomain" command in SafeNet Shell (lunash:>) or in Lunacm, appending a different legacy domain to the partition's own domain, allowing you to consolidate the content of multiple legacy HSMs/Tokens onto a single modern partition, if desired.

The following table illustrates what happens when objects from several legacy tokens (SafeNet CA4) are migrated to SafeNet Network HSM 5 partitions. Shown are different scenarios for the legacy domain(s) and for the SafeNet Network HSM partition domain(s).

Source Token/HSM			Target HSM Partition		
Token Name	Token Contents	Token Domain	Partition Name	Partition Contents	Partition Domain

Example = four legacy tokens (different legacy domains) to four partitions (where all partitions have different modern domains)

MyToken1	Key1a, Key1b, Cert1	LegacyDomain1	MyPartition1	Key1a, Key1b, Cert1	ModernDomain1 (with LegacyDomain1 set)
MyToken2	Key2a, Key2b, Cert2	LegacyDomain2	MyPartition2	Key2a, Key2b, Cert2	ModernDomain2 (with

Source Token/HSM			Target HSM Partition		
Token Name	Token Contents	Token Domain	Partition Name	Partition Contents	Partition Domain
					LegacyDomain2 set)
MyToken3	Key3a, Key3b, Cert3	LegacyDomain3	MyPartition3	Key3a, Key3b, Cert3	ModernDomain3 (with LegacyDomain3 set)
MyToken4	Key4a, Key4b, Cert4	LegacyDomain4	MyPartition4	Key4a, Key4b, Cert4	ModernDomain4 (with LegacyDomain4 set)

Example = four legacy tokens (different legacy domains) to four partitions (where all partitions have same modern domain)

MyToken1	Key1a, Key1b, Cert1	LegacyDomain1	MyPartition1	Key1a, Key1b, Cert1	ModernDomain1 (with LegacyDomain1 set)
MyToken2	Key2a, Key2b, Cert2	LegacyDomain2	MyPartition2	Key2a, Key2b, Cert2	ModernDomain1 (with LegacyDomain2 set)
MyToken3	Key3a, Key3b, Cert3	LegacyDomain3	MyPartition3	Key3a, Key3b, Cert3	ModernDomain1 (with LegacyDomain3 set)
MyToken4	Key4a, Key4b, Cert4	LegacyDomain4	MyPartition4	Key4a, Key4b, Cert4	ModernDomain1 (with LegacyDomain4 set)

Example = four legacy tokens (shared legacy domain) to four partitions (where all partitions have different modern domains)

MyToken1	Key1a, Key1b, Cert1	Common LegacyDomain1	MyPartition1	Key1a, Key1b, Cert1	ModernDomain1 (with LegacyDomain1 set)
MyToken2	Key2a, Key2b, Cert2		MyPartition2	Key2a, Key2b, Cert2	ModernDomain2 (with LegacyDomain1 set)
MyToken3	Key3a, Key3b, Cert3		MyPartition3	Key3a, Key3b, Cert3	ModernDomain3 (with LegacyDomain1 set)
MyToken4	Key4a, Key4b, Cert4		MyPartition4	Key4a, Key4b, Cert4	ModernDomain4 (with

Source Token/HSM			Target HSM Partition		
Token Name	Token Contents	Token Domain	Partition Name	Partition Contents	Partition Domain
					LegacyDomain1 set)

Example = four legacy tokens (shared legacy domain) to four partitions (where all partitions have same modern domain)

MyToken1	Key1a, Key1b, Cert1	Common LegacyDomain1	MyPartition1	Key1a, Key1b, Cert1	ModernDomain1 (with LegacyDomain1 set i.e., same modern domain for all 4 partitions and same legacy domain associated to all 4 partitions)
MyToken2	Key2a, Key2b, Cert2		MyPartition2	Key2a, Key2b, Cert2	
MyToken3	Key3a, Key3b, Cert3		MyPartition3	Key3a, Key3b, Cert3	
MyToken4	Key4a, Key4b, Cert4		MyPartition4	Key4a, Key4b, Cert4	

Example = four legacy tokens to one partition (legacy tokens all have same domain - run "partition setLegacyDomain" once before starting to clone the first legacy token content)

MyToken1	Key1a, Key1b, Cert1	Common LegacyDomain1	MyPartition1	Key1a, Key1b, Cert1	ModernDomain1 (with LegacyDomain1 set)
MyToken2	Key2a, Key2b, Cert2			Key2a, Key2b, Cert2	
MyToken3	Key3a, Key3b, Cert3			Key3a, Key3b, Cert3	
MyToken4	Key4a, Key4b, Cert4			Key4a, Key4b, Cert4 (i.e. contents of 4 tokens into one partition)	

Example = four legacy tokens to one partition (legacy tokens all have different domains - run "partition setLegacyDomain" once before starting to clone each and EVERY legacy token's content)

MyToken1	Key1a, Key1b, Cert1	LegacyDomain1	MyPartition1	Key1a, Key1b, Cert1	ModernDomain1 (with LegacyDomain1 set)
				Key2a, Key2b, Cert2	
				Key3a, Key3b, Cert3	
				Key4a, Key4b, Cert4 (i.e. contents of	

Source Token/HSM			Target HSM Partition		
Token Name	Token Contents	Token Domain	Partition Name	Partition Contents	Partition Domain
MyToken2	Key2a, Key2b, Cert2	LegacyDomain2		4 tokens into one partition)	ModernDomain1 (with LegacyDomain2 set)
MyToken3	Key3a, Key3b, Cert3	LegacyDomain3			ModernDomain1 (with LegacyDomain3 set)
MyToken4	Key4a, Key4b, Cert4	LegacyDomain4			ModernDomain1 (with LegacyDomain4 set)

Contact SafeNet Technical Support -- e-mail: support@safenet-inc.com or phone 800-545-6608 (+1 410-931-7520 International) for the relevant Key Migration document, which includes explicit instructions to migrate your cryptographic objects between different types of SafeNet HSM (generally from legacy models to current models of HSM).

Error Codes and Troubleshooting

This chapter lists the HSM error codes and offers troubleshooting tips for some common issues. It contains the following sections:

- "General Troubleshooting Tips" below
- "System Operational and Error Messages" on the next page
- "Keycard and Token Return Codes " on page 68
- "Library Codes" on page 83

General Troubleshooting Tips

Here are just a few quick things to check if you are experiencing problems:

- Check the output of the syslog for any information on potential problems (`syslog tail`).
- If you see an apparent 'hang' condition, connect and check the PED - it may be waiting for a PED action.
- Check if you allowed the PED to time out, or if you started a command that needed PED action while the PED was not connected. You will need to re-issue the failed command after re-inserting the token, and pay attention to the PED.
- If RSA signing seems slow, check the Capabilities and Policies to ensure that Confirmation (policy #29) is switched off - if your security policy demands that signing operations must be verified on the HSM, then expect almost a 50% performance reduction
- If you perform a Restore from Backup operation and some or all of the objects are shown with an error message like "LUNA_RET_SM_ACCESS_DOES_NOT_VALIDATE", you might have interrupted the restore operation (even a `partition showContents` command could have this effect). Re-issue the Restore command, ensuring that no other commands are run against the partition while the operation is in progress - if other persons might be using their own ssh sessions to access the appliance, it might be best to disconnect the network cable [s] and perform your restore operation from the local (serial) console.

Remote PED

If you find that Windows fails to detect SafeNet PED, especially if you have disconnected and reconnected the PED's USB cable to your computer the PED may not be receiving adequate power. SafeNet PED is powered by PED port connection only when it is connected to a SafeNet HSM. When SafeNet PED is used for Remote PED, it is connected to a computer USB port, which does not have the same electrical characteristics as the PED port on a SafeNet HSM. The PED switches on, but might not receive sufficient power to operate.

- If you are connecting locally, always connect the PED to the SafeNet HSM.
- If you are connecting to a computer for use as a Remote PED server, always connect the PED power supply in addition to the USB connection.

System Operational and Error Messages

Why do I often see extra slots that say "token not present"?

This happens for two reasons:

- PKCS#11 originated in a world of software cryptography, which only later acknowledged the existence of Hardware Security Modules, so initially it did not have the concept of physically removable crypto slots. PKCS#11 requires a static list of slots when an application starts. The cryptographic "token" can be inserted into, or removed from a slot dynamically (by a user), for the duration of the application.
- When the token is inserted, the running application must be able to detect that token. When the token is removed, the running application gets "token not present". Because we allow for the possibility of backup, and of "PKI Bundle", we routinely declare 'place-holder' slots that might later be filled by a physical SafeNet USB HSM, or a SafeNet Backup HSM, or a SafeNet DOCK2 with (potentially) two legacy token-style HSMs in its card-reader slots. As it happens, there are three (3) USB ports on a SafeNet Network HSM appliance, so we are allowing for a physical HSM connection to all of them.

In the Chrystoki.conf file (or the Windows crystoki.ini file), for SafeNet USB HSM, you can remove the empty slots by modifying the CardReader entry, like this:

```
CardReader = {
  LunaG5Slots=0;
}
```

For SafeNet Network HSM, which has its configuration file internal to the appliance, and not directly accessible for modification, you cannot change the default cryptographic slot allotments.

Error: 'hsm update firmware' failed. (10A0B : LUNA_RET_OPERATION_RESTRICTED) when attempting to perform hsm update firmware?

You must ensure that SRK is disabled before you run the firmware update. (SRK is fundamental to Secure Transport Mode and to enforced tamper-event acknowledgement in PED-authenticated SafeNet HSMs). This brings the external split of the MTK (the Secure Recovery Vector) back inside the HSM.

Also, as with any update, you should backup any important HSM contents before proceeding.

After the update is completed, you can enable SRK again. This creates a new split of the MTK to populate a new purple PED Key.

(Applies to PED-authenticated SafeNet HSMs.)

KR_ECC_POINT_INVALID Error when decrypting a file encrypted from BSAFE through ECIES using ECC key with any of the curves from the x9_t2 section.

As indicated on the BSAFE web site, they support only the NIST-approved curves (prime, Binary, and Koblitz). That includes most/all the curves from test items 0 through 37 in ckdemo, which is to say: the "secp", "X9_62_prime", and "sect" curves.

The X9.62 curves that are failing in this task are X9.62 binary/char2 curves which do not appear to be supported by BSAFE. So, you appear to be encountering a BSAFE limitation and not a SafeNet HSM problem.

Slow/interrupted response from the HSM, and the "hsm show" command shows LUNA_RET_SM_SESSION_REALLOC_ERROR

```
Appliance Details:
=====
Software Version:          4.4.0-27
Error: 'hsm show' failed. (310102 : LUNA_RET_SM_SESSION_REALLOC_ERROR)
```

```
Command Result : 65535 (Luna Shell execution)
```

The error LUNA_RET_SM_SESSION_REALLOC_ERROR means the HSM cannot expand the session table.

The HSM maintains a table for all of the open sessions. For performance reasons, the table is quite small initially. As sessions are opened (and not closed) the table fills up. When the table gets full, the HSM tries to expand the table. If there is not enough available RAM to grow the table, this error is returned.

RAM can be used up by an application that creates **and does not delete** a large number of session objects, as well as by an application that opens and **fails to close** a large number of sessions.

The obvious solution is proper housekeeping. Your applications **MUST** clean up after themselves, by closing sessions that are no longer in use - this deletes session objects associated with those sessions. If your application practice is to have long-lived sessions, and to open many objects in a given session, then your application should explicitly delete those session objects as soon as each one is no longer necessary.

By far, we see more of the former problem - abandoned sessions - and very often in conjunction with Java-based applications. Proper garbage collection includes deleting session objects when they are no longer useful, or simply closing sessions as soon as they are not required. Formally closing a session (or stopping / restarting the HSM) deletes all session objects within each affected session. These actions keep the session table small, so it uses the least possible HSM volatile memory.

Keycard and Token Return Codes

The following table summarizes HSM error codes (last updated for firmware 6.10.1) :

HSM Error	Hex Code	PKCS#11 or SFNT Defined CKR Error
LUNA_RET_OK	0x00000000	CKR_OK
LUNA_RET_CANCEL	0x00010000	CKR_CANCEL
LUNA_RET_FLAGS_INVALID	0x00040000	CKR_FLAGS_INVALID, removed from v2.0
LUNA_RET_TOKEN_NOT_PRESENT	0x00E00000	CKR_TOKEN_NOT_PRESENT
LUNA_RET_FORMER_INVALID_ENTRY_TYPE	0x00300130	CKR_DEVICE_ERROR
LUNA_RET_SP_TX_ERROR	0x00300131	CKR_DEVICE_ERROR
LUNA_RET_SP_RX_ERROR	0x00300132	CKR_DEVICE_ERROR
LUNA_RET_PED_ID_INVALID	0x00300140	CKR_DEVICE_ERROR
LUNA_RET_PED_UNSUPPORTED_PROTOCOL	0x00300141	CKR_DEVICE_ERROR

HSM Error	Hex Code	PKCS#11 or SFNT Defined CKR Error
LUNA_RET_PED_UNPLUGGED	0x00300142	CKR_PED_UNPLUGGED
LUNA_RET_PED_ERROR	0x00300144	CKR_DEVICE_ERROR
LUNA_RET_PED_UNSUPPORTED_CRYPTO_PROTOCOL	0x00300145	CKR_DEVICE_ERROR
LUNA_RET_PED_DEK_INVALID	0x00300146	CKR_DEVICE_ERROR
LUNA_RET_PED_CLIENT_NOT_RUNNING	0x00300147	CKR_PED_CLIENT_NOT_RUNNING
LUNA_RET_CL_ALIGNMENT_ERROR	0x00300200	CKR_DEVICE_ERROR
LUNA_RET_CL_QUEUE_LOCATION_ERROR	0x00300201	CKR_DEVICE_ERROR
LUNA_RET_CL_QUEUE_OVERLAP_ERROR	0x00300202	CKR_DEVICE_ERROR
LUNA_RET_CL_TRANSMISSION_ERROR	0x00300203	CKR_DEVICE_ERROR
LUNA_RET_CL_NO_TRANSMISSION	0x00300204	CKR_DEVICE_ERROR
LUNA_RET_CL_COMMAND_MALFORMED	0x00300205	CKR_DEVICE_ERROR
LUNA_RET_CL_MAILBOXES_NOT_AVAILABLE	0x00300206	CKR_DEVICE_ERROR
LUNA_RET_MM_NOT_ENOUGH_MEMORY	0x00310000	CKR_DEVICE_ERROR
LUNA_RET_MM_INVALID_HANDLE	0x00310001	CKR_DEVICE_ERROR
LUNA_RET_MM_USAGE_ALREADY_SET	0x00310002	CKR_DEVICE_ERROR
LUNA_RET_MM_ACCESS_OUTSIDE_ALLOCATION_RANGE	0x00310003	CKR_DEVICE_ERROR
LUNA_RET_MM_INVALID_USAGE	0x00310004	CKR_DEVICE_ERROR
LUNA_RET_MM_ITERATOR_PAST_END	0x00310005	CKR_DEVICE_ERROR
LUNA_RET_MM_FATAL_ERROR	0x00310006	CKR_DEVICE_ERROR
LUNA_RET_TEMPLATE_INCOMPLETE	0x00D00000	CKR_TEMPLATE_INCOMPLETE
LUNA_RET_TEMPLATE_INCONSISTENT	0x00D10000	CKR_TEMPLATE_INCONSISTENT *
LUNA_RET_ATTRIBUTE_TYPE_INVALID	0x00120000	CKR_ATTRIBUTE_TYPE_INVALID
LUNA_RET_ATTRIBUTE_VALUE_INVALID	0x00130000	CKR_ATTRIBUTE_VALUE_INVALID
LUNA_RET_ATTRIBUTE_READ_ONLY	0x00100000	CKR_ATTRIBUTE_READ_ONLY
LUNA_RET_ATTRIBUTE_SENSITIVE	0x00110000	CKR_ATTRIBUTE_SENSITIVE
LUNA_RET_OBJECT_HANDLE_INVALID	0x00820000	CKR_OBJECT_HANDLE_INVALID

HSM Error	Hex Code	PKCS#11 or SFNT Defined CKR Error
LUNA_RET_MAX_OBJECT_COUNT	0x00820001	CKR_MAX_OBJECT_COUNT_EXCEEDED
LUNA_RET_ATTRIBUTE_NOT_FOUND	0x00120010	CKR_ATTRIBUTE_TYPE_INVALID
LUNA_RET_CAN_NOT_CREATE_SECRET_KEY	0x00D10011	CKR_TEMPLATE_INCONSISTENT
LUNA_RET_CAN_NOT_CREATE_PRIVATE_KEY	0x00D10012	CKR_TEMPLATE_INCONSISTENT
LUNA_RET_SECRET_KEY_MUST_BE_SENSITIVE	0x00130013	CKR_ATTRIBUTE_VALUE_INVALID
LUNA_RET_SECRET_KEY_MUST_HAVE_SENSITIVE_ATTRIBUTE	0x00D00014	CKR_TEMPLATE_INCOMPLETE
LUNA_RET_PRIVATE_KEY_MUST_BE_SENSITIVE	0x00130015	CKR_ATTRIBUTE_VALUE_INVALID
LUNA_RET_PRIVATE_KEY_MUST_HAVE_SENSITIVE_ATTRIBUTE	0x00D00016	CKR_TEMPLATE_INCOMPLETE
LUNA_RET_SIGNING_KEY_MUST_BE_LOCAL	0x00680001	CKR_KEY_FUNCTION_NOT_PERMITTED
LUNA_RET_MULTI_FUNCTION_KEYS_NOT_ALLOWED	0x00D10018	CKR_TEMPLATE_INCONSISTENT
LUNA_RET_CAN_NOT_CHANGE_KEY_FUNCTION	0x00100019	CKR_ATTRIBUTE_READ_ONLY
LUNA_RET_KEY_SIZE_RANGE	0x00620000	CKR_KEY_SIZE_RANGE
LUNA_RET_KEY_TYPE_INCONSISTENT	0x00630000	CKR_KEY_TYPE_INCONSISTENT
LUNA_RET_KEY_INVALID_FOR_OPERATION	0x00630001	CKR_KEY_TYPE_INCONSISTENT
LUNA_RET_KEY_PARITY	0x00630002	CKR_KEY_TYPE_INCONSISTENT
LUNA_RET_KEY_UNEXTRACTABLE	0x006a0000	CKR_KEY_UNEXTRACTABLE
LUNA_RET_KEY_EXTRACTABLE	0x006a0001	KR_KEY_UNEXTRACTABLE
LUNA_RET_KEY_INDIGESTIBLE	0x00670000	CKR_KEY_INDIGESTIBLE
LUNA_RET_KEY_NOT_WRAPPABLE	0x00690000	CKR_KEY_NOT_WRAPPABLE
LUNA_RET_KEY_NOT_UNWRAPPABLE	0x00690001	CKR_KEY_NOT_WRAPPABLE
LUNA_RET_ARGUMENTS_BAD	0x00070000	CKR_ARGUMENTS_BAD

HSM Error	Hex Code	PKCS#11 or SFNT Defined CKR Error
LUNA_RET_INVALID_ENTRY_TYPE	0x00070001	CKR_INVALID_ENTRY_TYPE
LUNA_RET_DATA_INVALID	0x00200000	CKR_DATA_INVALID
LUNA_RET_SM_DATA_INVALID	0x00200002	CKR_DATA_INVALID
LUNA_RET_NO_RNG_SEED	0x00200015	CKR_DATA_INVALID
LUNA_RET_FUNCTION_NOT_SUPPORTED	0x00540000	CKR_FUNCTION_NOT_SUPPORTED
LUNA_RET_NO_OFFBOARD_STORAGE	0x00540001	CKR_FUNCTION_NOT_SUPPORTED
LUNA_RET_CL_COMMAND_NON_BACKUP	0x00540002	CKR_FUNCTION_NOT_SUPPORTED
LUNA_RET_BUFFER_TOO_SMALL	0x01500000	CKR_BUFFER_TOO_SMALL
LUNA_RET_DATA_LEN_RANGE	0x00210000	CKR_DATA_LEN_RANGE
LUNA_RET_GENERAL_ERROR	0x00050000	CKR_GENERAL_ERROR
LUNA_RET_DEVICE_ERROR	0x00300000	CKR_DEVICE_ERROR
LUNA_RET_UNKNOWN_COMMAND	0x00300001	CKR_FUNCTION_NOT_SUPPORTED
LUNA_RET_TOKEN_LOCKED_OUT	0x00300002	CKR_PIN_LOCKED
LUNA_RET_RNG_ERROR	0x00300003	CKR_DEVICE_ERROR
LUNA_RET_DES_SELF_TEST_FAILURE	0x00300004	CKR_DEVICE_ERROR
LUNA_RET_CAST_SELF_TEST_FAILURE	0x00300005	CKR_DEVICE_ERROR
LUNA_RET_CAST3_SELF_TEST_FAILURE	0x00300006	CKR_DEVICE_ERROR
LUNA_RET_CAST5_SELF_TEST_FAILURE	0x00300007	CKR_DEVICE_ERROR
LUNA_RET_MD2_SELF_TEST_FAILURE	0x00300008	CKR_DEVICE_ERROR
LUNA_RET_MD5_SELF_TEST_FAILURE	0x00300009	CKR_DEVICE_ERROR
LUNA_RET_SHA_SELF_TEST_FAILURE	0x0030000a	CKR_DEVICE_ERROR
LUNA_RET_RSA_SELF_TEST_FAILURE	0x0030000b	CKR_DEVICE_ERROR
LUNA_RET_RC2_SELF_TEST_FAILURE	0x0030000c	CKR_DEVICE_ERROR
LUNA_RET_RC4_SELF_TEST_FAILURE	0x0030000d	CKR_DEVICE_ERROR
LUNA_RET_RC5_SELF_TEST_FAILURE	0x0030000e	CKR_DEVICE_ERROR
LUNA_RET_SO_LOGIN_FAILURE_THRESHOLD	0x0030000f	CKR_SO_LOGIN_FAILURE_THRESHOLD
LUNA_RET_RNG_SELF_TEST_FAILURE	0x00300010	CKR_DEVICE_ERROR

HSM Error	Hex Code	PKCS#11 or SFNT Defined CKR Error
LUNA_RET_SM_UNKNOWN_COMMAND	0x00300011	CKR_DEVICE_ERROR
LUNA_RET_UM_TSN_MISSING	0x00300012	CKR_DEVICE_ERROR
LUNA_RET_SM_TSV_MISSING	0x00300013	CKR_DEVICE_ERROR
LUNA_RET_SM_UNKNOWN_TOSM_STATE	0x00300014	CKR_DEVICE_ERROR
LUNA_RET_DSA_PARAM_GEN_FAILURE	0x00300015	CKR_DEVICE_ERROR
LUNA_RET_DSA_SELF_TEST_FAILURE	0x00300016	CKR_DEVICE_ERROR
LUNA_RET_SEED_SELF_TEST_FAILURE	0x00300017	CKR_DEVICE_ERROR
LUNA_RET_AES_SELF_TEST_FAILURE	0x00300018	CKR_DEVICE_ERROR
LUNA_RET_FUNCTION_NOT_SUPPORTED_BY_HARDWARE	0x00300019	CKR_DEVICE_ERROR
LUNA_RET_HAS160_SELF_TEST_FAILURE	0x0030001a	CKR_DEVICE_ERROR
LUNA_RET_KCDSA_PARAM_GEN_FAILURE	0x0030001b	CKR_DEVICE_ERROR
LUNA_RET_KCDSA_SELF_TEST_FAILURE	0x0030001c	CKR_DEVICE_ERROR
LUNA_RET_HSM_INTERNAL_BUFFER_TOO_SMALL	0x0030001d	CKR_DEVICE_ERROR
LUNA_RET_COUNTER_WRAPAROUND	0x0030001e	CKR_DEVICE_ERROR
LUNA_RET_TIMEOUT	0x0030001f	CKR_TIMEOUT
LUNA_RET_NOT_READY	0x00300020	CKR_DEVICE_ERROR
LUNA_RET_RETRY	0x00300021	CKR_DEVICE_ERROR
LUNA_RET_SHA1_RSA_SELF_TEST_FAILURE	0x00300022	CKR_DEVICE_ERROR
LUNA_RET_SELF_TEST_FAILURE	0x00300023	CKR_DEVICE_ERROR
LUNA_RET_INCOMPATIBLE	0x00300024	CKR_DEVICE_ERROR
LUNA_RET_RIPEMD160_SELF_TEST_FAILURE	0x00300034	CKR_DEVICE_ERROR
LUNA_RET_TOKEN_LOCKED_OUT_CL	0x00300100	CKR_DEVICE_ERROR
LUNA_RET_TOKEN_LOCKED_OUT_MM	0x00300101	CKR_DEVICE_ERROR
LUNA_RET_TOKEN_LOCKED_OUT_UM	0x00300102	CKR_DEVICE_ERROR
LUNA_RET_TOKEN_LOCKED_OUT_SM	0x00300103	CKR_DEVICE_ERROR

HSM Error	Hex Code	PKCS#11 or SFNT Defined CKR Error
LUNA_RET_TOKEN_LOCKED_OUT_RN	0x00300104	CKR_DEVICE_ERROR
LUNA_RET_TOKEN_LOCKED_OUT_CA	0x00300105	CKR_DEVICE_ERROR
LUNA_RET_TOKEN_LOCKED_OUT_PM	0x00300106	CKR_DEVICE_ERROR
LUNA_RET_TOKEN_LOCKED_OUT_OH	0x00300107	CKR_DEVICE_ERROR
LUNA_RET_TOKEN_LOCKED_OUT_CCM	0x00300108	CKR_DEVICE_ERROR
LUNA_RET_TOKEN_LOCKED_OUT_SHA_DIGEST	0x00300109	CKR_DEVICE_ERROR
LUNA_RET_SM_ACCESS_REALLOC_ERROR	0x00310101	CKR_DEVICE_ERROR
LUNA_RET_SM_SESSION_REALLOC_ERROR	0x00310102	CKR_DEVICE_ERROR
LUNA_RET_SM_MEMORY_ALLOCATION_ERROR	0x00310103	CKR_DEVICE_ERROR
LUNA_RET_ENCRYPTED_DATA_INVALID	0x00400000	CKR_ENCRYPTED_DATA_INVALID
LUNA_RET_ENCRYPTED_DATA_LEN_RANGE	0x00410000	CKR_ENCRYPTED_DATA_LEN_RANGE
LUNA_RET_FUNCTION_CANCELED	0x00500000	CKR_FUNCTION_CANCELED
LUNA_RET_KEY_HANDLE_INVALID	0x00600000	CKR_KEY_HANDLE_INVALID
LUNA_RET_MECHANISM_INVALID	0x00700000	CKR_MECHANISM_INVALID
LUNA_RET_MECHANISM_PARAM_INVALID	0x00710000	CKR_MECHANISM_PARAM_INVALID
LUNA_RET_OPERATION_ACTIVE	0x00900000	CKR_OPERATION_ACTIVE
LUNA_RET_OPERATION_NOT_INITIALIZED	0x00910000	CKR_OPERATION_NOT_INITIALIZED
LUNA_RET_UM_PIN_INCORRECT	0x00a00000	CKR_PIN_INCORRECT
LUNA_RET_UM_PIN_INCORRECT_CONTAINER_ZEROIZED	0x00a00001	CKR_PIN_INCORRECT
LUNA_RET_UM_PIN_INCORRECT_CONTAINER_LOCKED	0x00a00002	CKR_PIN_INCORRECT
LUNA_RET_UM_PIN_LEN_RANGE	0x00a20000	CKR_PIN_LEN_RANGE
LUNA_RET_SM_PIN_EXPIRED	0x00a30000	CKR_PIN_EXPIRED
LUNA_RET_SM_EXCLUSIVE_SESSION_EXISTS	0x00b20000	CKR_SESSION_EXCLUSIVE_EXISTS
LUNA_RET_SM_SESSION_HANDLE_INVALID	0x00b30000	CKR_SESSION_HANDLE_INVALID

HSM Error	Hex Code	PKCS#11 or SFNT Defined CKR Error
LUNA_RET_SIGNATURE_INVALID	0x00c00000	CKR_SIGNATURE_INVALID
LUNA_RET_SIGNATURE_LEN_RANGE	0x00c10000	CKR_SIGNATURE_LEN_RANGE
LUNA_RET_UNWRAPPING_KEY_HANDLE_INVALID	0x00f00000	CKR_UNWRAPPING_KEY_HANDLE_INVALID
LUNA_RET_UNWRAPPING_KEY_SIZE_RANGE	0x00f10000	CKR_UNWRAPPING_KEY_SIZE_RANGE
LUNA_RET_UNWRAPPING_KEY_TYPE_INCONSISTENT	0x00f20000	CKR_UNWRAPPING_KEY_TYPE_INCONSISTENT
LUNA_RET_USER_ALREADY_LOGGED_IN	0x01000000	CKR_USER_ALREADY_LOGGED_IN
LUNA_RET_SM_OTHER_USER_LOGGED_IN	0x01000001	CKR_USER_ALREADY_LOGGED_IN
LUNA_RET_USER_NOT_LOGGED_IN	0x01010000	CKR_USER_NOT_LOGGED_IN
LUNA_RET_SM_NOT_LOGGED_IN	0x01010001	CKR_USER_NOT_LOGGED_IN
LUNA_RET_USER_PIN_NOT_INITIALIZED	0x01020000	CKR_USER_PIN_NOT_INITIALIZED
LUNA_RET_USER_TYPE_INVALID	0x01030000	CKR_USER_TYPE_INVALID
LUNA_RET_WRAPPED_KEY_INVALID	0x01100000	CKR_WRAPPED_KEY_INVALID
LUNA_RET_WRAPPED_KEY_LEN_RANGE	0x01120000	CKR_WRAPPED_KEY_LEN_RANGE
LUNA_RET_WRAPPING_KEY_HANDLE_INVALID	0x01130000	CKR_WRAPPING_KEY_HANDLE_INVALID
LUNA_RET_WRAPPING_KEY_SIZE_RANGE	0x01140000	CKR_WRAPPING_KEY_SIZE_RANGE
LUNA_RET_WRAPPING_KEY_TYPE_INCONSISTENT	0x01150000	CKR_WRAPPING_KEY_TYPE_INCONSISTENT
LUNA_RET_CERT_VERSION_NOT_SUPPORTED	0x00300300	CKR_DEVICE_ERROR
LUNA_RET_SIM_AUTHFORM_INVALID	0x0020011e	CKR_SIM_AUTHFORM_INVALID
LUNA_RET_CCM_TOO_LARGE	0x00210001	CKR_DATA_LEN_RANGE
LUNA_RET_TEST_VS_BSAFE_FAILED	0x00300820	CKR_DEVICE_ERROR
LUNA_RET_SFNT3120_ERROR	0x00300821	CKR_DEVICE_ERROR
LUNA_RET_SFNT3120_SELFTEST_FAILED	0x00300822	CKR_DEVICE_ERROR
LUNA_RET_SFNT3120_CRC	0x00300823	CKR_DEVICE_ERROR
LUNA_RET_SFNT3120_ALG_NO_SOFTWARE_SUPPORT	0x00300824	CKR_DEVICE_ERROR

HSM Error	Hex Code	PKCS#11 or SFNT Defined CKR Error
LUNA_RET_ISES_ERROR	0x00300880	CKR_DEVICE_ERROR
LUNA_RET_ISES_INIT_FAILED	0x00300881	CKR_DEVICE_ERROR
LUNA_RET_ISES_LNAU_TEST_FAILED	0x00300882	CKR_DEVICE_ERROR
LUNA_RET_ISES_RNG_TEST_FAILED	0x00300883	CKR_DEVICE_ERROR
LUNA_RET_ISES_CMD_FAILED	0x00300884	CKR_DEVICE_ERROR
LUNA_RET_ISES_CMD_PARAMETER_INVALID	0x00300885	CKR_DEVICE_ERROR
LUNA_RET_ISES_TEST_VS_BSAFE_FAILED	0x00300886	CKR_DEVICE_ERROR
LUNA_RET_PE1746_ERROR	0x00300887	CKR_DEVICE_ERROR
LUNA_RET_RM_ELEMENT_VALUE_INVALID	0x00200a00	CKR_DATA_INVALID
LUNA_RET_RM_ELEMENT_ID_INVALID	0x00200a01	CKR_DATA_INVALID
LUNA_RET_RM_NO_MEMORY	0x00310a02	CKR_DEVICE_MEMORY
LUNA_RET_RM_BAD_HSM_PARAMS	0x00300a03	CKR_DEVICE_ERROR
LUNA_RET_RM_POLICY_ELEMENT_DESTRUCTIVE	0x00200a04	CKR_DATA_INVALID
LUNA_RET_RM_POLICY_ELEMENT_NOT_DESTRUCTIVE	0x00200a05	CKR_DATA_INVALID
LUNA_RET_RM_CONFIG_CHANGE_ILLEGAL	0x00010a06	CKR_CANCEL
LUNA_RET_RM_CONFIG_CHANGE_FAILS_DEPENDENCIES	0x00010a07	CKR_CANCEL
LUNA_RET_LICENSE_ID_UNKNOWN	0x00200a08	CKR_DATA_INVALID
LUNA_RET_LICENSE_CAPACITY_EXCEEDED	0x00010a09	CKR_LICENSE_CAPACITY_EXCEEDED
LUNA_RET_RM_POLICY_WRITE_RESTRICTED	0x00010a0a	CKR_CANCEL
LUNA_RET_OPERATION_RESTRICTED	0x00010a0b	CKR_OPERATION_NOT_ALLOWED
LUNA_RET_CANNOT_PERFORM_OPERATION_TWICE	0x00010a0c	CKR_CANCEL
LUNA_RET_BAD_PPID	0x00200a0d	CKR_DATA_INVALID
LUNA_RET_BAD_FW_VERSION	0x00200a0e	CKR_DATA_INVALID
LUNA_RET_OPERATION_SHOULD_BE_	0x00200a0f	CKR_DATA_INVALID

HSM Error	Hex Code	PKCS#11 or SFNT Defined CKR Error
DESTRUCTIVE		
LUNA_RET_RM_CONFIG_ILLEGAL	0x00200a10	CKR_DATA_INVALID
LUNA_RET_BAD_SN	0x00200a11	CKR_DATA_INVALID
LUNA_RET_CHALLENGE_TYPE_INVALID	0x00200b00	CKR_DATA_INVALID
LUNA_RET_CHALLENGE_REQUIRES_PED	0x00010b01	CKR_CANCEL
LUNA_RET_CHALLENGE_NOT_REQUIRED	0x00010b02	CKR_CANCEL
LUNA_RET_CHALLENGE_RESPONSE_INCORRECT	0x00a00b03	CKR_PIN_INCORRECT
LUNA_RET_OH_OBJECT_VERSION_INVALID	0x00300c00	CKR_DEVICE_ERROR
LUNA_RET_OH_OBJECT_TYPE_INVALID	0x00300c01	CKR_DEVICE_ERROR
LUNA_RET_OH_OBJECT_ALREADY_EXISTS	0x00010c02	CKR_CANCEL
LUNA_RET_OH_OBJECT_OWNER_DOES_NOT_EXIST	0x00200c03	CKR_DATA_INVALID
LUNA_RET_STORAGE_TYPE_INCONSISTENT	0x00200c04	CKR_DATA_INVALID
LUNA_RET_CONTAINER_CAN_NOT_HAVE_MEMBERS	0x00200c05	CKR_DATA_INVALID
LUNA_RET_SAVED_STATE_INVALID	0x01600000	CKR_SAVED_STATE_INVALID
LUNA_RET_STATE_UNSAVEABLE	0x01800000	CKR_STATE_UNSAVEABLE
LUNA_RET_ERROR	0x80000000	CKR_GENERAL_ERROR
LUNA_RET_CONTAINER_HANDLE_INVALID	0x80000001	CKR_CONTAINER_HANDLE_INVALID
LUNA_RET_INVALID_PADDING_TYPE	0x80000002	CKR_DATA_INVALID
LUNA_RET_NOT_FOUND	0x80000007	CKR_FUNCTION_FAILED
LUNA_RET_TOO_MANY_CONTAINERS	0x80000008	CKR_TOO_MANY_CONTAINERS
LUNA_RET_CONTAINER_LOCKED	0x80000009	CKR_PIN_LOCKED
LUNA_RET_CONTAINER_IS_DISABLED	0x8000000a	CKR_PARTITION_DISABLED
LUNA_RET_SECURITY_PARAMETER_MISSING	0x8000000b	CKR_SECURITY_PARAMETER_MISSING
LUNA_RET_DEVICE_TIMEOUT	0x8000000c	CKR_DEVICE_TIMEOUT
LUNA_RET_OBJECT_DELETED	0x8000000d	HSM Internal ONLY

HSM Error	Hex Code	PKCS#11 or SFNT Defined CKR Error
LUNA_RET_INVALID_FUF_TARGET	0x8000000e	CKR_INVALID_FUF_TARGET
LUNA_RET_INVALID_FUF_HEADER	0x8000000f	CKR_INVALID_FUF_HEADER
LUNA_RET_INVALID_FUF_VERSION	0x80000010	CKR_INVALID_FUF_VERSION
LUNA_RET_KCV_PARAMETER_ALREADY_EXISTS	0x80000100	CKR_CLONING_PARAMETER_ALREADY_EXISTS
LUNA_RET_KCV_PARAMETER_COULD_NOT_BE_ADDED	0x80000101	CKR_DEVICE_MEMORY
LUNA_RET_INVALID_CERTIFICATE_DATA	0x80000102	CKR_CERTIFICATE_DATA_INVALID
LUNA_RET_INVALID_CERTIFICATE_TYPE	0x80000103	CKR_CERTIFICATE_DATA_INVALID
LUNA_RET_INVALID_CERTIFICATE_VERSION	0x80000104	CKR_CERTIFICATE_DATA_INVALID
LUNA_RET_INVALID_MODULUS_SIZE	0x80000105	CKR_ATTRIBUTE_VALUE_INVALID
LUNA_RET_WRAPPING_ERROR	0x80000107	CKR_WRAPPING_ERROR
LUNA_RET_UNWRAPPING_ERROR	0x80000108	CKR_UNWRAPPING_ERROR
LUNA_RET_INVALID_PRIVATE_KEY_TYPE	0x80000109	CKR_DATA_INVALID
LUNA_RET_TSN_MISMATCH	0x8000010a	CKR_DATA_INVALID
LUNA_RET_KCV_PARAMETER_MISSING	0x8000010b	CKR_CLONING_PARAMETER_MISSING
LUNA_RET_TWC_PARAMETER_MISSING	0x8000010c	CKR_CERTIFICATE_DATA_MISSING
LUNA_RET_TUK_PARAMETER_MISSING	0x8000010d	CKR_CERTIFICATE_DATA_MISSING
LUNA_RET_CPK_PARAMETER_MISSING	0x8000010e	CKR_KEY_NEEDED
LUNA_RET_MASKING_NOT_SUPPORTED	0x8000010f	CKR_FUNCTION_NOT_SUPPORTED
LUNA_RET_INVALID_ACCESS_LEVEL	0x80000110	CKR_ARGUMENTS_BAD
LUNA_RET_MAC_MISSING	0x80000111	CKR_MAC_MISSING
LUNA_RET_DAC_POLICY_PID_MISMATCH	0x80000112	CKR_DAC_POLICY_PID_MISMATCH
LUNA_RET_DAC_MISSING	0x80000113	CKR_DAC_MISSING
LUNA_RET_BAD_DAC	0x80000114	CKR_BAD_DAC
LUNA_RET_SSK_MISSING	0x80000115	CKR_SSK_MISSING
LUNA_RET_BAD_MAC	0x80000116	CKR_BAD_MAC
LUNA_RET_DAK_MISSING	0x80000117	CKR_DAK_MISSING

HSM Error	Hex Code	PKCS#11 or SFNT Defined CKR Error
LUNA_RET_BAD_DAK	0x80000118	CKR_BAD_DAK
LUNA_RET_HOK_MISSING	0x80000119	CKR_CERTIFICATE_DATA_MISSING
LUNA_RET_CITS_DAK_MISSING	0x8000011a	CKR_CITS_DAK_MISSING
LUNA_RET_SIM_AUTHORIZATION_FAILED	0x8000011b	CKR_SIM_AUTHORIZATION_FAILED
LUNA_RET_SIM_VERSION_UNSUPPORTED	0x8000011c	CKR_SIM_VERSION_UNSUPPORTED
LUNA_RET_SIM_CORRUPT_DATA	0x8000011d	CKR_SIM_CORRUPT_DATA
LUNA_RET_ECC_MIC_MISSING	0x8000011e	CKR_CERTIFICATE_DATA_MISSING
LUNA_RET_ECC_HOK_MISSING	0x8000011f	CKR_CERTIFICATE_DATA_MISSING
LUNA_RET_ECC_HOC_MISSING	0x80000120	CKR_CERTIFICATE_DATA_MISSING
LUNA_RET_ECC_DAK_MISSING	0x80000121	CKR_CERTIFICATE_DATA_MISSING
LUNA_RET_ECC_DAC_MISSING	0x80000122	CKR_CERTIFICATE_DATA_MISSING
LUNA_RET_ROOT_CERT_MISSING	0x80000123	CKR_CERTIFICATE_DATA_MISSING
LUNA_RET_HOC_MISSING	0x80000124	CKR_CERTIFICATE_DATA_MISSING
LUNA_RET_INVALID_CERTIFICATE_FUNCTION	0x80000125	CKR_CERTIFICATE_DATA_INVALID
LUNA_RET_N_TOO_LARGE	0x80000200	CKR_ARGUMENTS_BAD
LUNA_RET_N_TOO_SMALL	0x80000201	CKR_ARGUMENTS_BAD
LUNA_RET_M_TOO_LARGE	0x80000202	CKR_ARGUMENTS_BAD
LUNA_RET_M_TOO_SMALL	0x80000203	CKR_ARGUMENTS_BAD
LUNA_RET_WEIGHT_TOO_LARGE	0x80000204	CKR_ARGUMENTS_BAD
LUNA_RET_WEIGHT_TOO_SMALL	0x80000205	CKR_ARGUMENTS_BAD
LUNA_RET_TOTAL_WEIGHT_INVALID	0x80000206	CKR_ARGUMENTS_BAD
LUNA_RET_MISSING_SPLITS	0x80000207	CKR_ARGUMENTS_BAD
LUNA_RET_SPLIT_DATA_INVALID	0x80000208	CKR_ARGUMENTS_BAD
LUNA_RET_SPLIT_ID_INVALID	0x80000209	CKR_ARGUMENTS_BAD
LUNA_RET_M_OF_N_PARAMETER_NOT_AVAILABLE	0x8000020a	CKR_OPERATION_NOT_INITIALIZED
LUNA_RET_M_OF_N_ACTIVATION_REQUIRED	0x8000020b	CKR_OPERATION_NOT_INITIALIZED

HSM Error	Hex Code	PKCS#11 or SFNT Defined CKR Error
LUNA_RET_TOO_MANY_WEIGHTS	0x8000020e	CKR_ARGUMENTS_BAD
LUNA_RET_MISSING_WEIGHT_VALUE	0x8000020f	CKR_ARGUMENTS_BAD
LUNA_RET_MISSING_VALUE_FOR_M	0x80000210	CKR_ARGUMENTS_BAD
LUNA_RET_MISSING_VALUE_FOR_N	0x80000211	CKR_ARGUMENTS_BAD
LUNA_RET_MISSING_NUMBER_OF_VECTORS	0x80000212	CKR_ARGUMENTS_BAD
LUNA_RET_MISSING_VECTOR	0x80000213	CKR_ARGUMENTS_BAD
LUNA_RET_VECTOR_TOO_LARGE	0x80000214	CKR_ARGUMENTS_BAD
LUNA_RET_VECTOR_TOO_SMALL	0x80000215	CKR_ARGUMENTS_BAD
LUNA_RET_TOO_MANY_VECTORS_PROVIDED	0x80000216	CKR_ARGUMENTS_BAD
LUNA_RET_INVALID_VECTOR_SIZE	0x80000217	CKR_ARGUMENTS_BAD
LUNA_RET_M_OF_N_PARAMETER_EXIST	0x80000218	CKR_FUNCTION_FAILED
LUNA_RET_VECTOR_VERSION_INVALID	0x80000219	CKR_DATA_INVALID
LUNA_RET_VECTOR_OF_DIFFERENT_SET	0x8000021a	CKR_ARGUMENTS_BAD
LUNA_RET_VECTOR_DUPLICATE	0x8000021b	CKR_ARGUMENTS_BAD
LUNA_RET_VECTOR_TYPE_INVALID	0x8000021c	CKR_ARGUMENTS_BAD
LUNA_RET_MISSING_COMMAND_PARAMETER	0x8000021d	CKR_ARGUMENTS_BAD
LUNA_RET_M_OF_N_CLONING_IS_NOT_ALLOWED	0x8000021e	CKR_FUNCTION_NOT_SUPPORTED
LUNA_RET_M_OF_N_IS_NOT_REQUIRED	0x8000021f	CKR_OPERATION_NOT_INITIALIZED
LUNA_RET_M_OF_N_IS_NOT_INITIALIZED	0x80000220	CKR_OPERATION_NOT_INITIALIZED
LUNA_RET_M_OF_N_SECRET_INVALID	0x80000221	CKR_GENERAL_ERROR
LUNA_RET_CCM_NOT_PRESENT	0x80000300	CKR_FUNCTION_NOT_SUPPORTED
LUNA_RET_CCM_NOT_SUPPORTED	0x80000301	CKR_FUNCTION_NOT_SUPPORTED
LUNA_RET_CCM_UNREMOVABLE	0x80000302	CKR_DATA_INVALID
LUNA_RET_CCM_CERT_INVALID	0x80000303	CKR_DATA_INVALID
LUNA_RET_CCM_SIGN_INVALID	0x80000304	CKR_DATA_INVALID

HSM Error	Hex Code	PKCS#11 or SFNT Defined CKR Error
LUNA_RET_CCM_UPDATE_DENIED	0x80000305	CKR_DATA_INVALID
LUNA_RET_CCM_FWUPDATE_DENIED	0x80000306	CKR_DATA_INVALID
LUNA_RET_SM_ACCESS_ID_INVALID	0x80000400	CKR_DATA_INVALID
LUNA_RET_SM_ACCESS_ALREADY_EXISTS	0x80000401	CKR_DATA_INVALID
LUNA_RET_SM_MULTIPLE_ACCESS_DISABLED	0x80000402	CKR_FUNCTION_NOT_SUPPORTED
LUNA_RET_SM_UNKNOWN_ACCESS_TYPE	0x80000403	CKR_ARGUMENTS_BAD
LUNA_RET_SM_BAD_ACCESS_HANDLE	0x80000404	CKR_DATA_INVALID
LUNA_RET_SM_BAD_CONTEXT_NUMBER	0x80000405	CKR_DATA_INVALID
LUNA_RET_SM_UNKNOWN_SESSION_TYPE	0x80000406	CKR_DATA_INVALID
LUNA_RET_SM_CONTEXT_ALREADY_ALLOCATED	0x80000407	CKR_DATA_INVALID
LUNA_RET_SM_CONTEXT_NOT_ALLOCATED	0x80000408	CKR_DEVICE_MEMORY
LUNA_RET_SM_CONTEXT_BUFFER_OVERFLOW	0x80000409	CKR_DEVICE_MEMORY
LUNA_RET_SM_TOSM_DOES_NOT_VALIDATE	0x8000040A	CKR_USER_NOT_LOGGED_IN
LUNA_RET_SM_ACCESS_DOES_NOT_VALIDATE	0x8000040B	CKR_USER_NOT_AUTHORIZED
LUNA_RET_MTK_ZEROIZED	0x80000531	CKR_MTK_ZEROIZED
LUNA_RET_MTK_STATE_INVALID	0x80000532	CKR_MTK_STATE_INVALID
LUNA_RET_MTK_SPLIT_INVALID	0x80000533	CKR_MTK_SPLIT_INVALID
LUNA_RET_INVALID_IP_PACKET	0x80000600	CKR_DEVICE_ERROR
LUNA_RET_INVALID_BOARD_TYPE	0x80000700	CKR_DEVICE_ERROR
LUNA_RET_ECC_NOT_SUPPORTED	0x80000601	CKR_FUNCTION_NOT_SUPPORTED
LUNA_RET_ECC_BUFFER_OVERFLOW	0x80000602	CKR_DEVICE_ERROR
LUNA_RET_ECC_POINT_INVALID	0x80000603	CKR_ECC_POINT_INVALID **
LUNA_RET_ECC_SELF_TEST_FAILURE	0x80000604	CKR_DEVICE_ERROR
LUNA_RET_ECC_UNKNOWN_CURVE	0x80000605	CKR_ECC_UNKNOWN_CURVE

HSM Error	Hex Code	PKCS#11 or SFNT Defined CKR Error
LUNA_RET_HA_NOT_SUPPORTED	0x80000900	CKR_FUNCTION_NOT_SUPPORTED
LUNA_RET_HA_USER_NOT_INITIALIZED	0x80000901	CKR_OPERATION_NOT_INITIALIZED
LUNA_RET_HSM_STORAGE_FULL	0x80000902	CKR_HSM_STORAGE_FULL
LUNA_RET_CONTAINER_OBJECT_STORAGE_FULL	0x80000903	CKR_CONTAINER_OBJECT_STORAGE_FULL
LUNA_RET_KEY_NOT_ACTIVE	0x80000904	CKR_KEY_NOT_ACTIVE
LUNA_RET_CB_NOT_SUPPORTED	0x80000a01	CKR_FUNCTION_NOT_SUPPORTED
LUNA_RET_CB_PARAM_INVALID	0x80000a02	CKR_CALLBACK_ERROR
LUNA_RET_CB_NO_MEMORY	0x80000a03	CKR_DEVICE_MEMORY
LUNA_RET_CB_TIMEOUT	0x80000a04	CKR_CALLBACK_ERROR
LUNA_RET_CB_RETRY	0x80000a05	CKR_CALLBACK_ERROR
LUNA_RET_CB_ABORTED	0x80000a06	CKR_CALLBACK_ERROR
LUNA_RET_CB_SYS_ERROR	0x80000a07	CKR_CALLBACK_ERROR
LUNA_RET_CB_HIOS_HANDLE_INVALID	0x80000a10	CKR_CALLBACK_ERROR
LUNA_RET_CB_HIOS_ID_INVALID	0x80000a11	CKR_CALLBACK_ERROR
LUNA_RET_CB_HIOS_CLOSED	0x80000a12	CKR_CALLBACK_ERROR
LUNA_RET_CB_HIOS_CANCELED	0x80000a13	CKR_CALLBACK_ERROR
LUNA_RET_CB_HIOS_IO_ERROR	0x80000a14	CKR_CALLBACK_ERROR
LUNA_RET_CB_HIOS_SEND_TIMEOUT	0x80000a15	CKR_CALLBACK_ERROR
LUNA_RET_CB_HIOS_RECV_TIMEOUT	0x80000a16	CKR_CALLBACK_ERROR
LUNA_RET_CB_HIOS_STATE_INVALID	0x80000a17	CKR_CALLBACK_ERROR
LUNA_RET_CB_HIOS_OUTPUT_BUFFER_TOO_SMALL	0x80000a18	CKR_CALLBACK_ERROR
LUNA_RET_CB_HIOS_INPUT_BUFFER_TOO_SMALL	0x80000a19	CKR_CALLBACK_ERROR
LUNA_RET_CB_HANDLE_INVALID	0x80000a20	CKR_CALLBACK_ERROR
LUNA_RET_CB_ID_INVALID	0x80000a21	CKR_CALLBACK_ERROR
LUNA_RET_CB_REMOTE_ABORT	0x80000a22	CKR_CALLBACK_ERROR
LUNA_RET_CB_REMOTE_CLOSED	0x80000a23	CKR_CALLBACK_ERROR

HSM Error	Hex Code	PKCS#11 or SFNT Defined CKR Error
LUNA_RET_CB_REMOTE_ABANDONED	0x80000a24	CKR_CALLBACK_ERROR
LUNA_RET_CB_MUST_READ	0x80000a25	CKR_CALLBACK_ERROR
LUNA_RET_CB_MUST_WRITE	0x80000a26	CKR_CALLBACK_ERROR
LUNA_RET_CB_INVALID_CALL_FOR_THE_STATE	0x80000a27	CKR_CALLBACK_ERROR
LUNA_RET_CB_SYNC_ERROR	0x80000a28	CKR_CALLBACK_ERROR
LUNA_RET_CB_PROT_DATA_INVALID	0x80000a29	CKR_CALLBACK_ERROR
LUNA_RET_LOG_FILE_NOT_OPEN	0x80000d00	CKR_LOG_FILE_NOT_OPEN
LUNA_RET_LOG_FILE_WRITE_ERROR	0x80000d01	CKR_LOG_FILE_WRITE_ERROR
LUNA_RET_LOG_BAD_FILE_NAME	0x80000d02	CKR_LOG_BAD_FILE_NAME
LUNA_RET_LOG_FULL	0x80000d03	CKR_LOG_FULL
LUNA_RET_LOG_NO_KCV	0x80000d04	CKR_LOG_NO_KCV
LUNA_RET_LOG_BAD_RECORD_HMAC	0x80000d05	CKR_LOG_BAD_RECORD_HMAC
LUNA_RET_LOG_BAD_TIME	0x80000d06	CKR_LOG_BAD_TIME
LUNA_RET_LOG_AUDIT_NOT_INITIALIZED	0x80000d07	CKR_LOG_AUDIT_NOT_INITIALIZED
LUNA_RET_LOG_RESYNC_NEEDED	0x80000d08	CKR_LOG_RESYNC_NEEDED
LUNA_RET_AUDIT_LOGIN_TIMEOUT_IN_PROGRESS	0x80000d09	CKR_AUDIT_LOGIN_TIMEOUT_IN_PROGRESS
LUNA_RET_AUDIT_LOGIN_FAILURE_THRESHOLD	0x80000d0a	CKR_AUDIT_LOGIN_FAILURE_THRESHOLD

* This error (CKR_TEMPLATE_INCONSISTENT) might be encountered when using CKDemo in a new client with firmware older than version 6.22.0. Try CKDemo option 98, sub-option 16. If it is set to "enhanced roles", try selecting it to set it to "legacy Luna roles". The setting is a toggle, and flips every time you select it.

** This error, or "unable to read public key", might be encountered when using BSAFE to encrypt data with ECC public key using curves from the Brainpool suite. As indicated on the BSAFE website (May 2012) they do not appear to support Brainpool curves. Therefore, your own applications should not attempt that combination, and you should avoid attempting to specify Brainpool curves with BSAFE ECC when using SafeNet's CKDemo utility.

Library Codes

Hex value	Decimal value	Return code/error description
0	0	OKAY, NO ERROR
0xC0000000	3221225472	PROGRAMMING ERROR: RETURN CODE
0xC0000001	3221225473	OUT OF MEMORY
0xC0000002	3221225474	NON-SPECIFIC ERROR
0xC0000003	3221225475	UNEXPECTED NULL POINTER
0xC0000004	3221225476	PROGRAMMING ERROR: LOGIC
0xC0000005	3221225477	OPERATION WOULD BLOCK IF ATTEMPTED
0xC0000006	3221225478	BUFFER IS TOO SMALL
0xC0000100	3221225728	OPERATION CANCEL
0xC0000101	3221225729	INVALID SLOT IDENTIFIER
0xC0000102	3221225730	INVALID DATA
0xC0000103	3221225731	INVALID PIN
0xC0000104	3221225732	NO TOKEN PRESENT
0xC0000105	3221225733	FUNCTION IS NOT SUPPORTED
0xC0000106	3221225734	NON-CRYPTOKI ELEMENT CLONE
0xC0000107	3221225735	INVALID BUFFER SIZE FOR CHALLENGE
0xC0000108	3221225736	PIN IS LOCKED
0xC0000109	3221225737	INVALID VERSION

Hex value	Decimal value	Return code/error description
0xC000010a	3221225738	NEEDED KEY NOT PROVIDED
0xC000010b	3221225739	USER NAME IS IN USE
0xC0000200	3221225984	INVALID DISTINGUISHED ENCODING RULES CLASS
0xC0000303	3221226243	OPERATION TIMED OUT
0xC0000304	3221226244	RESET FAILED
0xC0000400	3221226496	INVALID TOKEN STATE
0xC0000401	3221226497	DATA APPEARS CORRUPTED
0xC0000402	3221226498	INVALID FILENAME
0xC0000403	3221226499	FILE IS READ-ONLY
0xC0000404	3221226500	FILE ERROR
0xC0000405	3221226501	INVALID OBJECT IDENTIFIER
0xC0000406	3221226502	INVALID SOCKET ADDRESS
0xC0000407	3221226503	INVALID LISTEN SOCKET
0xC0000408	3221226504	CACHE IS NOT CURRENT
0xC0000409	3221226505	CACHE IS NOT MAPPED
0xC000040a	3221226506	OBJECT IS NOT IN LIST
0xC000040b	3221226507	INVALID INDEX
0xC000040c	3221226508	OBJECT ALREADY EXISTS
0xC000040d	3221226509	SEMAPHORE ERROR

Hex value	Decimal value	Return code/error description
0xC000040e	3221226510	END OF LIST ENCOUNTERED
0xC000040f	3221226511	WOULD ASSIGN SAME VALUE
0xC0000410	3221226512	INVALID GROUP NAME
0xC0000411	3221226513	NOT HSM BACKUP TOKEN
0xC0000412	3221226514	NOT PARTITION BACKUP TOKEN
0xC0000413	3221226515	SIM NOT SUPPORTED
0xC0000500	3221226752	SOCKET ERROR
0xC0000501	3221226753	SOCKET WRITE ERROR
0xC0000502	3221226754	SOCKET READ ERROR
0xC0000503	3221226755	CLIENT MESSAGE ERROR
0xC0000504	3221226756	SERVER DISCONNECTED
0xC0000505	3221226757	CLIENT DISCONNECTED
0xC0000506	3221226758	SOCKET WOULD BLOCK
0xC0000507	3221226759	SOCKET ADDRESS IS IN USE
0xC0000508	3221226760	SOCKET BAD FILE DESCRIPTOR
0xC0000509	3221226761	HOST RESOLUTION ERROR
0xC000050a	3221226762	INVALID HOST CERTIFICATE
0xC0000600	3221227008	NO BUFFER AVAILABLE
0xC0000601	3221227009	INVALID ENUMERATION OPTION

Hex value	Decimal value	Return code/error description
0xC0000700	3221227264	SSL ERROR
0xC0000701	3221227265	SSL CTX ERROR
0xC0000702	3221227266	SSL CIPHER LIST ERROR
0xC0000703	3221227267	SSL CERT VERIFICATION LOCATION ERROR
0xC0000704	3221227268	SSL LOAD SERVER CERT ERROR
0xC0000705	3221227269	SSL LOAD SERVER PRIVATE KEY ERROR
0xC0000706	3221227270	SSL VALIDATE SERVER PRIVATE KEY ERROR
0xC0000707	3221227271	SSL CREATE SSL ERROR
0xC0000708	3221227272	SSL LOAD CLIENT CERT ERROR
0xC0000709	3221227273	SSL GET CERTIFICATE ERROR
0xC000070a	3221227274	SSL INVALID CERT STRUCTURE
0xC000070b	3221227275	SSL LOAD CLIENT PRIVATE KEY ERROR
0xC000070c	3221227276	SSL GET PEER CERT ERROR
0xC000070d	3221227277	SSL WANT READ ERROR
0xC000070e	3221227278	SSL WANT WRITE ERROR
0xC000070f	3221227279	SSL WANT X509 LOOKUP ERROR
0xC0000710	3221227280	SSL SYSCALL ERROR
0xC0000711	3221227281	SSL FAILED HANDSHAKE
0xC0000800	3221227520	INVALID CERTIFICATE TYPE

Hex value	Decimal value	Return code/error description
0xC0000900	3221227776	INVALID PORT
0xC0000901	3221227777	SESSION SCRIPT EXISTS
0xC0001000	3221229568	PARTITION LOCKED
0xC0001001	3221229569	PARTITION NOT ACTIVATED
0xc0002000	3221233664	FAILED TO CREATE THREAD
0xc0002001	3221233665	CALLBACK ERROR
0xc0002002	3221233666	UNKNOWN CALLBACK COMMAND
0xc0002003	3221233667	SHUTTING DOWN
0xc0002004	3221233668	REMOTE SIDE DISCONNECTED
0xc0002005	3221233669	SOCKET CLOSED
0xC0002006	3221233670	INVALID COMMAND
0xC0002007	3221233671	UNKNOWN COMMAND
0xC0002008	3221233672	UNKNOWN COMMAND VERSION
0xC0002009	3221233673	FILE LOCK FAILED
0xC0002010	3221233680	FILE LOCK ERROR
0xc0002011	3221233681	FAILED TO CREATE PROCESS
0xc0002012	3221233682	USB PED NOT FOUND
0xc0002013	3221233683	USB PED NOT RESPONDING
0xc0002014	3221233684	USB PED OPERATION CANCELLED

Hex value	Decimal value	Return code/error description
0xc0002015	3221233685	USB PED TOO MANY CONNECTED
0xc0002016	3221233686	USB PED OUT OF SYNC
0xC0001100	3221229824	UNABLE TO CONNECT

Vendor-defined Return Codes

Code	Name
0x00000141	CKR_INSERTION_CALLBACK_NOT_SUPPORTED
0x0052	CKR_FUNCTION_PARALLEL
0x00B2	CKR_SESSION_EXCLUSIVE_EXISTS
(CKR_VENDOR_DEFINED + 0x04)	CKR_RC_ERROR
(CKR_VENDOR_DEFINED + 0x05)	CKR_CONTAINER_HANDLE_INVALID
(CKR_VENDOR_DEFINED + 0x06)	CKR_TOO_MANY_CONTAINERS
(CKR_VENDOR_DEFINED + 0x07)	CKR_USER_LOCKED_OUT
(CKR_VENDOR_DEFINED + 0x08)	CKR_CLONING_PARAMETER_ALREADY_EXISTS
(CKR_VENDOR_DEFINED + 0x09)	CKR_CLONING_PARAMETER_MISSING
(CKR_VENDOR_DEFINED + 0x0a)	CKR_CERTIFICATE_DATA_MISSING
(CKR_VENDOR_DEFINED + 0x0b)	CKR_CERTIFICATE_DATA_INVALID
(CKR_VENDOR_DEFINED + 0x0c)	CKR_ACCEL_DEVICE_ERROR
(CKR_VENDOR_DEFINED + 0x0d)	CKR_WRAPPING_ERROR
(CKR_VENDOR_DEFINED + 0x0e)	CKR_UNWRAPPING_ERROR
(CKR_VENDOR_DEFINED + 0x0f)	CKR_MAC_MISSING
(CKR_VENDOR_DEFINED + 0x10)	CKR_DAC_POLICY_PID_MISMATCH
(CKR_VENDOR_DEFINED + 0x11)	CKR_DAC_MISSING
(CKR_VENDOR_DEFINED + 0x12)	CKR_BAD_DAC
(CKR_VENDOR_DEFINED + 0x13)	CKR_SSK_MISSING
(CKR_VENDOR_DEFINED + 0x14)	CKR_BAD_MAC

Code	Name
(CKR_VENDOR_DEFINED + 0x15)	CKR_DAK_MISSING
(CKR_VENDOR_DEFINED + 0x16)	CKR_BAD_DAK
(CKR_VENDOR_DEFINED + 0x17)	CKR_SIM_AUTHORIZATION_FAILED
(CKR_VENDOR_DEFINED + 0x18)	CKR_SIM_VERSION_UNSUPPORTED
(CKR_VENDOR_DEFINED + 0x19)	CKR_SIM_CORRUPT_DATA
(CKR_VENDOR_DEFINED + 0x1a)	CKR_USER_NOT_AUTHORIZED
(CKR_VENDOR_DEFINED + 0x1b)	CKR_MAX_OBJECT_COUNT_EXCEEDED
(CKR_VENDOR_DEFINED + 0x1c)	CKR_SO_LOGIN_FAILURE_THRESHOLD
(CKR_VENDOR_DEFINED + 0x1d)	CKR_SIM_AUTHFORM_INVALID
(CKR_VENDOR_DEFINED + 0x1e)	CKR_CITS_DAK_MISSING
(CKR_VENDOR_DEFINED + 0x1f)	CKR_UNABLE_TO_CONNECT
(CKR_VENDOR_DEFINED + 0x20)	CKR_PARTITION_DISABLED
(CKR_VENDOR_DEFINED + 0x21)	CKR_CALLBACK_ERROR
(CKR_VENDOR_DEFINED + 0x22)	CKR_SECURITY_PARAMETER_MISSING
(CKR_VENDOR_DEFINED + 0x23)	CKR_SP_TIMEOUT
(CKR_VENDOR_DEFINED + 0x24)	CKR_TIMEOUT
(CKR_VENDOR_DEFINED + 0x25)	CKR_ECC_UNKNOWN_CURVE
(CKR_VENDOR_DEFINED + 0x26)	CKR_MTK_ZEROIZED
(CKR_VENDOR_DEFINED + 0x27)	CKR_MTK_STATE_INVALID
(CKR_VENDOR_DEFINED + 0x28)	CKR_INVALID_ENTRY_TYPE
(CKR_VENDOR_DEFINED + 0x29)	CKR_MTK_SPLIT_INVALID
(CKR_VENDOR_DEFINED + 0x2a)	CKR_HSM_STORAGE_FULL
(CKR_VENDOR_DEFINED + 0x2b)	CKR_DEVICE_TIMEOUT
(CKR_VENDOR_DEFINED + 0x2C)	CKR_CONTAINER_OBJECT_STORAGE_FULL
(CKR_VENDOR_DEFINED + 0x2D)	CKR_PED_CLIENT_NOT_RUNNING
(CKR_VENDOR_DEFINED + 0x2E)	CKR_PED_UNPLUGGED
(CKR_VENDOR_DEFINED + 0x2F)	CKR_ECC_POINT_INVALID

Code	Name
(CKR_VENDOR_DEFINED + 0x30)	CKR_OPERATION_NOT_ALLOWED
(CKR_VENDOR_DEFINED + 0x31)	CKR_LICENSE_CAPACITY_EXCEEDED
(CKR_VENDOR_DEFINED + 0x32)	CKR_LOG_FILE_NOT_OPEN
(CKR_VENDOR_DEFINED + 0x33)	CKR_LOG_FILE_WRITE_ERROR
(CKR_VENDOR_DEFINED + 0x34)	CKR_LOG_BAD_FILE_NAME
(CKR_VENDOR_DEFINED + 0x35)	CKR_LOG_FULL
(CKR_VENDOR_DEFINED + 0x36)	CKR_LOG_NO_KCV
(CKR_VENDOR_DEFINED + 0x37)	CKR_LOG_BAD_RECORD_HMAC
(CKR_VENDOR_DEFINED + 0x38)	CKR_LOG_BAD_TIME
(CKR_VENDOR_DEFINED + 0x39)	CKR_LOG_AUDIT_NOT_INITIALIZED
(CKR_VENDOR_DEFINED + 0x3A)	CKR_LOG_RESYNC_NEEDED
(CKR_VENDOR_DEFINED + 0x3B)	CKR_AUDIT_LOGIN_TIMEOUT_IN_PROGRESS
(CKR_VENDOR_DEFINED + 0x3C)	CKR_AUDIT_LOGIN_FAILURE_THRESHOLD
(CKR_VENDOR_DEFINED + 0x3D)	CKR_INVALID_FUF_TARGET
(CKR_VENDOR_DEFINED + 0x3E)	CKR_INVALID_FUF_HEADER
(CKR_VENDOR_DEFINED + 0x3F)	CKR_INVALID_FUF_VERSION
(CKR_VENDOR_DEFINED + 0x40)	CKR_ECC_ECC_RESULT_AT_INF
(CKR_VENDOR_DEFINED + 0x41)	CKR_AGAIN
(CKR_VENDOR_DEFINED + 0x42)	CKR_TOKEN_COPIED
(CKR_VENDOR_DEFINED + 0x43)	CKR_SLOT_NOT_EMPTY
(CKR_VENDOR_DEFINED + 0x44)	CKR_USER_ALREADY_ACTIVATED
(CKR_VENDOR_DEFINED + 0x45)	CKR_STC_NO_CONTEXT
(CKR_VENDOR_DEFINED + 0x46)	CKR_STC_CLIENT_IDENTITY_NOT_CONFIGURED
(CKR_VENDOR_DEFINED + 0x47)	CKR_STC_PARTITION_IDENTITY_NOT_CONFIGURED
(CKR_VENDOR_DEFINED + 0x48)	CKR_STC_DH_KEYGEN_ERROR
(CKR_VENDOR_DEFINED + 0x49)	CKR_STC_CIPHER_SUITE_REJECTED
(CKR_VENDOR_DEFINED + 0x4a)	CKR_STC_DH_KEY_NOT_FROM_SAME_GROUP

Code	Name
(CKR_VENDOR_DEFINED + 0x4b)	CKR_STC_COMPUTE_DH_KEY_ERROR
(CKR_VENDOR_DEFINED + 0x4c)	CKR_STC_FIRST_PHASE_KDF_ERROR
(CKR_VENDOR_DEFINED + 0x4d)	CKR_STC_SECOND_PHASE_KDF_ERROR
(CKR_VENDOR_DEFINED + 0x4e)	CKR_STC_KEY_CONFIRMATION_FAILED
(CKR_VENDOR_DEFINED + 0x4f)	CKR_STC_NO_SESSION_KEY
(CKR_VENDOR_DEFINED + 0x50)	CKR_STC_RESPONSE_BAD_MAC
(CKR_VENDOR_DEFINED + 0x51)	CKR_STC_NOT_ENABLED
(CKR_VENDOR_DEFINED + 0x52)	CKR_STC_CLIENT_HANDLE_INVALID
(CKR_VENDOR_DEFINED + 0x53)	CKR_STC_SESSION_INVALID
(CKR_VENDOR_DEFINED + 0x54)	CKR_STC_CONTAINER_INVALID
(CKR_VENDOR_DEFINED + 0x55)	CKR_STC_SEQUENCE_NUM_INVALID
(CKR_VENDOR_DEFINED + 0x56)	CKR_STC_NO_CHANNEL
(CKR_VENDOR_DEFINED + 0x57)	CKR_STC_RESPONSE_DECRYPT_ERROR
(CKR_VENDOR_DEFINED + 0x58)	CKR_STC_RESPONSE_REPLAYED
(CKR_VENDOR_DEFINED + 0x59)	CKR_STC_REKEY_CHANNEL_MISMATCH
(CKR_VENDOR_DEFINED + 0x5a)	CKR_STC_RSA_ENCRYPT_ERROR
(CKR_VENDOR_DEFINED + 0x5b)	CKR_STC_RSA_SIGN_ERROR
(CKR_VENDOR_DEFINED + 0x5c)	CKR_STC_RSA_DECRYPT_ERROR
(CKR_VENDOR_DEFINED + 0x5d)	CKR_STC_RESPONSE_UNEXPECTED_KEY
(CKR_VENDOR_DEFINED + 0x5e)	CKR_STC_UNEXPECTED_NONCE_PAYLOAD_SIZE
(CKR_VENDOR_DEFINED + 0x5f)	CKR_STC_UNEXPECTED_DH_DATA_SIZE
(CKR_VENDOR_DEFINED + 0x60)	CKR_STC_OPEN_CIPHER_MISMATCH
(CKR_VENDOR_DEFINED + 0x61)	CKR_STC_OPEN_DHNIST_PUBKEY_ERROR
(CKR_VENDOR_DEFINED + 0x62)	CKR_STC_OPEN_KEY_MATERIAL_GEN_FAIL
(CKR_VENDOR_DEFINED + 0x63)	CKR_STC_OPEN_RESP_GEN_FAIL
(CKR_VENDOR_DEFINED + 0x64)	CKR_STC_ACTIVATE_MACTAG_U_VERIFY_FAIL
(CKR_VENDOR_DEFINED + 0x65)	CKR_STC_ACTIVATE_MACTAG_V_GEN_FAIL

Code	Name
(CKR_VENDOR_DEFINED + 0X66)	CKR_STC_ACTIVATE_RESP_GEN_FAIL
(CKR_VENDOR_DEFINED + 0X67)	CKR_CHALLENGE_INCORRECT
(CKR_VENDOR_DEFINED + 0X68)	CKR_ACCESS_ID_INVALID
(CKR_VENDOR_DEFINED + 0X69)	CKR_ACCESS_ID_ALREADY_EXISTS
(CKR_VENDOR_DEFINED + 0x114)	CKR_OBJECT_READ_ONLY
(CKR_VENDOR_DEFINED + 0x136)	CKR_KEY_NOT_ACTIVE

High-Availability (HA) Configuration and Operation

This chapter describes how to configure and use SafeNet HSMs to provide load-balancing and redundancy for mission-critical applications. It contains the following sections:

- ["High Availability \(HA\) Overview" below](#)
- ["Load Balancing" on page 96](#)
- ["Key Replication" on page 97](#)
- ["Failover" on page 98](#)
- ["Recovery and Reconnection" on page 101](#)
- ["Performance" on page 110](#)
- ["Standby Members" on page 112](#)
- ["Planning Your Deployment" on page 116](#)
- ["Configuring HA" on page 119](#)
- ["Using HA With Your Applications" on page 125](#)
- ["Managing and Troubleshooting Your HA Groups" on page 128](#)
- ["Adding, Removing, Replacing, or Reconnecting HA Group Members" on page 130](#)
- ["Frequently Asked Questions" on page 140](#)

High Availability (HA) Overview

SafeNet HSM products include availability and scalability capabilities for mission-critical applications that require uninterrupted up-time. These features allow you to use the SafeNet HSM client to group multiple devices, or partitions, into a single logical group – known as an HA (High Availability) group. When an HA group is defined, cryptographic services remain available to the applications that use the client, as long as at least one member in the group remains functional and connected to the application server. In addition, the client performs load balancing among the group members, allowing many cryptographic commands to be automatically distributed across the HA group, and enabling linear performance gains for many applications.

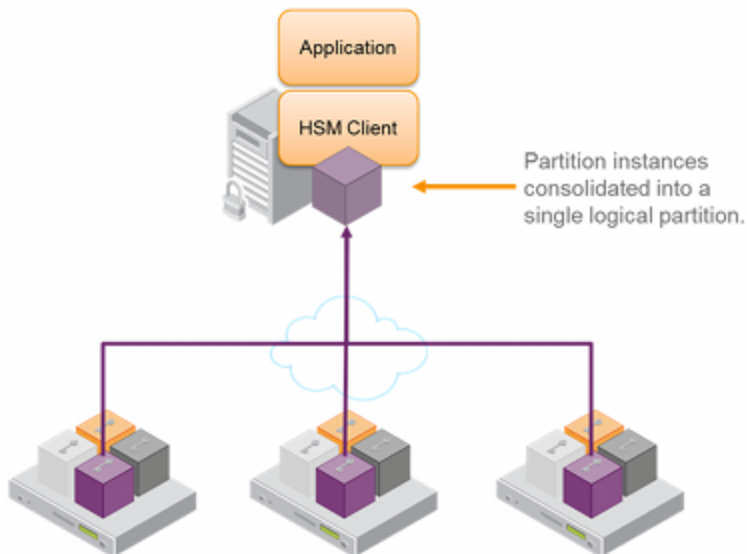
How HA is Implemented

The SafeNet high-availability (HA) and load-balancing functionality is implemented in the SafeNet HSM client, and uses the cloning¹ function to replicate/synchronize content across HA-group members. There is no direct connection between the members of an HA group. All communications between the members of an HA group are managed by the client. The HSMs and appliances are not involved and, except for being instructed to clone objects to certain HSMs during a synchronization operation, are unaware that they might be configured in an HA group. This allows you to configure HA on a per-application basis.

To create an HA group, you register the client to each HSM you want to include in the HA group, and then use the client-side administration commands to define the HA group and set any desired configuration options. You can configure several options including:

- setting automatic or manual recovery mode
- setting some HSMs as standby members
- performing various manual synchronization and recovery operations

Once defined, the SafeNet HSM client presents the HA group as a virtual slot, which is a consolidation of all the physical HSMs in the HA group. Any operations that access the slot are automatically distributed between the group members, to provide load balancing, and all key material is automatically replicated and synchronized between each member of the HA group.



Example: Database Encryption

This section walks through a specific sample use case of some of the HA logic with a specific application – namely a transparent database encryption.

¹The duplication or copying of HSM or application partition contents to other HSMs or application partitions that share the cloning domain secret. Cloning copies objects (certificates, keys, data), in a secure manner, via trusted path, from the user space on one HSM to an equivalent space on a second HSM. The trusted path can be direct connection between HSMs or application partitions on the same host, or can be via Remote Backup Protocol (RBC) between distant HSMs.

Typical Database Encryption Key Architecture

Database engines typically use a two-layered key architecture. At the top layer is a master encryption key that is the root of data protection. Losing this key is equivalent to losing the database, so it obviously needs to be highly durable. At the second layer are table keys used to protect table-spaces and/or columns. These table keys are stored with the database as blobs encrypted by the master encryption key (MEK). This architecture maps to the following operations on the HSM:

1. Initial generation of master key for each database.
2. Generation and encryption of table keys with the master key.
3. Decryption of table keys when the database needs to access encrypted elements.
4. Generation of new master keys during a re-key and then re-encrypting all table keys with it.
5. Generation and encryption of new table keys for storage in the database (often done in a software module).

The HSM is not involved in the use of table keys. Instead it provides the strong protection of the MEK which is used to protect the table keys. Users must follow backup procedures to ensure their MEK is as durable as the database itself. Refer to the backup section of this manual for proper backup procedures.

HSM High Availability with Database Encryption

When the HSMs are configured as an HA group, the database's master key is automatically and transparently replicated to all the members when the key is created or re-keyed. If an HSM group member was offline or fails during the replication, it does not immediately receive a copy of the key. Instead the HA group proceeds after replicating to all of the active members. Once a member is re-joined to the group the HSM client automatically replicates the new master keys to the recovered member.

With this in mind, before every re-key event the user should ensure the HA group has sufficient redundancy. A re-key will succeed so long as one HA group member exists, but proceeding with too few HSMs will result in an availability risk. For example, proceeding with only one HSM means the new master key will be at risk since it exists only on a single HSM. Even with sufficient redundancy, SafeNet recommends maintaining an offline backup of a database's master key.

HSM Load Balancing with Database Encryption

While a database is up and running, the master key exists on all members in the HA group. As such, requests to encrypt or decrypt table keys are distributed across the entire group. So the load-balancing feature is able to deliver improved performance and scalability when the database requires a large number of accesses to the table keys. With that said, most deployments will not need much load-balancing as the typical database deployment results in a small number of table keys.

While the table keys are re-keyed, new keys are generated in the HSM and encrypted for storage in the database. Within an HA group, these keys are generated on the primary HSM and then, even though they exist on the HSM for only a moment, they are replicated to the entire HSM group as part of the availability logic. These events are infrequent enough that this extra replication has minimal impact.

Conclusion

The SafeNet high availability and load balancing features provide an excellent set of tools to scale applications and manage availability of cryptographic services without compromising the integrity of cryptographic keys. They do not need to be copied out of an HSM and stored in a file to achieve high levels of availability. Indeed, recovery from many failures is much more rapid with Luna's keys-in-hardware approach since each HSM maintains its own copy of all keys directly inside it. A broad range of deployment options are supported that allow solution architects to achieve the

availability needed in a manner that optimizes the cost and performance without compromising the assurance of the solution.

Load Balancing

The default behavior of the client library is to attempt to load-balance the application's cryptographic requests across each active member of an HA group. Any standby members in the HA group are not used to perform cryptographic operations, and are therefore not part of the load-balancing scheme (see ["Standby Members" on page 112](#)).

The top-level algorithm is a round-robin scheme that is modified to favor the least busy device in the set. As each new command is processed, the SafeNet HSM client looks at how many commands it has scheduled on every device in the group. If all devices have an equal number of outstanding commands, the new command is scheduled on the next device in the list – creating a round-robin behavior. However, if the devices have a different number of commands outstanding on them, the new command is scheduled on the device with the fewest commands queued – creating a least-busy behavior. This modified round-robin has the advantage of biasing load away from any device currently performing a lengthy-command. In addition to this least-busy bias, the type of command also affects the scheduling algorithm, as follows:

- Single-part (stateless) cryptographic operations are load-balanced.
- Multi-part (stateful) commands are not load-balanced. Multi-part operations carry cryptographic context across individual commands. The cost of distributing this context to different HA group members is generally greater than the benefit. For this reason multi-part commands are all targeted at the primary member. Multi-part operations are typically not used, or are infrequent actions, so most applications are not affected by this restriction.
- Key management commands are not load balanced. Key management commands affect the state of the keys stored in the HSM. As such, these commands are targeted at all HSMs in the group. That is, the command is performed on the primary HSM and then the result is replicated to all members in the HA group. Key management operations are also an infrequent occurrence for most applications .

It is important to understand that the least-busy algorithm uses the number of commands outstanding on each device as the indication of its busyness. When an application performs a repeated command set, this method works very well. When the pattern is interrupted, however, the type of command can have an impact. For example, when the HSM is performing signing and an atypical asymmetric key generation request is issued, some number of the application's signing commands are scheduled on the same device (behind the key generation). Commands queued behind the key generation therefore have a large latency driven by the key generation. However, the least-busy characteristic automatically schedules more commands to other devices in the HA group, minimizing the impact of the key generation.

It is also important to note that the load-balancing algorithm operates independently in each application process. Multiple processes on the same client or on different clients do not share their "busyness" information while making their scheduling choice. In most cases this is reasonable, but some mixed use cases might cause certain applications to hog the HSMs.

Finally, when an HA group is shared across many servers, different initial members can be selected while the HA group is being defined on each server. The member first assigned to each group becomes the primary. This approach optimizes an HA group to distribute the key management and/or multi-part cryptographic operation load more equally.

In summary, the load-balancing scheme used by SafeNet is a combination of round-robin and least-busy for most operations. However, as required, the algorithm adapts to various conditions and use cases so it might not always emulate a round-robin approach.

Example

There is no "master" HSM appliance in the SafeNet Network HSM HA model. Where you might see or hear mention of a "Primary" member, that refers only to the member that happens to be the first on the configuration list. If you edit the list to place the name of a different SafeNet Network HSM on top, then that becomes the new HA Group "primary" member.

When the client makes a request on a virtual HA slot, the request goes to the first member in the HA group, as listed in the **Chrystoki.conf** file (Linux/UNIX) or **Crystoki.ini** file (Windows), unless it is busy. A member is busy if it has not yet responded to the most recent request that was sent to it. If the primary member is busy, the client sends the request to the next non-busy member of the HA Group.

In practice, that means the primary member gets all the requests until the volume reaches a level that saturates the ability of the primary, or a blocking request from another source prevents acceptance of new requests - except for load-balanced operations. So, multi-part operations, which are not load balanced because of the overhead that would be incurred due to context-tracking, are assigned to the primary by default. On a 7000 signings/second SafeNet Network HSM doing exclusively 1024-bit RSA signings, your client would need to have approximately 30 simultaneous threads offering a total of nearly 7000 requests per second before the second member would begin seeing any requests. In other words, until the primary is fully occupied, the HA group looks like it is operating as a "hot-standby" arrangement, unless some operations are explicitly redirected by load balancing.

The numbers above are ideal, of course. If you add network latency, or if you increase the key-size, or if you interleave other crypto operations, then the numbers must drop for the individual member, and the secondary member becomes part of the overall performance. And a third member, if you have a third active member in your group, and so on.

If you have any group members set to "Standby" status, then they do not contribute to group performance, even if the client can saturate the active members.

Key Replication

Whenever an application creates key material, the HA functionality transparently replicates the key material to all members of the HA group before reporting back to the application that the new key is ready. The HA library always starts with what it considers its primary HSM (initially the first member defined in an HA group). Once the key is created on the primary it is automatically replicated to each member in the group. If a member fails during this process the key replication to the failed member is aborted after the fail-over time out. If any member is unavailable during the replication process (that is, the unit failed before or during the operation), the HA library keeps track of this and automatically replicates the key when that member rejoins the group. Once the key is replicated on all active members of the HA group a success code is returned to the application.

Whether automatic or manual, object replication security is based on the use of the SafeNet cloning protocol to provide mutual authentication, confidentiality and integrity for each object that is copied from one partition to another. When partition objects are synchronized, the SafeNet HSM client is used as a secure conduit to coordinate the duplication of these objects across all partitions. An object created on LunaA partition#1A is duplicated on LunaB Partition#1B using the following process:

1. The object is created on LunaA.
2. The duplicated object is then encrypted using a key derived from common Domain material (Red Key) shared by each SafeNet HSM in the HA group.
3. LunaA transfers the encrypted object to the SafeNet Client utilizing the encrypted NTL connection between itself and the client (the object is now double encrypted).
4. The client then securely transfers the object to LunaB.
5. LunaB decrypts the object and stores it in the partition

The cloning protocol is such that it must be invoked separately for each object to be cloned and the sequence of calls required to implement the protocol must be issued by an authorized client library (residing on a client platform that has been authenticated to each of the SafeNet HSMs involved in the HA group). This ensures that the use of the cloning function calls is controlled and the protocol cannot be misused to permit the unauthorized transfer of objects to or from one of the partitions in the HA group.

Manual Synchronization

To manually synchronize the contents of the members of an HA group, use the LunaCM command **hagroup synchronize**.

Failover

When an HA group is running normally the client library continues to schedule commands across all members as described above. The client continuously monitors the health of each member at two different levels:

- First, the connectivity with the member is monitored at the networking layer. Disruption of the network connection invokes a fail-over event within a twenty second timeout.
- Second, every command sent to a device is continuously monitored for completion. Any command that fails to complete within twenty seconds also invokes a fail-over event. Most commands are completed within milliseconds. However, some commands can take extended periods to complete – either because the command itself is time-consuming (for example, key generation), or because the device is under extreme load. To cover these events the HSM automatically sends “heartbeats” every two seconds for all commands that have not completed within the first two seconds. The twenty second timer is extended every time one of these heartbeats arrives at client, thus preventing false fail-over events.

A failover event involves dropping a device from the available members in the HA group. All commands that were pending on the failed device are transparently rescheduled on the remaining members of the group. When a failure occurs, the application experiences a latency stall on some of the commands in process (on the failing unit) but otherwise sees no impact on the transaction flow . Note that the least-busy scheduling algorithm automatically minimizes the number of commands that stall on a failing unit during the twenty second timeout.

If the primary unit fails, clients automatically select the next member in the group as the new primary. Any key management or single-part cryptographic operations are transparently restarted on a new group member. In the event that the primary unit fails, any in-progress, multi-part, cryptographic operations must be restarted by the application, as the operation returns an error code.

As long as one HA group member remains functional, cryptographic service is maintained to an application no matter how many other group members fail. As discussed in "[Failover](#)" above , members can also be put back into service without restarting the application.

How Do You (or Software) Know That a Member Has Failed?

When an HA Group member first fails, the HA status for the group shows "device error" for the failed member. All subsequent calls return "token not present", until the member (HSM Partition or PKI token) is returned to service.

At the library level, what happens when a device fails or doesn't respond?

This is two separate situations. A device might not be responding because something is blocking (such as PED operations prior to HSM firmware 6.24) or, for example, because the requested operation is an RSA keygen for a keysize of 4096 or larger. Likely the device will come back. The client continues to wait so long as it receives heartbeats from the HSM (for SafeNet Network HSM, that would be as long as the NTLS connection remains alive).

A failure would be an actual failure message from the HSM (or for SafeNet Network HSM, could also be that the NTLS connection dropped). In the case of a failure, the client library drops the member and continues with others. It will try to reconnect that member at a minimum retry rate of once per minute (configurable) as long as it continues to receive heartbeats from the HSM for the pending command, and then stop trying that member. You can specify a number of retries from 3 to an unlimited number.

What happens to an application if a device fails mid-operation? What if it's a multi-part operation?

Multi part operations do NOT fail over. The entire operation returns a failure (CKR_DEVICE_ERROR). Your application deals with the failure in whatever way it is coded to do so.

Any operation that fails mid-point would need to be resent from the calling application. That is, if you don't receive a 'success' response, then you must try again. This is obviously more likely to happen in a multi-part operation because those are longer, but a failure could conceivably happen during a single atomic operation as well.

With HA, if the library attempts to send a command to an HSM and it is unavailable, it will automatically retry sending that command to the next HSM in the configuration after the timeout expires.

Multi-part operations would typically be block encryption or decryption, or any other command where the previous state of the HSM is critical to the processing of the next command. It is understandable that these need to be re-sent since the HSMs do not synchronize 'internal memory state' ... only stored key material.

Reaction to Failures

This section looks at possible failures in an overall HA system, and what needs to be done. The assumption is that HA has been properly configured and the HA group has been seen to be functioning properly. In a complex system, it is possible to come up with any number of failure scenarios, such as this (partial) list for an HA group:

- Failure at the HSM or appliance
 - HSM card failure
 - HSM re-initialization
 - Deactivated partition
 - Power failure of a member
 - Reboot of member
 - NTL failure
 - STC failure
- Failure at the client
 - Power failure of the client
 - Reboot of client
 - Network keepalive failure
- Failure between client and group members
 - Network failure near the member appliance
(so only one member might disappear from client's view)
 - Network failure near the client
(client loses contact with all members)

HSM-Side Failures

The categories of failure at the HSM side of an HA arrangement are temporary or permanent.

Temporary

Temporary failures like reboots, or failures of power or network are self-correcting, and as long as you have set HA automatic recovery parameters that are sufficiently lenient, then recovery is automatic shortly after the HSM partition becomes visible to the HA client.

Permanent

Permanent failures require overt intervention at the HSM end, including possibly complete physical replacement of the unit, or at least initialization of the HSM.

All that concerns the HA service is that the particular unit is gone, and isn't coming back. If an entire SafeNet Network HSM unit is replaced, then obviously you must go through the entire appliance and HSM configuration of a new unit, before introducing it to the HA group. If a non-appliance HSM (resides in the Client host computer, such as SafeNet PCIe HSM or SafeNet USB HSM) is replaced, then it must be initialized and a new partition created.

Either way, your immediate options are

- to use a new name for the partition, or
- to make the HA SafeNet HSM Client forget the dead member (lunacm command **ha removeMember**) so you can reuse the old name.

Then, you must ensure that automatic synchronization is enabled (lunacm command **ha synchronize -enable**), and manually introduce a new member to the group (lunacm command **ha addMember**). After that, you can carry on using your application with full HA redundancy.

Because your application should be using only the HA virtual slot (lunacm command **ha HAOnly**), your application should not have noticed that one HA group member went away, or that another one was added and synchronized. The only visible sign might have been a brief dip in performance, but only if your application was placing high demand on the HSM(s).

Client-Side Failures

For **SafeNet Network HSM**, any failure of the client (such as operating system problems), that does not involve corruption or removal of files on the host, should resolve itself when the host computer is rebooted.

- If the host seems to be working fine otherwise, but you have lost visibility of the HSMs in lunacm or your client, verify that the SafeNet drivers are running, and retry.
- If that fails, reboot.
- If that fails, restore your configuration from backup of your host computer.
- If that fails, re-install SafeNet HSM Client, re-perform certificate exchanges, creation of HA group, adding of members, setting HAOnly, etc.

For **SafeNet PCIe HSM, and SafeNet USB HSM**, the client is the host of the HSMs, so if HA has been working, then any sudden failure is likely to be

- OS or driver related (so your response is to restart) or
- corruption of files (so your response is to re-install).

If a re-install is necessary, you will need to recreate the HA group and re-add all members and re-assert all settings (like HAOnly).

Failures Between the HSM and Client (SafeNet Network HSM only)

The only failure that could likely occur between a SafeNet Network HSM (or multiple SafeNet Enterprise HSMs) and a client computer coordinating an HA group is a network failure. In that case, the salient factor is whether the failure occurred

- near the client or
- near one (or more) of the SafeNet Network HSM appliances.

If the failure occurs near the client, and you have not set up port bonding on the client, then the client would lose sight of all HA group members, and the client application would fail. The application would resume according to its timeouts and error-handling capabilities, and HA would resume automatically if the members reappeared within the recovery window that you had set.

If the failure occurs near a SafeNet Network HSM member of the HA group, then that member might disappear from the group until the network failure is cleared, but the client would still be able to see other members, and would carry on normally.

If the recovery window is exceeded, then you must manually restart HA, or use `lunacm` to trigger a manual recover request so that the application tries to recover again. Manual recovery performs only a single retry.

Recovery and Reconnection

An important aspect of the HA feature is the ability of an HA group to carry on the client's cryptographic operations when a group member becomes unavailable, or to resume operation if the client temporarily loses connection to the entire group of HSMs.

The various processes combine to ensure that your client application continues to work with your HSMs. The first section in this topic (below) deals with recovery of individual HSM units failing or losing connection and concerns the integrity of the HA group. The second section, later in this topic, deals with the client losing contact with the entire HA group, and the manner in which it re-establishes connection and context.

Recovery after failure of a member

After a failure, the recovery process is typically straightforward. Depending on the deployment, an automated or manual recovery process might be appropriate. In either case there is no need to restart an application.

Automatic recovery

With automatic recovery, the client automatically performs periodic recovery attempts while a member is failed. The frequency of these checks is adjustable and the number of re-tries can be limited. Each time a reconnection is attempted, one application command experiences a slight delay while the client attempts to recover. As such, the retry frequency cannot be set any faster than once per minute. Even if a manual recovery process is selected, the application does not need to be restarted. Simply run the client recovery command and the recovery logic inside the client makes a recovery attempt the next time the application uses the HSM. As part of recovery, any key material created while the member was offline is automatically replicated to the recovered unit .

Failed units

Sometimes a failure of a device is permanent. In this event, the only solution is to deploy a new member to the group. In this case, you can remove the failed unit from the HA group, add a new device to the group and then start the recovery process. The running clients automatically resynchronize keys to the new member and start scheduling operations to it. See ["Adding, Removing, Replacing, or Reconnecting HA Group Members" on page 130](#) for more information.

Manual recovery

Finally, sometimes both an HSM and application fail at the same time. If no new key material was created while an HSM was offline, the recovery is straightforward: simply return the HSM to service and then restart the application. However, if new key material was created after an HSM failed but before the application failed, a manual re-synchronization (using the **hagroup synchronize** command) might be required.

To perform a manual recovery:

- Confirm which member, or members, have the current key material (normally the unit(s) that was online at the time the application failed).
- Place it/(them) back in service with the application.
- For each member that has stale key material (a copy of an object that was deleted; or an old copy of an object whose attributes were changed), delete all their key material after first making sure they are not part of the HA group. Be particularly careful that the member is not part of the HA group or the action might destroy active key material by causing an accidental synchronization during the delete operation.
- After the HSM is cleared of key material, rejoin it to the group and the synchronization logic automatically repopulates the device's key material from the active units.

Usage

When a client is configured to use automatic recovery the manual recovery commands must not be used. Invoking them can cause multiple concurrent recovery processes which result in error codes and possible key corruption.

Most customers should enable auto-insert in all configurations. We anticipate that the only reason you might wish to choose manual recovery is if you do not want to impart the retry time to periodic transactions. That is, each time a recovery is attempted a single application thread experiences an increased latency while the library uses that thread to attempt the re-connection (the latency impact is a few hundred milliseconds).

Recovery Conditions

HA recovery is hands-off resumption by failed HA Group members, or it is manual re-introduction of a failed member, if automatic recovery is not enabled. Some reasons for a member to fail from the group might be:

- the appliance loses power (but regains power in less than the 2 hours that the HSM preserves its activation state)
- the network link from the unit is lost and then regained.

HA recovery takes place if the following conditions are true:

- HA automatic recovery is enabled, or if you detect a unit failure and manually re-introduce the unit (or its replacement)
- HA group has at least 2 nodes
- HA node is reachable (connected) at client startup
- HA node recover retry limit is not reached.

Otherwise manual recover is the only option to bring back the downed connection(s)

If all HA nodes fail (no links from client) no recovery is possible.

The HA recovery logic makes its first attempt at recovering a failed member when your application makes a call to its HSM (the group). That is, an idle client does not start the recovery-attempt process.

On the other hand, a busy client would notice a slight pause every minute, as the library attempts to recover a dropped HA group member (or members) until the member has been reinstated or until the timeout has been

reached and it stops trying. Therefore, set the number of retries according to your normal situation (the kinds and durations of network interruptions you experience, for example).

Enabling and Configuring Automatic Recovery

In previous releases, automatic recovery was not on by default, and needed to be explicitly enabled with `vtl haAdmin -autorecovery` command.

Beginning with SafeNet HSM release 6.0, HA automatic recovery is automatically enabled when you set the recovery retry count using the LunaCM command "[hagroup retry](#)" on [page 1](#) in the *LunaCM Reference Guide*. Use the command "[hagroup interval](#)" on [page 1](#) to specify the interval, in seconds, between each retry attempt. The default is 60 seconds.

Failure of All Members

Formerly, if all members of an HA group were to fail, then all logged-in sessions were gone, and operations that were active when the last group member went down, were terminated. Only if the client application was able to recover all that state information would it not have been necessary to restart or re-initialize in order to resume client operations with the SafeNet Network HSM HA group.

This changes with release 6.2.2 Client, where the library now preserves session state and token objects if the client loses all connection with the HA group. When the group (or member of it) becomes available again, the client application can resume the session automatically, including re-login, and all token objects that were in use before the outage are still available. See "[HA Auto-Reconnect](#)" on [page 107](#) for more detail.

Auto-insert

Automatic reintroduction or "auto-insert" is supported. A failed (and fixed, or replacement) HSM appliance can be re-introduced if the application continues without restart. Restarting the application causes it to take a fresh inventory of available HSMs, and to use only those HSMs within its HA group. You cannot [re]introduce a SafeNet Network HSM that was not in the group when the application started.

Auto-insert is now the default behavior (from Client 6.2.1 and later).

1. Auto-insert requires that "Active recovery mode" is enabled.
2. A running client automatically detects SafeNet Network HSM appliance insertion and removal to/from its configuration.
3. Connection to the new SafeNet Network HSM appliance occurs only if the client HA configuration also has a new HA member or an HA member gone missing.
4. A running client does not automatically disconnect from the appliance that has been removed from its configuration until the appliance goes offline (for example, disconnected or powered down).
5. A running client uses the new HA member that is being added to the HA group configuration and does not require the client to restart to do so.
6. A running client stops attempting to use the removed HA member that is being revoked from the HA configuration and does not require the client to restart to do so.
7. When a new member is added to the HA group, entries similar to the following appear in the client HA Log:

```
Mon Feb 1 11:06:55 2016 : [6619] HA group: 11079656446993 detected new member member:
286668019649
```

```
Mon Feb 1 11:07:25 2016 : [6619] HA group: 11079656446993 recovery attempt #1 succeeded
for member: 286668019649
```

8. When a HA member is removed from the HA group, entries similar to the following appear in the client HA Log:

```
Mon Feb 1 11:07:45 2016 : [6619] HA group: 11079656446993 member: 286668019649 revoked
```

9. When a new SafeNet Network HSM appliance is registered with a client that has HA configured with “Active recovery mode”, entries similar to the following appear in the client HA Log:

```
Sun Jan 31 21:01:52 2016 : [3820] HA subsystem detected new server : 192.20.11.175
```

```
Sun Jan 31 21:01:56 2016 : [3820] HA subsystem server 192.20.11.175 connected
```

Entries like these appear only if item 3, above, is true.

10. When an existing SafeNet Network HSM appliance is removed from client that has HA configured with “Active recovery mode”, entries similar to the following appear in the client HA Log:

```
Tue Feb 2 15:45:12 2016 : [28001] HA subsystem detected removal of server : 192.20.11.86
```

Synchronization

Synchronization of token objects is a manual process using the **hagroup synchronize** command. Synchronization locates any object that exists on any one physical HSM partition (that is a member of the HA group), but not on all others, and replicates that object to any partitions (among the group) where it did not exist.

This is distinct from the replication that occurs when you create or delete an object on the HA virtual slot. Creation or deletion against the virtual slot causes that change to be immediately replicated to all connected members (addition or deletion).

Effect of PED Operations

PED operations block cryptographic operations, so that while a member of an HA is performing a PED operation, it will appear to the HA group as a failed member. When the PED operation is complete, failover and recovery HA logic are invoked to return the member to normal operation.

Effect of Application Restarts

If an HA group member fails and an application restarts before the failed member recovers, it is not possible to recover that device until you restart the application again.

This is as designed. You originally had your application running with X number of members. One failed, but was not removed from the group, so retries were occurring, but the application was operating with X-1 members available. Then you restarted. When the application came up after that restart, it saw only X-1 members. Having just started, it now has no notion that the Xth member exists. You cannot add to that number within an application. To go from the number that the application now recognizes, X-1, to the new, larger number of participants X-1 +1 (or X), you must restart the application while all X members are available.

Network failures

If network connectivity fails to one or more connected SafeNet Network HSM appliances, the HA group will be restored automatically subject to timeouts and retries, as follows:

- While the client application is active, and one HA group member is connected and active, other members can automatically resume in the HA group as long as retries have not stopped.
- If all members fail or if the client does not have a network connection to at least one group member, then HA

auto-reconnect now preserves the session and token-object state ("[HA Auto-Reconnect](#)" on page 107). You must set `hagroup recoverymode` to "activeEnhanced" for the recovery to be completely hands-free.

Process interaction

Other events and processes interact at different levels and in different situations as described below.

At the lowest communication level, the transport protocol (TCP) is responsible for making and operating the communication connection between client and appliance (whether HA is involved or not). For SafeNet Network HSM, the default protocol timeout of 2 hours was much too long, so SafeNet configured that to 10 seconds when HA is involved. This means that:

- In a period of no activity by client or appliance, the appliance's TCP will wonder if the client is still there, and will send a packet after 10 seconds of silence.
- If that packet is acknowledged, the 10-second TCP timer restarts, and the cycle repeats indefinitely.
- If the packet is NOT acknowledged, then TCP sends another after 10 seconds, and then another after a further 10 seconds. At the two minute mark, with no response, the connection is considered dead, and higher levels are alerted to perform their cleanup.

So altogether, a total of two minutes can elapse since the last time the other participant was heard from. This is at the transport layer.

Above that level, the NTLS layer provides the connection security and some other housekeeping. Any time a client sends a request for a cryptographic operation, the HSM on the appliance begins working on that operation.

While the HSM processes the request, appliance-side NTLS sends a "keep-alive PING" every two seconds, until the HSM returns the answer, which NTLS then conveys across the link to the requesting client. NTLS (nor any layer above) does not perform any interpretation of the ping.

It simply drops a slow, steady trickle of bytes into the pipe, to keep the TCP layer active. This normally has little effect, but if your client requests a lengthy operation like (say) an 8192-bit keygen, then the random-number-generation portion of that operation could take many minutes to complete, during which the HSM would legitimately be sending nothing back to the client. The NTLS ping ensures that the connection remains alive during long pauses.

Configuration settings

In the SafeNet configuration file, "*DefaultTimeout*" (default value is 500 seconds) governs how long the client will wait for a result from an HSM, for a cryptographic call. In the case of SafeNet Network HSM, the copy of the config file inside the appliance is not accessible externally. The config file in the client installation is accessible to modify, but "*DefaultTimeout*" in that file affects only a locally connected HSM (such as might be the case if you had a SafeNet Remote Backup HSM attached to your client computer). The config file in the client has no effect on the configuration inside the network-attached SafeNet Network HSM appliance, and thus can have no effect on the interaction between client and SafeNet Network HSM appliance.

ReceiveTimeout is how long the library will wait for a dropped connection to come back.

If the *ReceiveTimeout* is tripped, for a given appliance, the HA client stops talking to that appliance and deals with the remaining members of the HA group to serve your application's crypto requests.

A minute later, the HA client tries to contact the member that failed to reply.

If the connection is successfully re-established, the errant appliance resumes working in the group, being assigned application calls as needed (governed by application workload and HA logic).

If the connection is not successfully re-established, the client continues working with the remaining group members. Another minute passes, and the client once again tries the missing appliance to see if it is ready to actively resume working in the HA group.

The retries continue until the missing member resumes, or until the pre-set (by you) number of retries is reached (maximum of 500). If the retry count is reached with no success, the client stops trying that member. The failed appliance is still a member of the group (it is still in the list of HA group members maintained on the client), but the client no longer tries to send it application calls, and no longer encourages it to establish a connection. You must fix the appliance (or its network connection) and manually recover it into the group for the client to resume including it in operations.

HA Automatic add/remove feature

The automatic add/remove feature allows individual HA group members to be added to the group or removed from it, automatically. The feature is available from release 6.2.1 and later, and has the following characteristics.

- It works only when “Active recovery mode” is enabled.
- A running client automatically detects SafeNet Network HSM appliance insertion-to and removal-from its configuration.
- Connection to the SafeNet Network HSM appliance takes place only if the client HA configuration also has a new HA member or an HA member gone missing.
- A running client does not automatically disconnect from the appliance that has been removed from its configuration until the appliance goes offline (for example, disconnected or powered down).
- A running client uses the new HA member that is being added to the HA group configuration and does not require the client to restart to do so.
- A running client stops using the removed HA member that is being revoked from the HA configuration and does not require the client to restart to do so.

- When a new member is added to the HA group, entries like the following appear in the client HA Log:

```
Mon Feb 1 11:06:55 2016 : [6619] HA group: 11079656446993 detected new member member:
286668019649
Mon Feb 1 11:07:25 2016 : [6619] HA group: 11079656446993 recovery attempt #1 succeeded
for member: 286668019649
```

- When an HA member is removed from the HA group, entries like the following appear in the client HA log:

```
Mon Feb 1 11:07:45 2016 : [6619] HA group: 11079656446993 member: 286668019649 revoked
```

- When a new SafeNet Network HSM appliance is registered, the client with HA configured with “Active recovery mode” shows entries similar to the following in the client HA log:

```
Sun Jan 31 21:01:52 2016 : [3820] HA subsystem detected new server : 192.20.11.175
Sun Jan 31 21:01:56 2016 : [3820] HA subsystem server 192.20.11.175 connected (*)
(*This log entry appears only if the third bullet-point paragraph in this list is true "...only if the client HA
configuration also has a new HA member or an HA member gone missing." )
```

- When an existing SafeNet Network HSM appliance is removed, the client with HA configured with Active recovery mode, shows entries similar to the following in the client HA log:

```
Tue Feb 2 15:45:12 2016 : [28001] HA subsystem detected removal of server : 192.20.11.86
```

- The client re-scans the configuration file every ten seconds, ensuring that a maximum of 10 seconds would elapse before a new member is picked up, or an existing one is removed due to disconnection or failure.
- All changes to the configuration file should be made by standard VTL and LunaCM commands. No new commands or options are added for this feature; it is invoked automatically when **hagroup recoverymode -mode activeBasic** or **hagroup recoverymode -mode activeEnhanced** is set.



Note: `hagroup recoveryMode` is now active by default. "Passive" mode no longer exists.

- For this feature to be available, the client software must be version 6.2.1 or newer. There is no appliance version or HSM firmware version dependency.

See also "[HA Auto-Reconnect](#)" below for the situation where the application client has become disconnected from the entire HA group.

Solaris (and other Unix)

Due to a problem in the TCP/IP configuration of some Solaris systems, inconvenient delays may have been experienced with some Solaris clients.

The problem occurred if an application was started on a Solaris client while one or more expected SafeNet Network HSM appliances is unavailable. The Solaris client machine experienced a considerable delay (minutes) before the remaining SafeNet Enterprise HSMs could be seen and used by the application. This was a TCP/IP setup issue in Solaris, in which the system attempted to set up sockets for each expected connection, and retried the unsuccessful attempts until timeout, before permitting successful connections to proceed.

To control this problem, the client-side library now imposes a ten-second retry window per expected appliance, and then moves on. (Thus, if your Client was configured to use three SafeNet Network HSM appliances, and two of them were unavailable, the Client would retry the first missing appliance for ten seconds, then the second missing appliance for a further ten seconds, for a total of twenty seconds of retries, before resuming operation with the remaining available appliance). This applies to Linux and Unix variants.

For Windows, the per-appliance timeout is 24 seconds.

HA Auto-Reconnect

This section deals with reconnection of the operating client to the HA group, when the client has lost connection to all members of the HA group. This might occur due to network connectivity issues, or if a socket is closed by any of:

- firewall rules/operation
- switch problems
- actions of operating systems, including User Account Control (UAC)
- settings and actions of ESXi hosts
- migration of running Virtual Machines (such as using VMotion)
- other causes

If an application is Java or C based, then the application can call `c_finalize`, `c_initialize`, `c_opensession`, and `c_login` in order to continue without need for an application restart. But when, for example, an application is using Microsoft-based products with LunaCSP or LunaKSP, there is no opportunity to perform the sequence of calls to resume operation. We address this problem with improvements to HA recovery mode, in which the PKCS#11 context and sessions are preserved.

Active Automatic recovery (the way it was)

Older versions of HA used a passive recovery process that was triggered only in conjunction with a PKCS#11 library call. This method was not sufficiently robust, and was replaced in release 6.2 by active automatic recovery using an HA Active Recovery Thread (ARCT). The ARCT sends a non-session-based message that is processed by NTLS.

This allows recovery as soon as a failed member returns, and does not wait for a PKCS#11 operation. Thus, if a failed member returns to duty before an active member fails, then synchronization occurs immediately, and the secondary member is ready to take over from the active member if that now fails.

Members can reconnect without the need to call `finalize/initialize` in the client application, which allows, for example multiple services that use a single JVM to recover connections independently.

In the event that all HA members failed to respond to the ARCT managing message, the HA slot was deemed to be unrecoverable.

Note: As of release 6.2.2, "passive" recovery mode is discontinued, and two modes of active recovery are available:



"activeBasic" using manual opening of session and manual login
and

"activeEnhanced" with automatic reopening of session and auto login

Preserving HA context (the HA Auto Reconnect way)

The above feature discussion dealt with scenarios where individual HA group members suffered a fault or lost network connection with the client, but left a client-and-HA group vulnerable to the situation where a connection failure cut the client off from the entire HA group.

With Client Release 6.2.2, the client HA context is preserved, and Token Object status is retained, allowing reconnection to the HA group without need to restart client applications, and with no need to finalize the existing session and start a new one. Only ephemeral session objects are lost. In summary, previous enhancements improved the handling where individual members were disconnected, but the client maintained connection to at least one group member. The enhancement described here handles the situation where the HA client loses contact with the entire group; it can now recover and resume operation if the connection is restored.

What do the HA Auto Reconnect options mean?

OPTION: `hagroup recoverymode -mode activeBasic` implies that context is automatically preserved, but the recovery from a complete client disconnection and reconnection requires intentional manual intervention. The known object handles (result of the most recent `findobject` search) allow the application to continue to use those known-object handles after the owner of the application manually intervenes to reopen a session and re-login.

This allows customers, who wish to do so, to use the HA feature while conforming to PKCS#11 standard handling after a `CKR_TOKEN_NOT_PRESENT` error.

OPTION: `hagroup recoverymode -mode activeEnhanced` implies that context is automatically preserved (including known-object handles), and the recovery from a client disconnection and reconnection does *not* require any manual intervention. Reopening a session and re-logging-in are automatic.

This option serves customers who require hands-off recovery.

What about timeouts and retries?

The boundaries of this feature are established by the general HA settings. That is, the ability to reconnect from a total separation of the client from its HA group is constrained by the retry, interval, and timeout settings that govern the loss and recovery of individual members from and to the group. No additional settings are possible or required.

The number and frequency of connection retries for an individual member also applies to retries for the entire group. Any limit that you have set, after which the client will no longer attempt to reconnect to a member, is also the limit after which the client will no longer attempt to reconnect to a lost group, and will need manual intervention.

LunaCM commands affected include ["hagroup recoverymode" on page 1](#) and ["hagroup listgroups" on page 1](#)

When to use HA Auto Reconnect

The feature is useful when your application is making use of token objects. Use activeBasic mode when strict PKCS#11 compliance is necessary. Use activeEnhanced mode when hands-free resumption of your application upon HA reconnect is the primary concern.

Steps to use HA Auto Reconnect

1. Configure an HA group, as instructed in ["Configuring HA" on page 119](#)
2. HA reconnect depends on HA autoRecovery. Enable HA autoRecovery by setting the recovery retry count with ["hagroup retry" on page 1](#) :

```
lunacm:> hagroup retry -count -1
```

3. Specify the interval, in seconds, between retry attempts with command ["hagroup interval" on page 1](#) :

```
lunacm:> hagroup interval-interval 120
```

4. Enable HAOnly with command ["hagroup haonly" on page 1](#) :

```
lunacm:> hagroup HAOnly enable
```

5. HA logging is highly recommended. Set it with command ["hagroup halog" on page 1](#) :

```
lunacm:> haGroup halog -maxlength 2560000
```

```
HA Log maximum file size was successfully set to 2560000.
```

```
Command Result : No Error
```

```
lunacm:> hagroup halog -path "c:\Program Files\SafeNet\LunaClient\halog"
```

```
HA Log path successfully set to c:\Program Files\SafeNet\LunaClient\halog.
```

```
Command Result : No Error
```

6. Set the recovery mode to "activeEnhanced" with command ["hagroup recoverymode" on page 1](#) :

```
lunacm:> hagroup recoverymode -mode activeEnhanced
```

7. Resume using your application with the HA group.



Note: Use of HA Auto Reconnect is transparent to your application. You can change between activeBasic and activeEnhanced modes without restarting your application.



Note: If you encounter error CKR_TOKEN_NOT_PRESENT, modify your application to retry the operation every minute.

If your application is generating session objects, those are not preserved. They will need to be recreated, and will appear with new handles.

Performance

For repetitive operations, like a high volume of signings using the same key, an HA group can expand SafeNet Network HSM performance in linear fashion as HA group members are added. HA groups of 16 members have undergone long-term, full-throttle testing, with excellent results.

Do keep in mind that simply adding more and more SafeNet Network HSM appliances to an HA group is not an infallible recipe for endless performance improvement. For best overall performance, all HA group members should be driven near their individual performance "sweet spot", which for SafeNet Network HSM 5.2 and later is around 30 simultaneous threads per HSM. If you assemble an HA group that is considerably larger than your server(s) can drive, then you might not achieve full performance from all.

The best approach is an HA group balanced in size for the capability of the application servers that will be driving the group, and the expected loads - with an additional unit to provide capacity for bursts of traffic and for redundancy.

Maximizing Performance

SafeNet Network HSM 6.x in HA can provide performance improvement for asymmetric single-part operations. Gigabit ethernet connections are recommended to maximize performance. For example, we have seen as much as a doubling of asymmetric single-part operations in a two-member group in a controlled laboratory environment (without crossing subnet boundaries, without competing traffic or other latency-inducing factors).

Multi-part operations are not load-balanced by the SafeNet HA due to the overhead that would be needed to perform context replication for each part of a multi-part operation.

Single-part cryptographic operations are load-balanced by the SafeNet HA functionality under most circumstances (see note on PE1746¹Enabled setting). Load-balancing these operations provides both scalability (better net throughput of operations) and redundancy by supporting transparent fail-over.

Performance is Dependent on the Type of Operation

Performance is also affected by the kind of operation you are performing. HA is better for performance when all HSM operations are performed on keys and material that reside within the HSM. This changes if part of the operation involves importing and unwrapping of keys; it can be instructive to consider what happens when such HSM operations are performed both with and without HA.

With HA

- One encryption (to wrap the key)
- One decryption in the HSM (to unwrap the key)
- Object Creation on the HSM (the unwrapped key is created and stored as a key object)
- Key Replication happens for HA
 - RSA 4096-bit operation used to derive a shared secret between HSM
 - Encryption of the key on the primary HA member using the shared secret

¹crypto integrated circuit within the K6 HSM (the stand-alone Luna PCI-E, and the HSM inside the Luna SA appliance).

- Decryption of the key on the secondary HA member hsm using the shared secret
- Object Creation on the second HA member
- One encryption (uses the unwrapped key object to encrypt the data)

Without HA

- One encryption (to wrap the key)
- One decryption in the HSM (to unwrap the key)
- Object Creation on the HSM (the unwrapped key is created and stored as a key object)
- One encryption (uses the unwrapped key object to encrypt the data)

From the above it is apparent that, with HA, many more operations are performed. Most significant in the above case are the RSA 4096-bit operation and the additional Object Creation performed. Those two operations are by far the slowest operations in the list, and so this type of task would have much better performance without HA.

By contrast, if the task had made use of objects already within the HSM, then at most a single synchronization would have propagated the objects to all HA members, and all subsequent operations would have seen a performance boost from HA operation. The crucial consideration is whether the objects being manipulated are constant or are constantly being replaced.

HA and FindObjects

Performance implications of HA in general, and of C_FindObjects in particular, are discussed in detail at ["Using HA With Your Applications" on page 125](#).

Briefly, C_FindObjects can be called with an option to search for ALL objects, or to search more specifically for a subset of all objects in the HA group. The search for ALL objects can be lengthy and initially slows performance of the HA group.

If your application is the type that launches and then remains running while performing ongoing crypto operations, then a call to C_FindObjects ALL at the beginning is just a momentary performance hit and then your application benefits from maximum HA performance thereafter.

However, if your application is programmed to launch, run a small number of crypto operations and then close, the use of C_FindObjects ALL can impose a significant performance penalty. For that kind of application, you should use C_FindObjects with very specific search parameters for the fastest possible creation of the minimum Virtual Objects Table necessary for your application.



Note: The cached object list is ephemeral, and only exists for the current session. If you restart the application, HA must recreate the object list cache. Best practice is to execute C_FindObjects to create the cached object list at application start up.



Note: Beginning with release 6.2.1, the initial call to C_FindObjects ALL is optimized, but we still recommend that you avoid running "C_FindObjects" with the "all" option if you can avoid it by using a more limited search.

HA and the PE1746Enabled Setting

The SafeNet HSM client accepts a configuration file entry known as "PE1746Enabled". This configures the way SafeNet HSM handles symmetric encryption and decryption operations for certain algorithms – namely ECB and

CBC modes of AES and TDES. By default (beginning with release 5.4) an entry is always present in the [Misc] section of the configuration file, and its value is set to “PE1746Enabled=0”, or unset.

To set this configuration option, “PE1746Enabled=1”.

When set, this value configures the library to use fast-path cryptography directly to symmetric encryption engines. This has the advantage of enabling high performance bulk crypto performance, but has the disadvantage of creating a direct context between the client library and the engine. This means that the library cannot easily load-balance operations across HSMs. This mode should be used only by applications that perform large data encryption operations (>1K data sizes).

When PE1746Enabled=0, the library uses its standard command path to the HSM. The advantage of this is that all single-part cryptographic operations can be load-balanced. The disadvantage is lower performance for larger data sizes. Applications should maintain this setting whenever possible to ensure the scalability and fail-over advantages.

In summary:

- when PE1746Enabled=1 load-balancing is not used for symmetric cryptographic operations; instead all symmetric operations are directed at the client’s primary member -- you see better performance, but no scalability across HSMs.
- when PE1746Enabled=0 all single-part cryptographic operations (with data size less-than-or-equal-to 1K) are load-balanced.

A single-part crypto operation is typically one that has small data sizes (< 1Kb), but is also dependent on how the library makes its API calls (PKCS #11 supports explicit multi-part API calls through the use of C_EncryptUpdate and C_DecryptUpdate). When an application uses the “Update” APIs the cryptographic operation is, by definition, multi-part. When the application does not use these APIs (i.e. uses C_EncryptInit followed by C_Encrypt) then an operation is single-part up to a 64KB data size.

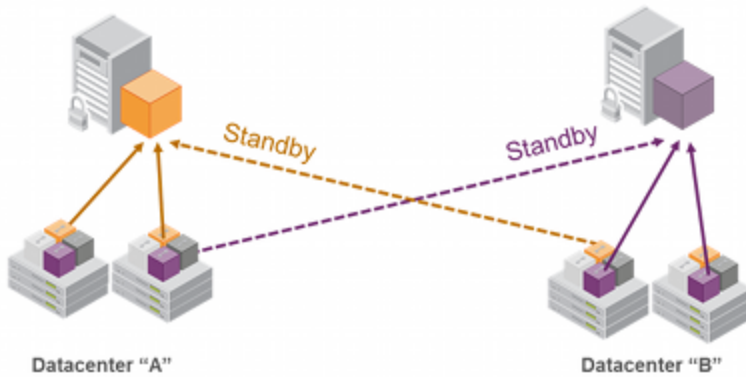
Additionally, the HSM has a limit of 1000 contexts for SafeXcel 1746 operations, which is a consideration when many client threads are involved, and depends upon the number of concurrent threads.

Whenever possible, run your application with PE1746Enabled=0.

Standby Members

By default, all members in an HA group are treated as active so that they are kept current with key material and are used to load-balance cryptographic services. In some deployment scenarios, however, it makes sense to define some members as standby. Standby members are registered just like active members except that they are defined as “standby” after they are added to the HA group.

As depicted below, applications can be deployed in geographically dispersed locations. In this scenario, you can use Luna’s standby capability to use the HSMs in the remote data center to cost-effectively improve availability. In this mode, only the local units (non-standby) are used for active load-balancing. However, as key material is created, it is automatically replicated to both the active (local) units and standby (remote) unit. In the event of a failure of all local members, the standby unit is automatically promoted to active status. You can use this feature is to reduce costs, while improving reliability. This approach allows remote HSMs that have high latency to be avoided when not needed. However, in the worst case scenario where all the local HSMs fail, the remote member automatically activates itself and keeps the application running.



Note: In normal operation, the HA standby units do not perform any cryptographic operations. However, the HA service must log into all units in a group (C_OpenSession/Login is performed against all members), including standby units. This is necessary because, in the case where the standby unit is called into action, it must already be up-to-date with respect to key material that is being used in the group - it cannot synchronize with HSMs that have failed or that have gone off-line. Therefore, when the HA group consists of PED-authenticated HSMs, they must all be Activated, including the standby HSM(s).

Standby Behaviour

Standby members become active only to keep the group alive. In an HA group that includes more than one standby member, if all active members go down/off-line, only one standby member becomes active in the group. Other standby members remain on standby until/unless that first standby member goes down, at which point the next standby member becomes active.

In other words, in an HA group, the load-sharing and redundancy capability is as large as all the active members, but if all active members become unavailable to the application, then the group load-sharing and redundancy consists of one member, even if other standby members are still available.



Note: Member configuration should be complete before implementing an HA group. Changing an HSM's status from active to standby while it is performing cryptographic operations may result in unexpected behavior.

To set an HSM to standby status

In ["Configuring HA" on page 119](#), we created an HA group with label "myHAGroup" and group number 742276409, with two active members, serial number 65003001 and serial number 65005001.

1. Create a third member, as previously described, and add it to the HA group.

```
lunacm: hagroup addmember -group 742276409 -serialnumber 66010002
```

```
Member 65005002 successfully added to group 742276409.
```

```
Group configuration is:
```

```
HA Group Label: myHAGroup
HA Group Number: 742276409
HA Group Slot ID: Not Available
Synchronization: enabled
Group Members: 65003001, 65005001, 66010002
```

Needs sync: no

Slot #	Member S/N	Member Label	Status
=====	=====	=====	=====
0	65003001	sal75legpar1	alive
1	65005001	sal72legpar1	alive
1	66010002	sal72legpar1	alive

Command Result : No Error

LunaCM v6.0.0 - Copyright (c) 2006-2015 SafeNet, Inc.

Available HSMs:

```
Slot Id -> 0
Label -> sal75legpar1
Serial Number -> 65003001
Model -> LunaSA
Firmware Version -> 6.22.0
Configuration -> Luna User Partition, No SO (PW) Signing With Cloning Mode
Slot Description -> Net Token Slot
```

```
Slot Id -> 1
Label -> sal72legpar1
Serial Number -> 65005001
Model -> LunaSA
Firmware Version -> 6.22.0
Configuration -> Luna User Partition, No SO (PW) Signing With Cloning Mode
Slot Description -> Net Token Slot
```

```
Slot Id -> 2
Label -> sal77legpar1
Serial Number -> 66010002
Model -> LunaSA
Firmware Version -> 6.22.0
Configuration -> Luna User Partition, No SO (PW) Signing With Cloning Mode
Slot Description -> Net Token Slot
```

```
Slot Id -> 3
HSM Label -> myHAGroup
HSM Serial Number -> 742276409
HSM Model -> LunaVirtual
HSM Firmware Version -> 6.22.0
HSM Configuration -> Luna Virtual HSM (PW) Signing With Cloning Mode
HSM Status -> N/A - HA Group
```

Current Slot Id: 0

2. Set the member to standby status.

```
lunacm: hagroup addstandby -group 742276409 -serialnumber 66010002
Member 65005002 successfully added to group 742276409.
Group configuration is:
  HA Group Label: myHAGroup
  HA Group Number: 742276409
```

```

HA Group Slot ID: Not Available
Synchronization: enabled
Group Members: 65003001, 65005001, 66010002
Needs sync: no

```

Slot #	Member S/N	Member Label	Status
0	65003001	sal75legpar1	alive
1	65005001	sal72legpar1	alive
1	66010002	sal72legpar1	standby

Command Result : No Error

LunaCM v6.0.0 - Copyright (c) 2006-2015 SafeNet, Inc.

Available HSMs:

```

Slot Id -> 0
Label -> sal75legpar1
Serial Number -> 65003001
Model -> LunaSA
Firmware Version -> 6.22.0
Configuration -> Luna User Partition, No SO (PW) Signing With Cloning Mode
Slot Description -> Net Token Slot

```

```

Slot Id -> 1
Label -> sal72legpar1
Serial Number -> 65005001
Model -> LunaSA
Firmware Version -> 6.22.0
Configuration -> Luna User Partition, No SO (PW) Signing With Cloning Mode
Slot Description -> Net Token Slot

```

```

Slot Id -> 2
Label -> sal77legpar1
Serial Number -> 66010002
Model -> LunaSA
Firmware Version -> 6.22.0
Configuration -> Luna User Partition, No SO (PW) Signing With Cloning Mode
Slot Description -> Net Token Slot

```

```

Slot Id -> 3
HSM Label -> myHAGroup
HSM Serial Number -> 742276409
HSM Model -> LunaVirtual
HSM Firmware Version -> 6.22.0
HSM Configuration -> Luna Virtual HSM (PW) Signing With Cloning Mode
HSM Status -> N/A - HA Group

```

Current Slot Id: 0

Planning Your Deployment

This section describes the supported configurations and any limitations or constraints to consider when setting up an HA group.

HA Group Members

It is important that all members in an HA group have the same configuration and version. That means that each HA group member must use the same authentication method, either PED-authenticated or password-authenticated, and be at the same software version. Running HA groups with different versions is unsupported. Ensure that HSMs are configured identically to ensure smooth high availability and load balancing operation. SafeNet HSMs come with various key management configurations: cloning mode, key-export mode, etc. HA functionality is supported with both cloning and Scalable Key Storage variants – provided all members in the group have the same configuration. Clients automatically and transparently use the correct secure key replication method based on the group's configuration.

It is also critical that all members in an HA group share the same Security Domain role (Red PED key for PED-authenticated devices, or domain password for password-authenticated devices). The Security Domain defines which HSMs are allowed to share key material. Because HA group members are, by definition, intended to be peers, they must be in the same Security Domain.

The SafeNet HA and load-balancing feature works on a per-client and per-partition bases. This provides a lot of flexibility. For example, it is possible to define a different sub-set of HSMs in each client and even in each client's partitions (in the event that a single client uses multiple partitions). SafeNet recommends to avoid these complex configurations and to keep the HA topography uniform for an entire HSM. That is, treat HSM members at the HSM level as atomic and whole. This simplifies the configuration management associated with the HA feature.

Mix and Match Software Is Not Supported

All SafeNet Network HSM appliances in an HA group must be at the same revision level. If you have SafeNet Network HSM units at different version levels, perform updates as necessary, before attempting to create an HA group -this applies to the system software version, not to the HSM firmware, which **can** differ among group members.

Mix and Match Firmware Is Not Recommended

Generally, keep all HA members at the same firmware version. As well, all members should have the same optional capability updates applied. If mismatches are permitted among members, synchronization might be disrupted if your application attempts to use a mechanism or a capability that not all members support. In the previous section, we indicate that HSM firmware can differ between members of an HA group, but this is not intended for ongoing operation; rather, it allows you to keep all members within a group while you individually update their firmware, to ensure minimal disruption during the updates.

While it is possible to have HSMs with different firmware versions within an HA group, this is not generally recommended. Be aware that the capability of the group (in terms of features and available algorithms) is that of the member with the oldest firmware.

For example, if you had an HA group that included HSMs with two different firmware versions, then certain capabilities that are part of the newer firmware would be unavailable to Clients connecting to the HA group. Specifically, operations that make use of newer cryptographic mechanisms and algorithms would likely fail. The client's calls might be initially assigned to a newer-firmware HSM and could therefore appear to work for a time, but if the task was load-balanced to an HSM that did not support the newer features it would fail. Similarly, if the newer-firmware HSM dropped out of the group, operations would fail. Your Clients must not invoke those algorithms because not every member of the group supports them. The solution is to upgrade the older units to the most recent firmware and software versions (where possible) or else to limit clients to only the lowest supported feature set.

HA Group Members Must Not Be on the Same Appliance

In any one HA group, always ensure that member partitions or member PKI tokens (USB-attached SafeNet USB HSMs, or SafeNet CA4/PCM token HSMs in a USB-attached SafeNet DOCK2 card reader) are on different / separate appliances. **Do not** attempt to include more than one HSM partition or PKI token (nor one of each) from the same appliance in a single HA group. This is not a supported configuration. Allowing two partitions from one HSM, or a partition from the HSM and an attached HSM (as for PKI), into a single HA group would defeat the purpose of HA by making the SafeNet appliance a potential single-point-of-failure.

Running HA on a group of Scalable Key Storage SafeNet Network HSM appliances

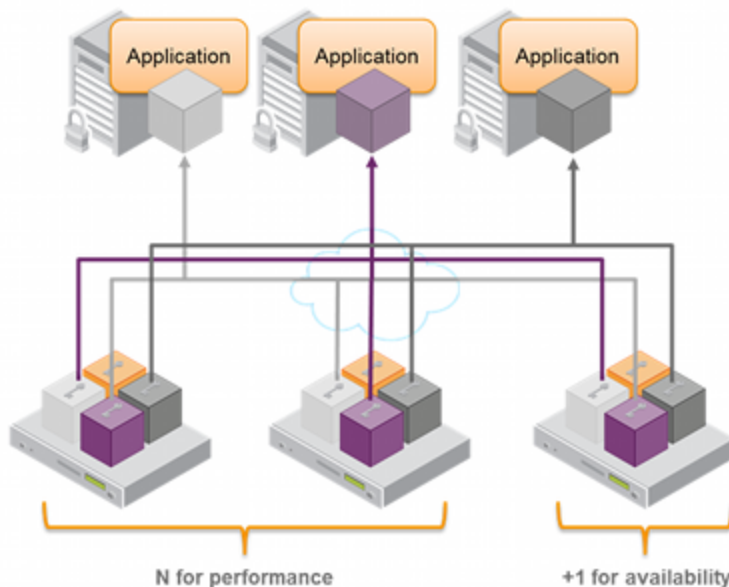
SIM replication is supported. HA will work, but key replication must be performed manually, that is, key creation in such an environment will fail to replicate.

Running HA on a group of export SafeNet Network HSM appliances

This configuration is supported, although you cannot clone/replicate private keys.

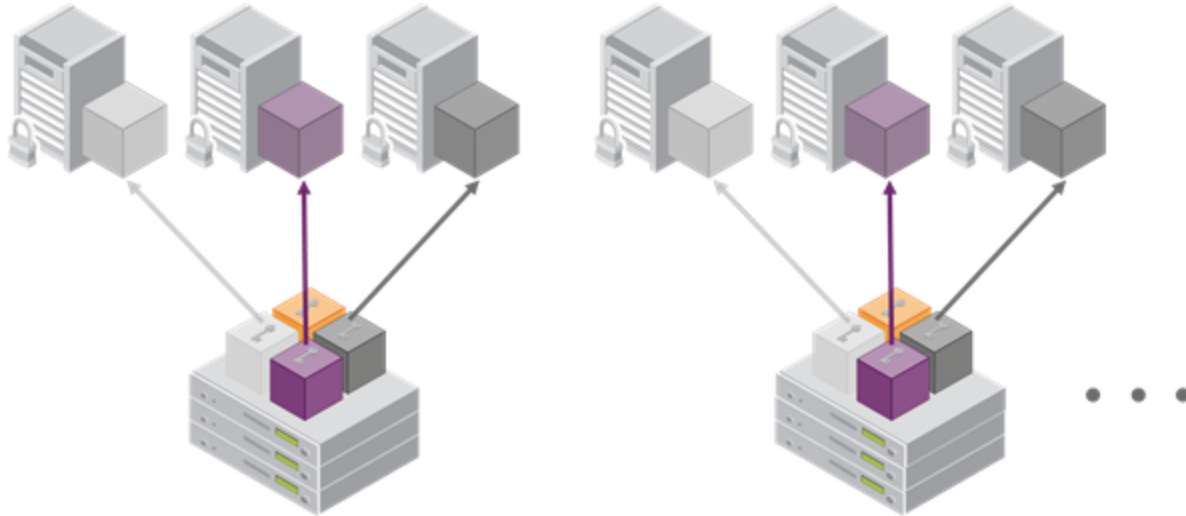
High Availability Group Sizing

As of SafeNet HSM release 6.x, the high availability function supports the grouping of up to thirty-two members. However, the maximum practical group size for your application is driven by a trade-off between performance and the cost of replicating key material across the entire group. A common practice is to set the group size to $N+1$ where N is defined by the desired performance per application server(s). As depicted below, this solution gives the desired performance with a single extra HSM providing the availability requirement. The number of HSMs per group of application servers varies based on the application use case but, as depicted, groups of three are typical.



As performance needs grow beyond the performance capacity of three HSMs, it often makes sense to define a second independent group of application servers and HSMs to further isolate applications from any single point of

failure. This has the added advantage of facilitating the distribution of HSM and application sets in different data centers.



Network Requirements

The network topography of the HA group is generally not important to the proper functioning of the group. As long as the client has a network path to each member the HA logic will function. Keep in mind that having a varying range of latencies between the client and each HA member causes a command scheduling bias towards the low-latency members. It also implies that commands scheduled on the long-latency devices have a larger overall latency associated with each command. In this case, the command latency is a characteristic of the network; to achieve uniform load distribution ensure that latencies to each device in the group are similar (or use standby mode). Gigabit Ethernet network connections are recommended.

Upgrading and Redundancy and Rotation

For SafeNet Network HSM HA function we suggest that all SafeNet Network HSM appliances in an HA group be at the same appliance software and firmware level. The issue is not about firmware level, per se - what might happen is that a newer firmware could contain newer algorithms that are not supported in the replaced firmware. If your client is configured to take advantage of newer/better algorithms when they become available, it might do so while one member of an HA group has new firmware, but another member has not yet been updated, and therefore does not yet support the requested algorithm. The client might not be able to interpret the resulting imbalance. Therefore, when you intend to upgrade/update any of the SafeNet Network HSM units in an HA group, or when you intend to upgrade/update the SafeNet Network HSM Client software, you might schedule some downtime for your application, if you anticipate a problem.

If the application is so critical that you cannot permit that much scheduled downtime, then you can set up a second complete set of Client computer and associated HA group. One set can service the application load while the other set is being upgraded or otherwise maintained. For such up-time-critical applications, you might already have such a backup set of Client-plus-HA-group that you would rotate in and out of service during regular maintenance windows.

Configuring HA

For this section you need at least two SafeNet Network HSM appliances with PED Authentication, or two with Password Authentication. You may not use Password Authenticated SafeNet Network HSM and PED Authenticated SafeNet Network HSM simultaneously in an HA group.

Partitions that are to take part in an HA group do not need to be identical (see below for the example that mixes several differences), but they should have the same firmware version and generally similar Policy settings, to avoid conflicts. For example, you would not want to have a group with a mix of partitions, some with FIPS mode switched on and some with FIPS mode switched off, because a call for a non-FIPS-approved operation would fail on any member that is not allowed to perform that operation, and attempts to synchronize the contents of group members would fail to replicate objects that were not permitted on some members. The library is not aware of individual member settings; only whether the members are available when needed, or not.

Note: You must Activate individual HSM partitions directly and individually - you cannot perform Activation on a virtual HA partition.



In general, when an HA group is established, you (or your applications) can interact with the virtual partition to perform crypto operations, and the library decides which physical partitions are involved - based on load and other considerations - but administrative activities must be performed directly on individual physical HSM partitions.

Now proceed to create the HA group.

Create the HA Group



Note: Your LunaCM instance needs to update the **Chrystoki.conf** (Linux/UNIX) or **crystoki.ini** file (Windows) when setting up or reconfiguring HA. Ensure that you have sufficient privileges.

After creating partitions

- on (at least) two SafeNet appliances, and setting up NTLS between those partitions and your client, or
- on two HSMs on the local host, or
- on a mix of local and remote application partitions,

use LunaCM to configure HA on your client.

For this example, assume

- two local HSMs,
- two remote HSM appliances (one partition from each)
- a mix of PSO partitions and legacy partitions (not required, just mentioning so the slot list distribution is obvious, and to show that it is possible to mix - HA is not affected),
- a mix of firmware versions (illustrating that it is possible to mix f/w versions in HA - but remember that the group has the capabilities of the oldest firmware, not any newer)
- each partition has the same password/challenge secret (previously set by command **role changePW -oldpw**)

<pw> **-newpw** with the old and new partition challenge/password secrets specified in the command, to invoke changing the secondary credentials),

- each partition is activated (the partition has Policies 22 and 23 turned on, and an Owner/Crypto Officer (or Crypto User) authentication has been performed)

```
C:\Program Files\SafeNet\LunaClient>lunacm
LunaCM v15.11.16-135. Copyright (c) 2006-2016 SafeNet, Inc.
```

Available HSMs:

```

Slot Id -> 0
Label -> mylegacypar1
Serial Number -> 16298193222735
Model -> LunaSA 6.2.0
Firmware Version -> 6.24.0
Configuration -> Luna User Partition, No SO (PED) Signing With Cloning Mode
Slot Description -> Net Token Slot

Slot Id -> 1
Label -> mysapsopar1
Serial Number -> 16298193222734
Model -> LunaSA 6.2.0
Firmware Version -> 6.24.0
Configuration -> Luna User Partition With SO (PED) Signing With Cloning
Mode
Slot Description -> Net Token Slot

Slot Id -> 2
Tunnel Slot Id -> 4
Label -> parwithpso
Serial Number -> 349297122742
Model -> K6 Base
Firmware Version -> 6.24.0
Configuration -> Luna User Partition With SO (PED) Signing With Cloning
Mode
Slot Description -> User Token Slot

Slot Id -> 3
Tunnel Slot Id -> 4
Label -> mypcie6
Serial Number -> 150022
Model -> K6 Base
Firmware Version -> 6.24.0
Configuration -> Luna HSM Admin Partition (PED) Signing With Cloning Mode
Slot Description -> Admin Token Slot
HSM Configuration -> Luna HSM Admin Partition (PED)
HSM Status -> OK

Slot Id -> 5
Label -> myG5par
Serial Number -> 16302360890475
Model -> G5Base
Firmware Version -> 6.22.0
Configuration -> Luna User Partition With SO (PED) Signing With Cloning
Mode
```



```

Slot Description ->      User Token Slot

Slot Id ->              6
Label ->               SafeG5
Serial Number ->       7001812
Model ->               G5Base
Firmware Version ->    6.22.0
Configuration ->       Luna HSM Admin Partition (PED) Signing With Cloning Mode
Slot Description ->    Admin Token Slot
HSM Configuration ->   Luna HSM Admin Partition (PED)
HSM Status ->         OK

```

Current Slot Id: 0

Command Result : No Error

lunacm:>

1. Use the **hagroup createGroup** command , to create the HA group with one member.

```
lunacm:> hagroup createGroup -serialNumber 349297122742 -label myhagroup -p someuserpin
```

```

New group with label "myhagroup" created with group number 1349297122742.
Group configuration is:

```

```

HA Group Label:  myhagroup
HA Group Number: 1349297122742
HA Group Slot ID: Not Available
Synchronization: enabled
  Group Members: 349297122742
    Needs sync:  no
Standby Members: <none>

```

```

Slot #      Member S/N                      Member Label      Status
=====
----- 349297122742                    parwithpso       alive

```

Command Result : No Error

LunaCM v15.11.16-135. Copyright (c) 2006-2015 SafeNet, Inc.

Available HSMs:

```

Slot Id ->              0
Label ->               mylegacypar1
Serial Number ->       16298193222735
Model ->               LunaSA 6.2.0
Firmware Version ->    6.24.0
Configuration ->       Luna User Partition, No SO (PED) Signing With Cloning Mode
Slot Description ->    Net Token Slot

Slot Id ->              1

```

```

Label ->          mysapsopar1
Serial Number -> 16298193222734
Model ->          LunaSA 6.2.0
Firmware Version -> 6.24.0
Configuration -> Luna User Partition With SO (PED) Signing With Cloning
Mode

Slot Description -> Net Token Slot

Slot Id ->        2
Tunnel Slot Id -> 4
Label ->          parwithpso
Serial Number -> 349297122742
Model ->          K6 Base
Firmware Version -> 6.24.0
Configuration -> Luna User Partition With SO (PED) Signing With Cloning
Mode

Slot Description -> User Token Slot

Slot Id ->        3
Tunnel Slot Id -> 4
Label ->          mypcie6
Serial Number -> 150022
Model ->          K6 Base
Firmware Version -> 6.24.0
Configuration -> Luna HSM Admin Partition (PED) Signing With Cloning Mode
Slot Description -> Admin Token Slot
HSM Configuration -> Luna HSM Admin Partition (PED)
HSM Status ->    OK

Slot Id ->        5
Label ->          myG5par
Serial Number -> 16302360890475
Model ->          G5Base
Firmware Version -> 6.22.0
Configuration -> Luna User Partition With SO (PED) Signing With Cloning
Mode

Slot Description -> User Token Slot

Slot Id ->        6
Label ->          SafeG5
Serial Number -> 7001812
Model ->          G5Base
Firmware Version -> 6.22.0
Configuration -> Luna HSM Admin Partition (PED) Signing With Cloning Mode
Slot Description -> Admin Token Slot
HSM Configuration -> Luna HSM Admin Partition (PED)
HSM Status ->    OK

Slot Id ->        10
HSM Label ->      myhagroup
HSM Serial Number -> 1349297122742
HSM Model ->      LunaVirtual
HSM Firmware Version -> 6.24.0
HSM Configuration -> Luna Virtual HSM (PED) Signing With Cloning Mode
HSM Status ->    N/A - HA Group

Current Slot Id: 0

```

```
lunacm:>
```



Note: For PED-authenticated HSMs, have a SafeNet PED connected, the partition already activated, and provide the partition challenge secret as the password (must be the same for all members). For Password-authenticated HSMs, the partition password is the challenge, and must be common to all members.

The group is represented by the virtual partition, which must have the same authentication.



Note: You cannot mix PED-authenticated and Password-authenticated HSM partitions in an HA group, because the different authentication methods prevent them having the same cloning domain, which is required for HA synchronization.

- Your `chrystoki.conf/crystoki.ini` file should now have a new section:

```
VirtualToken = {
VirtualToken00Members = 65003001;
VirtualToken00SN = 742276409;
VirtualToken00Label = myHAGroup;
}
```



CAUTION: Never insert TAB characters into the `chrystoki.ini` (Windows) or `crystoki.conf` (UNIX) file.

So far, we have an HA group with one member, which is the SafeNet PCIe HSM user partition from the original slot list. Next we would add additional HSM partitions (slots) to the group, to make it a true, functional HA group.

- Use the **hagroup addmember** command to add another member to the HA group, that member being the SafeNet USB HSM user partition from the original list:

```
lunacm:> hagroup addMember -slot 5 -group myhagroup -password someuserpin
Member 16302360890475 successfully added to group myhagroup. New group
configuration is:
```

```
HA Group Label: myhagroup
HA Group Number: 1349297122742
HA Group Slot ID: 10
Synchronization: enabled
Group Members: 349297122742, 16302360890475
Needs sync: no
Standby Members: <none>
```

Slot #	Member S/N	Member Label	Status
-----	-----	-----	-----
-----	349297122742	parwithpso	alive
-----	16302360890475	myG5par	alive

Please use the command "ha synchronize" when you are ready to replicate data between all members of the HA group.

(If you have additional members to add, you may wish to wait until you have added them before synchronizing to save time by avoiding multiple synchronizations.)

Command Result : No Error

lunacm:>

4. Check `Chrystoki.conf/crystoki.ini` again, the `VirtualToken` section should now look like this:

```
VirtualToken = {
VirtualToken01Label = myhagroup
VirtualToken01SN = 1349297122742;
VirtualToken01Members = 349297122742,16302360890475;
}
```

5. To extend the example, we can add one of the SafeNet Network HSM remote partitions to the group, again with command **hagroup addMember**:

```
lunacm:> hagroup addMember -slot 0 -group myhagroup -password someuserpin
Member 16298193222735 successfully added to group myhagroup. New group
configuration is:
```

```
    HA Group Label:  myhagroup
    HA Group Number: 1349297122742
    HA Group Slot ID: 10
    Synchronization: enabled
        Group Members: 349297122742, 16302360890475, 16298193222735
        Needs sync:    no
    Standby Members: <none>
```

Slot #	Member S/N	Member Label	Status
=====	=====	=====	=====
-----	349297122742	parwithpso	alive
-----	16302360890475	myG5par	alive
-----	16298193222735	mylegacypar1	alive

Please use the command "ha synchronize" when you are ready to replicate data between all members of the HA group. (If you have additional members to add, you may wish to wait until you have added them before synchronizing to save time by avoiding multiple synchronizations.)

Command Result : No Error

lunacm:>

6. Use the command **hagroup synchronize -group <grouplabel> -password <password> -enable** when you are ready to replicate data between/among all members of the HA group.

```

lunacm:> hagroup synchronize -group myhagroup -password someuserpin -enable

    HA Synchronization is already enabled

    No synchronization performed/needed.

Command Result : No Error

lunacm:>

```

If you have additional members to add, you might wish to wait until you have added them before synchronizing to save time by avoiding multiple synchronizations. The 'synchronize' command replicates all objects on all partitions across all other partitions. As there are no objects on our newly created partitions yet, we do not need to run this command.



Note: Do not use this command when recovering a group member that has failed (or was taken down for maintenance). Use the command **hagroup recover -group <grouplabel>**.

Verification Steps

- We have the two physical slots on SafeNet HSM sa175 and SafeNet HSM sa172, and now a third virtual slot which points at both physical slots at once, via load balancing. To test your HA setup, run multitoken against slot 3:

```
./multitoken -mode rsasigver -key 1024 -slots 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3
```



Note: (Each of the "3"s in the above sample invokes one thread performing the selected signing operation.)

If you are satisfied that your HA setup is working, then you can begin using your application against the HA "slot" label (which, in the example above, was "myhagroup"). If you have included more SafeNet HSM application Partitions in your HA group, then the virtual slot assignment will differ accordingly, but that doesn't matter to your application, because the application should be invoking the label, not a particular slot-number.

HA Standby Mode [optional]

If you wish to add an additional member that will be designated a standby member, and not a regular participant in the group, see ["Standby Members" on page 112](#).

Using HA With Your Applications

This section describes how HA affects your applications, and describes the settings you can use and actions you can take to mitigate any performance or stability issues.

HAOnly

By default, the client lists both the physical slots and virtual slots for the HA group. Directing applications at the physical slots bypasses the high availability and load balancing functionality. An application must be directed at the virtual slots to activate the high availability and load balancing functionality. A configuration setting referred to as

HAonly hides the physical slots, and is recommended to prevent incorrect application configurations. Doing so also simplifies the PKCS #11 slot ordering for a dynamic HA group.

What is the impact of the 'haonly' flag, and why might you wish to use it?

The "haonly" flag shows only HA slots (virtual slots) to the client applications. It does not show the physical slots. We recommend that you use "haonly", unless you have particular reason for not using it. Having "haonly" set is the proper way for clients to deal with HA groups - it prevents the possible confusion of having both physical and virtual slots available.

Recall that automatic replication/synchronization across the group occurs only if you cause a change (keygen or other addition, or a deletion) via the virtual HA slot. If your application changes the content of a physical slot, this results in the group being out-of-sync, and requires a manual re-sync to replicate a new object across all members. Similarly, if you delete from a physical slot directly, the next manual synchronization will cause the deleted object to be repopulated from another group member where that object was never deleted. Therefore, to perform a lasting deletion from a single physical slot (if you choose not to do it via the virtual slot) requires that you manually delete from every physical slot in the group, or risk your deleted object coming back.

Also, from the perspective of the Client, a member of the HA group can fail and, with "haonly" set, the slot count does not change. If "haonly" is not set, and both virtual and physical slots are visible, then failure of unit number 1 causes unit number 2 to become slot 1, and so on. That could cause problems if your application is not designed to deal gracefully with such a change.

Authentication

HA operation requires that the client(s) be able to access all member HSM partitions and the objects they contain.

For password-authenticated HSMs, this means that the HA client must provide the relevant Crypto Officer or Crypto User password. For PED-authenticated HSM, the group-member partitions must be in logged-in/activated state, and the HA client must provide the relevant challenge secret or password.

Crypto User can clone public objects

Prior to firmware 6.27.0, the Crypto Officer could create public objects, but could not clone them, which blocked HA replication of such objects when Crypto User authentication was used. With HSM firmware 6.27.0 and later, the Crypto User can clone objects that it creates. This facilitates HA operation and synchronization of such objects by Crypto User, without the need for your applications to invoke Crypto Officer authority.

Key Generation

An application that continuously generates key material will need to have its HA group carefully selected. Contact SafeNet support for help in architecting your HA group for these applications.

Example

Multi-token is a general-purpose demonstration/exercise tool for SafeNet HSMs. It is not optimized for all tasks. If you run the extract/insert options (for Scalable Key Storage) in multitoken against SafeNet Network HSM with several threads against the HA slot, performance appears to be about 10 times slower than in non-HA single slot mode.

The reason is that in this mode multitoken continuously creates session objects that need to be replicated to the additional physical HA slots. This creates overhead that does not exist in single slot mode. For optimum real-life performance, your applications should avoid this approach.

Application Object Handles

Application developers should be aware that the PKCS #11 object handle model is fully virtualized with the SafeNet HA logic. As such, the application must not assume fixed handle numbers across instances of an application. A handle's value remains consistent for the life of a process; but it might be a different value the next time the application is executed.

HA and FindObjects

HA vs Individual

When an application is addressing a single HSM, all objects in the application partition can be addressed by their handles. When an application is addressing an HA group, a layer of abstraction overlays the physical slots and presents a virtual slot to represent them. All calls and operations by applications should be directed against the virtual (HA) slot. Asserting HAOnly avoids potential problems that can arise if applications interfere directly with physical slots when HA is in force.

Inside HA

For best ongoing functioning of an application with HA, the application makes calls to examine specific objects in the virtual slot, or to perform operations on those objects. Objects are represented by handles in the physical slots, but the same object might have a different handle in two-or-more HSMs. Furthermore, those handles could change if an HSM drops out (fails or loses communication) and is replaced. The HA functionality deals with the differences by establishing a virtual-to-physical mapping, such that an application requests a virtual handle, and HA "invisibly" takes care of matching that virtual handle to actual physical objects.

In order to make those mapping connections, HA refers to a Virtual Object Table that establishes a virtual handle for OUIDs in the group. If the mapping of virtual handle to physical already exists, it was previously determined from that Virtual Object Table. If there is no existing match of virtual handle to physical, then HA looks in the Virtual Handle Table to discover the OUID associated with that virtual handle, then goes to the physical slot (HSM application partition) to retrieve the physical handle that matches the OUID. The physical handle is matched to the virtual handle, and now the application can examine attributes or perform crypto operations on the mapped object, needing only to know the virtual handle.

C_FindObjects Behavior

The next question, of course, is where did the Virtual Object Table come from? The Virtual Object Table, that assigned virtual handles to physical-object OUIDs in the group, was created by HA responding to your application's C_FindObjects call sometime after the client application was launched - that is, the table is ephemeral and must be recreated for each instance of your application. It is then maintained by HA while your application is open. The table is populated:

- partially if your C_FindObjects asks for specific attributes, or
- fully if your C_FindObjects searches for ALL objects.

A non-specific C_FindObjects ALL can be a lengthy process, since it must search through all physical objects. A more narrow, focused and specific C_FindObjects generally takes less time.

If an earlier call to C_FindObjects resulted in a partial Virtual Object Table, and then a later call to C_FindObjects looks for objects that were not previously included, then the Virtual Object Table is expanded incrementally.

The HA client builds a Virtual Object Table for **each** running application that loads the Luna library.

Application Interaction with HA

In the previous section, most of the described action is invisible to your application. Your application makes a call to a crypto slot (the HA virtual slot) and gets a response back. Between those two events, HA is taking care of

- intercepting the call,
- translating virtual to physical,
- launching actions on the appropriate HSM,
- receiving the result, and
- forwarding that result to your application,
- as well as ensuring that any changes in objects on the physical HSM that performed the action are propagated to all members of the HA group.

Some client applications launch once and then run continuously for a long time. In this case, once the mapping of physical object handles to virtual object handles has completed, the Virtual-to-Physical table is available and can be referred instantly, with no need to rebuild (the time-consuming part).

Some client applications launch, perform a search, then close quickly. If such an application can search by description, handles, or other attributes, then minimal performance penalty is imposed. If such an application (open-and-shut type) performs C_FindObjects ALL before running a call against an object, then this imposes a significant performance penalty, as the complete Virtual Object Table must be constructed each time.

HA needs to be aware of only the objects that your application needs to use; a larger or less-specific search that returns more objects than required by your application is wasteful, and degrades performance unnecessarily.



Note: The cached object list is ephemeral, and only exists for the current session. If you restart the application, HA must recreate the object list cache. Best practice is to execute C_FindObjects to create the cached object list at application start up.



Note: Beginning with release 6.2.1, the initial call to C_FindObjects ALL is optimized, but we still recommend that you avoid running "C_FindObjects" with the "all" option if you can avoid it by using a more limited and focused search.

Managing and Troubleshooting Your HA Groups

You can use VTL and the LunaCM **hagroup** commands to monitor and manage your HA groups.

Slot Enumeration

The client-side utility command "vtl listslot" or the LunaCM **slot list** command shows all detected slots, including HSM partitions on the primary HSM, partitions on connected external HSMs, and HA virtual slots. Here is an example:

```
bash-3.2# ./vtl listslot
```

```
Number of slots: 11
```

The following slots were found:

Slot #	Description	Label	Serial #	Status
slot #1	LunaNet Slot	-	-	Not present
slot #2	LunaNet Slot	sa76_p1	150518006	Present

slot #3	LunaNet Slot	sa77_p1	150475010	Present
slot #4	LunaNet Slot	G5179	700179008	Present
slot #5	LunaNet Slot	pki1	700180008	Present
slot #6	LunaNet Slot	CA4223	300223001	Present
slot #7	LunaNet Slot	CA4129	300129001	Present
slot #8	HA Virtual Card Slot	-	-	Not present
slot #9	HA Virtual Card Slot	-	-	Not present
slot #10	HA Virtual Card Slot	ha3	343610292	Present
slot #11	HA Virtual Card Slot	G5_HA	1700179008	Present



Note: - The deploy/undeploy of a PKI device increments/decrements the SafeNet Network HSM client slot enumeration list (slots appear or disappear from the list, and the slot numbers adjust for the change). HA group virtual slots always appear toward the end of the list, following the physical slots. The actual slot number can vary based on the currently connected external HSMs (tokens, G5).

Due to the above behavior, we generally recommend that you run the `lunacm:> haGroup haonly` command, or the `vtl haAdmin HAOnly enable` command, so that only the HA slot is visible and any confusion or improper slot use is eliminated.

Determining Which Device is in Use

Use the “`ntls show`” command.

Determining Which Devices are Active

CA extension call “`CA_GetHAState`” lists all active devices. The LunaCM **hagroup listgroup** command also lists members.

Duplicate Objects

If you create an object on your HA slot, and then duplicate that object in some fashion (for example, by Scalable Key Storage'ing [wrapping] it off and then back on again, or performing a backup/restore with the 'add' option), that object will be seen as only one object on the HA slot because HA uses the object's fingerprint to build an object list. Two objects will in fact exist on each of the physical slots and could be seen by a non-HA utility/query to the HSM.

There are TWO implications from this situation:

- One implication is that repeated duplication (perhaps an application that performs periodic backups, and restores using the 'add' option rather than 'replace') could cause the Partition to reach the maximum number of Partition objects while seemingly having fewer objects. If the system ever tells you that your Partition is full, but HA says otherwise, then use a tool like `ckdemo` that can view the "physical" slots directly (as opposed to the HA slot) on the HSM, and delete any objects that are unnecessary.
- A second implication is that the HA feature uses object fingerprints to match different instances of an object on different physical HSMs. This can result in error messages if your application does not properly create and destroy session objects, and perhaps creates an object identical to one which has been removed in a separate concurrent session. The problem is self-correcting, but the flurry of error messages could be worrying if you don't understand where they are coming from.

Adding, Removing, Replacing, or Reconnecting HA Group Members

This section describes how add a new member to an HA group, reconnect an offline member, or replace a failed unit.

Adding or Removing an HA Group Member

Use the following LunaCM commands to add or remove a normal or standby member to or from an HA group:

- `hagroup addmember`
- `hagroup addstandby`
- `hagroup removemember`
- `hagroup removestandby`

See "[hagroup](#)" on [page 1](#) in the *LunaCM Command Reference Guide* for detailed descriptions and syntax for each `hagroup` command.



Note: You must restart the application to have the added or removed member recognized.

Reconnecting an Off-line Unit

In HA mode, if an HSM appliance goes off-line/drops-out (due to failure, maintenance, or other reason), the application load is spread over the remaining HSM Partitions on appliances in the HA Group. When the unit is restarted, the application does **not** need be stopped and restarted, before the re-introduced unit can be used by the application. For the unit that was withdrawn (or for a replacement unit), if it was powered off for more than a short outage, you must re-activate the Partitions before they can be re-included into the HA Group.

The following two reconnection scenarios are available:

To recover the same group member

1. Restart the failed member and verify that it has started properly.
2. Do not perform a manual re-synchronization between the members. Instead, use the following LunaCM command:

```
lunacm:> ha -recover -group <group_name>
```

For Auto-insert of a replacement for failed group member when in Active mode

1. Configure the new SafeNet Network HSM as follows:
 - name it differently from the failed member appliance, The name must be different to avoid any possibility of conflict between the old and new SSL certificates, which incorporate the hostnames of the respective appliances.
 - make it part of the same cloning domain as others in the HA group. At initialization, the HSM gets its cloning domain from the same red domain PED Key.

If you require that the replacement appliance must have the same name as the replaced appliance, then you will need to stop your application before introducing the new appliance.

2. Create a partition with the same characteristics as others in the HA group (password, autoActivation, auto MofN, client assignments, etc.).

3. Determine the serial number of the failed member partition.
4. Retrieve the server certificate of the new SafeNet Network HSM.
5. Do one of :

- a. Replace the failed SafeNet Network HSM with the new one using the following VTL command:

```
vtl replaceServer -o <oldServerName> -n <newServerName> -c <newServerCertFile>
```

OR

- b. Delete the failed SafeNet Network HSM member and then add the new, replacing SafeNet Network HSM member using the following VTL commands:

```
vtl replaceServer -n <failedServerName>
vtl addServer -n <newServerName> -c <newServerCertFile>
```

6. Add the new partition of the new SafeNet Network HSM to the HA group using the relevant command below:

```
lunacm:> ha -addMember -group <group number> -serialNum <serialnumber>
      -password <password>
```

OR

```
lunacm:> ha -addMember -group <group number> -slot <slotnumber>
      -password <password>
```

7. Remove the failed member from the HA group, using the relevant command below:

```
lunacm:> ha -removeMember -group <groupNumber> -serialNum <serialnumber>
```

OR

```
lunacm:> ha -removeMember -group <groupNumber> -slot <slotnumber>
```

8. This step is required if lunacm is the only application running on the system. Do not perform a manual re-synchronization between the members. Instead, use the following command:

```
lunacm:> ha -recover -group <group_name>
```

To replace a failed group member with a new appliance when HA is in legacy mode

1. Configure the new SafeNet Network HSM as follows:
 - name it differently from the failed member appliance, The name must be different to avoid any possibility of conflict between the old and new SSL certificates, which incorporate the hostnames of the respective appliances.
 - make it part of the same cloning domain as others in the HA group. At initialization, the HSM gets its cloning domain from the same red domain PED Key.

If you require that the replacement appliance must have the same name as the replaced appliance, then you will need to stop your application before introducing the new appliance.

2. Create a partition with the same characteristics as others in the HA group (password, autoActivation, auto MofN, client assignments, etc.).
3. Do not delete the failed SafeNet Network HSM member from the Chrystoki.conf (Unix/Linux) or Crystoki.ini (Windows) configuration file.

4. Determine the serial number of the failed member partition.
5. Retrieve the server certificate of the new SafeNet Network HSM.
6. Replace the failed SafeNet Network HSM with the new one using the following VTL command:

```
vtl replaceServer -o <oldServerName> -n <newServerName> -c <newServerCertFile>
```
7. Add the new partition of the new SafeNet Network HSM to the HA group using the relevant command below:
 - lunacm:> ha -addMember -group <group number> -serialNum <serialnumber> -password <password>
 - lunacm:> ha -addMember -group <group number> -slot <slotnumber> -password <password>
8. Remove the failed member from the HA group, using the relevant command below:
 - lunacm:> ha -removeMember -group <groupNumber> -serialNum <serialnumber>
 - lunacm:> ha -removeMember -group <groupNumber> -slot <slotnumber>
9. Do not perform a manual re-synchronization between the members. Instead, use the following command:
 - lunacm:> ha -recover -group <group_name>

Replacing a Failed SafeNet Network HSM

Before getting into replacing HSMs in an HA group, this first section describes relevant system conditions and settings to have a SafeNet Network HSM configured and in an authenticated relationship with a client computer. In particular, we are interested in the client-side config file and the client's certificate folder in ordinary, single-appliance mode, and then in HA. You would already have set up the a SafeNet Network HSM as described in the configuration manual, for network setup and creation of the appliance-side certificate (see "Generate a New HSM Server Certificate").

Chrystoki.ini before client-side certificate creation

```
[Chrystoki2]
LibNT=C:\Program Files\SafeNet\LunaClient\cryptoki.dll

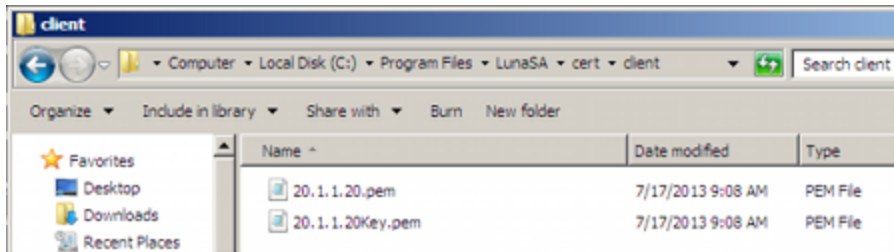
[LunaSA Client]
SSLConfigFile=C:\Program Files\SafeNet\LunaClient\openssl.cnf
ReceiveTimeout=20000
NetClient=1
ServerCAFile=C:\Program Files\SafeNet\LunaClient\cert\server\CAFile.pem
ClientCertFile=C:\Program Files\SafeNet\LunaClient\cert\client\ClientNameCert.pem
ClientPrivKeyFile=C:\Program Files\SafeNet\LunaClient\cert\client\ClientNameKey.pem

[Luna]
DefaultTimeOut=500000
PEDTimeout1=100000
PEDTimeout2=200000
PEDTimeout3=10000

[CardReader]
RemoteCommand=1
```

1. Create client-side certs (see "[vtl createCert](#)" on page 1 in the *Utilities Reference Guide*).

Generated client certificates



Chrystoki.ini after client-side certificate creation

```
[Chrystoki2]
LibNT=C:\Program Files\SafeNet\LunaClient\cryptoki.dll

[LunaSA Client]
SSLConfigFile=C:\Program Files\SafeNet\LunaClient\openssl.cnf
ReceiveTimeout=20000
NetClient=1
ServerCAFile=C:\Program Files\SafeNet\LunaClient\cert\server\CAFile.pem
ClientCertFile=C:\Program Files\SafeNet\LunaClient\cert\client\20.1.1.20.pem
ClientPrivKeyFile=C:\Program Files\SafeNet\LunaClient\cert\client\20.1.1.20.pem

[Luna]
DefaultTimeOut=500000
PEDTimeout1=100000
PEDTimeout2=200000
PEDTimeout3=10000

[CardReader]
RemoteCommand=1
```

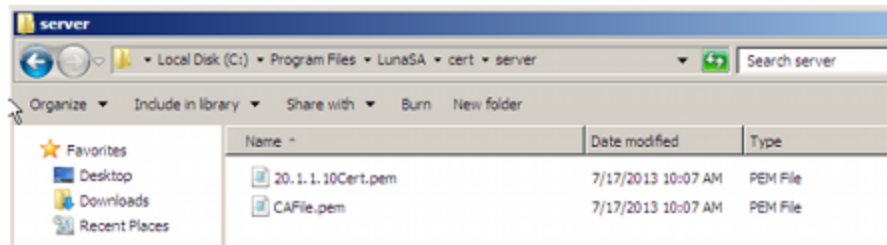
- Copy SafeNet Network HSM server.pem to client.



Note: At this point there are still no certificates in cert\server directory.

- Use "vtl addserver" to register the SafeNet Network HSM with the client.
CAFile.pem is generated in the cert\server directory.

Cert\server directory after CAFile.pem is generated



Crystoki.ini after "vtl addserver"

```
[Chrystoki2]
LibNT=C:\Program Files\SafeNet\LunaClient\cryptoki.dll
```

```
[LunaSA Client]
SSLConfigFile=C:\Program Files\SafeNet\LunaClient\openssl.cnf
ReceiveTimeout=20000
NetClient=1
ServerCAFile=C:\Program Files\SafeNet\LunaClient\cert\server\CAFile.pem
ClientCertFile=C:\Program Files\SafeNet\LunaClient\cert\client\20.1.1.20.pem
ClientPrivKeyFile=C:\Program Files\SafeNet\LunaClient\cert\client\20.1.1.20.pem
ServerName00=20.1.1.20
ServerPort00=1792
```

```
[Luna]
DefaultTimeOut=500000
PEDTimeout1=100000
PEDTimeout2=200000
PEDTimeout3=10000
```

```
[CardReader]
RemoteCommand=1
```

vtl verify results

```
C:\Program Files\SafeNet\LunaClient>vtl verify
```

The following SafeNet Network HSM Slots/Partitions were found:

Slot	Serial #	Label
====	=====	=====
1	154702010	p1

```
C:\Program Files\SafeNet\LunaClient>
```

Replace a SafeNet Network HSM Using the same IP

For an existing HA group, bring in a replacement SafeNet Network HSM.

1. Change the IP of the new appliance to match the one that was removed.
2. Perform RegenCert on the new SafeNet Network HSM.



Note: “vtl verify” on client at this time would fail because the cert that the client has is for the old, removed SafeNet Network HSM.

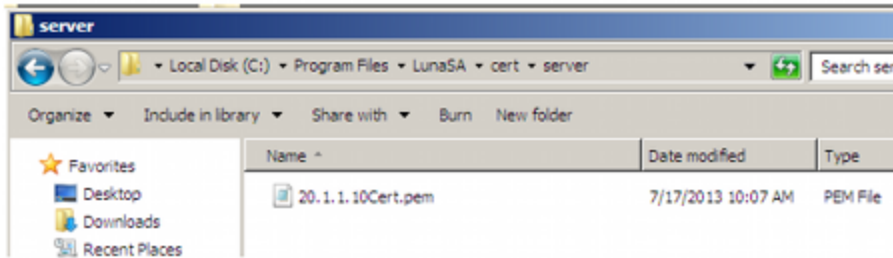
3. Execute “vtl deleteserver -n <original IP>

Deleting old SafeNet Network HSM from Client

```
C:\Program Files\SafeNet\LunaClient>vtl listservers
Server: 20.1.1.20
```

```
C:\Program Files\SafeNet\LunaClient>vtl deleteserver -n 20.1.1.20
Server: 20.1.1.20 successfully removed from server list.
```

```
C:\Program Files\SafeNet\LunaClient>
```

Contents of cert\server after “deleteserver” (CAFile.pem has been deleted)

4. Copy new server.pem to client

Copying new server.pem to client

```
C:\Program Files\SafeNet\LunaClient>pscp admin@20.1.1.20:server.pem .
admin@20.1.1.20's password:
server.pem          | 1 kB | 1.1 kB/s | ETA: 00:00:00 | 100%
```

5. Run vtl addserver using new server.pem

vtl addserver using new server.pem

```
C:\Program Files\SafeNet\LunaClient>vtl addserver -n 20.1.1.20 -c server.pem
New server: 20.1.1.20 successfully added to server list.
```

```
C:\Program Files\SafeNet\LunaClient>
```

6. Run vtl verify.

vtl verify results

```
C:\Program Files\SafeNet\LunaClient>vtl verify
```

The following SafeNet Network HSM Slots/Partitions were found:

Slot	Serial #	Label
====	=====	=====
1	154702010	p1

```
C:\Program Files\SafeNet\LunaClient>
```

Summary

If a SafeNet Network HSM must be replaced, the old IP can be used, but the SafeNet Network HSM certificate must be regenerated. The IP must be removed from the server list on the client and then added back using the new “server.pem”

Client side requirements review:

- Use `vtl deleteserver` to remove IP from list and delete CAFile.pem from cert\server
- Copy “new” server.pem to client
- Use `vtl addserver` to re-add IP and create CAFile.pem

Client-side - Reconfigure HA if a SafeNet Network HSM Must Be Replaced

1. Note HA partition serial numbers

```
C:\Program Files\SafeNet\LunaClient>vtl verify
The following SafeNet Network HSM Slots/Partitions were found:
Slot   Serial #       Label
====   =====       =====
1      154702011      HA1
1      154702012      HA2

C:\Program Files\SafeNet\LunaClient>
```

2. Run "lunacm ha -newGroup..."

A group is created with HA1 as Primary.

```
C:\Program Files\SafeNet\LunaClient>vtl haadmin -newGroup -label SomeHAGrp -serial
154702011 -password userpin
New group with label "SomeHAGrp" created at group number 1154702011.
Group configuration is:
```

```
        HA Group label:  SomeHAGrp
        HA Group Number: 1154702011
        HA Group Slot #:  unknown
        Synchronization: enabled
        Group Members:   154702011
        Standby Members: <none>
        In Sync:         yes
```

```
C:\Program Files\SafeNet\LunaClient>
```

Crystoki.ini after HA group is created

```
[Crystoki2]
LibNT=C:\Program Files\SafeNet\LunaClient\cryptoki.dll

[LunaSA Client]
SSLConfigFile=C:\Program Files\SafeNet\LunaClient\openssl.cnf
ReceiveTimeout=20000
NetClient=1
ServerCAFile=C:\Program Files\SafeNet\LunaClient\cert\server\CAFile.pem
ClientCertFile=C:\Program Files\SafeNet\LunaClient\cert\client\20.1.1.20.pem
ClientPrivKeyFile=C:\Program Files\SafeNet\LunaClient\cert\client\20.1.1.20.pem
ServerName00=20.1.1.20
ServerPort00=1792

[Luna]
DefaultTimeOut=500000
PEDTimeout1=100000
PEDTimeout2=200000
PEDTimeout3=10000

[CardReader]
RemoteCommand=1

[VirtualToken]
```



```
VirtualToken00Label=SomeHAGrp
VirtualToken00SN=1154702011
VirtualToken00Members=154702011
[HASynchronize]
SomeHAGrp=1
```

3. Add a secondary SafeNet Network HSM partition to the HA group with lunacm:> ha - addMember.

```
lunacm:> ha -addMember -group SomeHAGrp -serialNum 154702012 -password userpin
New group with label "SomeHAGrp" created at group number 1154702011.
Group configuration is:
```

```
HA Group label: SomeHAGrp
HA Group Number: 1154702011
HA Group Slot #: 6
Synchronization: enabled
  Group Members: 154702011, 154702012
Standby Members: <none>
  In Sync: yes
```

Please use the command 'vtl haAdmin -synchronize' when you are ready to replicate data among all members of the HA group. (If you have additional members to add, you might wish to wait until you have added them before synchronizing to save time by avoiding multiple synchronizations.)

```
C:\Program Files\SafeNet\LunaClient>
```

Crystoki.ini after second HA member is added

```
[Chrystoki2]
LibNT=C:\Program Files\SafeNet\LunaClient\cryptoki.dll

[LunaSA Client]
SSLConfigFile=C:\Program Files\SafeNet\LunaClient\openssl.cnf
ReceiveTimeout=20000
NetClient=1
ServerCAFile=C:\Program Files\SafeNet\LunaClient\cert\server\CAFile.pem
ClientCertFile=C:\Program Files\SafeNet\LunaClient\cert\client\20.1.1.20.pem
ClientPrivKeyFile=C:\Program Files\SafeNet\LunaClient\cert\client\20.1.1.20.pem
ServerName00=20.1.1.20
ServerPort00=1792

[Luna]
DefaultTimeout=500000
PEDTimeout1=100000
PEDTimeout2=200000
PEDTimeout3=10000

[CardReader]
RemoteCommand=1

[VirtualToken]
VirtualToken00Label=SomeHAGrp
VirtualToken00SN=1154702011
VirtualToken00Members=154702011, 154702012
```

```
[HASynchronize]
SomeHAGrp=1
```

Crystoki.ini after HA Only is enabled

```
[Chrystoki2]
LibNT=C:\Program Files\SafeNet\LunaClient\cryptoki.dll

[LunaSA Client]
SSLConfigFile=C:\Program Files\SafeNet\LunaClient\openssl.cnf
ReceiveTimeout=20000
NetClient=1
ServerCAFile=C:\Program Files\SafeNet\LunaClient\cert\server\CAFile.pem
ClientCertFile=C:\Program Files\SafeNet\LunaClient\cert\client\20.1.1.20.pem
ClientPrivKeyFile=C:\Program Files\SafeNet\LunaClient\cert\client\20.1.1.20.pem
ServerName00=20.1.1.20
ServerPort00=1792

[Luna]
DefaultTimeOut=500000
PEDTimeout1=100000
PEDTimeout2=200000
PEDTimeout3=10000

[CardReader]
RemoteCommand=1

[VirtualToken]
VirtualToken00Label=SomeHAGrp
VirtualToken00SN=1154702011
VirtualToken00Members=154702011, 154702012

[HASynchronize]
SomeHAGrp=1
```

```
[HAConfiguration]
HAOnly=1
```

Crystoki.ini after "autorecovery" is enabled

```
[Chrystoki2]
LibNT=C:\Program Files\SafeNet\LunaClient\cryptoki.dll

[LunaSA Client]
SSLConfigFile=C:\Program Files\SafeNet\LunaClient\openssl.cnf
ReceiveTimeout=20000
NetClient=1
ServerCAFile=C:\Program Files\SafeNet\LunaClient\cert\server\CAFile.pem
ClientCertFile=C:\Program Files\SafeNet\LunaClient\cert\client\20.1.1.20.pem
ClientPrivKeyFile=C:\Program Files\SafeNet\LunaClient\cert\client\20.1.1.20.pem
ServerName00=20.1.1.20
ServerPort00=1792

[Luna]
DefaultTimeOut=500000
PEDTimeout1=100000
```

```
PEDTimeout2=200000
PEDTimeout3=10000
```

```
[CardReader]
RemoteCommand=1
```

```
[VirtualToken]
VirtualToken00Label=SomeHAGrp
VirtualToken00SN=1154702011
VirtualToken00Members=154702011, 154702012
```

```
[HASynchronize]
SomeHAGrp=1
```

```
[HAConfiguration]
HAOnly=1
reconnAtt=500
```

4. Show HA configuration results with `vtl haAdmin -show`

```
C:\Program Files\SafeNet\LunaClient>vtl haadmin -show
```

```
===== HA Global Configuration Settings =====
```

```
          HA Auto Recovery:  enabled
Maximum Auto Recovery Retry:  500
Auto Recovery Poll Interval:  60 seconds
          HA Logging:        disabled
Only Show HA Slots:          yes
```

```
===== HA Group and Member Information =====
```

```
          HA Group label:  SomeHAGrp
          HA Group Number: 1154702011
          HA Group Slot #:  1
Synchronization:  enabled
          Group Members:   154702011, 154702012
          Standby Members: <none>
```

Slot #	Member S/N	Member Label	Status
-	154702011	HA1	alive
-	154702012	HA2	alive

```
C:\Program Files\SafeNet\LunaClient> >
```

Replacing the Secondary HA Group Member

When the SafeNet Network HSM to be replaced, in an HA Group, is a secondary member, the process is similar to above. You must delete the secondary from the HA Group and re-add it with the new partition serial number. It is not necessary to delete and recreate the group.

If a SafeNet Network HSM must be replaced, the old IP address can be used, but the SafeNet Network HSM certificate must be regenerated. The IP address must be removed from the server list on the client and then added back using the new “server.pem” received from the replacement SafeNet Network HSM.

If the SafeNet Network HSM being replaced is the Primary, you must delete the HA Group and recreate it using the new Primary SafeNet Network HSM partition serial number and then add the original Secondary SafeNet Network HSM partition serial number - the cert from the original Secondary is already in place on the client, and no change is needed to that.

Frequently Asked Questions

This section provides additional information by answering questions that are frequently asked by our customers.

Can we manage NTLS connections through a load balancer (like NetScaler, Barracuda, A10, etc.)?

No. NTLS will not work through a load-balancer because it is an end-to-end TLS pipe between client and SafeNet Network HSM.

We want to use a backup application server that would operate in standby mode until awakened by a failure of our primary application server. Can we use a virtual IP in the SafeNet Network HSM setup, so that both primary and secondary are accepted for NTLS as the same client by SafeNet Network HSM?

Yes. At the client, generate the client cert with the command " vtl createCert -n <any IP address, real or virtual> "

Both client computers must have the SafeNet Network HSM appliance's server cert in their client-side server-cert folders.

The SafeNet Network HSM appliance must have the client certificate (built with the virtual IP address)

Also the following lines in the Chrystoki.conf file must point to the same cert and Keyfile on the clustered application servers:

```
LunaSA Client ={
  ClientCertFile=\usr\LunaClient\cert\client\

```

Our application keeps the HSM full. Can we double the capacity by creating an HA group and having a second HSM?

No. HA provides redundancy and can increase performance, but not capacity. Every HSM in an HA group gets synchronized with the other member[s], which means that the content of any one HSM in an HA group must be a clone of the content of any other member of that group. So, with more HA group members, you get more copies, not more space.

HSM Initialization

This chapter describes how to initialize your HSM. It contains the following sections:

- ["Initialization Overview for Password-Authenticated HSMs" below](#)
- ["Initialization Overview for PED-authenticated HSMs" on page 143](#)
- ["HSM Initialization and Zeroization" on page 146](#)
- ["Re-initialize an HSM" on page 147](#)
- ["Initialize an HSM With Existing Domain and Shared PED Keys" on page 147](#)

Initialization Overview for Password-Authenticated HSMs

(This page is not instructions. This page is background information that might help make some operations more obvious.)

For SafeNet HSMs, there are two kinds of initialization:

- "hard" init - occurs when the HSM is in a factory [re]fresh state
- "soft" init - occurs when the HSM is not in factory [re]fresh state

Both are launched by the same command, `hsm init -l <hsmlabel>`.

Condition/Effect	Soft init	Hard init
SO authentication required?	Yes	No
Can set new HSM label	Yes	Yes
Creates new SO identity	No	Yes
Creates new Domain	No	Yes
Destroys partitions	Yes	No (none exist to destroy) *
Destroys SO objects	Yes	No (none exist to destroy) *

* `hsm factoryReset` was performed, and destroyed partitions and objects, before the hard init... otherwise, it could not be a hard init.

Hard Initialization

Coming from the factory, the SafeNet Network HSM:

- has network settings left over from our manufacturing process and not recommended for your production network
- has only default certificates in place

- has an undifferentiated HSM with no associations or ownership declared
- has not yet had virtual HSMs (HSM Partitions) created or assigned
- has not been introduced to the Clients (your Clients) with which it will be working.

Network setup of the appliance takes care of the first two items on that list. See "[Configure IP and Network Parameters](#)" on page 1 in the *Configuration Guide*.

Initialization takes care of the third item, which pertains specifically to the HSM portion of the appliance.

When you initialize a new (or factoryReset) HSM, several things happen, but the most important ones from your operational perspective are:

- you set up Security Officer or HSM Administrator (two names for the same entity) ownership of the HSM, and
- you apply a cloning domain to permit secure backup and restore, and secure cloning/replication of HSM objects to other HSMs.

For SafeNet Network HSM with Password Authentication, all authentication secrets, including the Security Officer authentication and the Cloning Domain secret are text strings that you type in at a keyboard (either via local serial console, or via SSH session).

From the `hsm init` command, the eventual outcome is an initialized HSM, that can be accessed by a specific SO password and Cloning Domain. How you get there can vary slightly, depending upon starting conditions. For this description, we assume a factory-fresh HSM. Alternatively, you can run `hsm factoryReset` (at the local serial console) to place the HSM in a similar "like new" state.

Initializing

- Issue the `hsm init` command, with a suitable HSM label - also include the HSM's SO password and a string for cloning domain (the domain determines with which other HSMs your HSM can clone objects).
- At this point, the HSM is initialized.
- Further actions are needed to prepare for use by your Clients, but you can now log in as SO/HSM Admin and perform HSM administrative actions.

Logging In, Once You Have Initialized

- To login, you issue the `hsm login` command at the `lunash:>` command line.
- When prompted, type the password. The HSM checks what it receives against what it expects. If it finds a mismatch, it records a bad-login attempt against the bad-login counter. You have two more chances to present the correct SO/HSM Admin authentication, or the SO is locked out and the HSM must be re-initialized (where it is zeroized and all contents are gone) before it can be used again.
- When you login successfully, the bad-login counter is reset to zero.

Soft Initialization

The above description covers the situation where your HSM is new from the factory, or where you have recently run `hsm factoryReset` command. The result of `hsm init` is different if the HSM is **not** in factory reset state.

If you run `hsm init -l <hsmlabel>` on an HSM that is currently in initialized state, then you are performing a "re-initialization", or a soft init, and not a full, hard initialization.

In this situation, `hsm init -label <hsmlabel>` means remove any partitions (and their contents) and erase any token objects that reside in SO space on the HSM. The SO identity is preserved, as is the cloning domain. The HSM label can be any string - you do not need to retain the previous label - you can change the previous label with this command. The password is required, to prove that you are entitled to perform the initialization.

Why choose Hard Init or Soft Init?

A good example of a situation where you might generally prefer to perform **soft initialization** is when provisioning with Crypto Command Center for virtual clients. When a client (virtual or otherwise) is done with a SafeNet HSM resource - say, a partition or a group of partitions - the resource must be cleared (removed and re-created, re-deployed) for the next customer.

Either kind of initialization operation takes care of destroying the partitions and contents, but a **soft init** leaves the SO identity and the cloning domain intact. The HSM remains within its established environment, under control of the Crypto Command Center administrator, who has no need to change SO and domain, but who does wish to create new user partitions for the next deployment.

A **hard initialization** (`factoryreset` followed by `init`) prepares the HSM for any environment, since the `factoryreset` removes any traces of the previous environment (SO and domain) and makes the HSM ready to accept new authentication.

The **hard init** is always the safest approach to take, since you can always choose to use the existing password and cloning-domain strings - emulating the end result of a soft init, but if you have security-policy reasons for not allowing the SO or domain to remain, the **hard init** addresses those reasons.

Similarly, if you had been validating an HSM in a laboratory environment before deploying it to a production HA environment, a **soft init** would leave the HSM with the lab domain in place. The domain used by your production HA group of HSMs would not match, thereby preventing the new HSM from cloning with that group (so no HA, no synchronization). A **hard init** of the new HSM when introducing it to the HA group would ensure that it was initialized with the domain needed to participate in that HA group.

Initialization Overview for PED-authenticated HSMs

For SafeNet HSMs, there are two kinds of initialization:

- "hard" init - occurs when the HSM is in a factory [re]fresh state
- "soft" init - occurs when the HSM is not in factory [re]fresh state

Both are launched by the same command, `hsm init -l <hsmlabel>`.

Condition/Effect	Soft init	Hard init
SO authentication required?	Yes	No
Can set new HSM label	Yes	Yes
Creates new SO identity	No	Yes
Creates new Domain	No	Yes
Destroys partitions	Yes	No (none exist to destroy) *
Destroys SO objects	Yes	No (none exist to destroy) *

* `hsm factoryReset` was performed, and destroyed partitions and objects, before the hard init... otherwise, it could not be a hard init.

Hard Initialization

Coming from the factory, the SafeNet HSM:

- has an undifferentiated HSM with no associations or ownership declared
- has not yet had virtual HSMs (HSM Partitions) created or assigned
- has not been introduced to the Clients (your Clients) with which it will be working.

Initialization takes care of the ownership by establishing roles and separations of authority.

When you initialize a new (or `factoryReset`) HSM, several things happen, but the most important ones from your operational perspective are:

- you set up Security Officer or HSM Administrator (two names for the same entity) ownership of the HSM, and
- you apply a cloning domain to permit secure backup and restore, and secure cloning/replication of HSM objects to other HSMs.

For SafeNet HSMs with PED (Trusted Path) Authentication, the HSM Security Officer authentication and the Cloning Domain secret are kept on portable physical memory devices called PED Keys (["About PED Keys" on page 186](#)). PED Keys can interact with the HSM via the SafeNet PED which provides a protected data path or "Trusted Path". Use of the Trusted Path for authentication prevents observation or interception of passwords such as you would type at a keyboard when seeking access to password authenticated HSMs.

From the `hsm init` command, the eventual outcome is an initialized HSM, that can be accessed by a specific blue SO PED Key and red Domain PED Key (or the Password authentication equivalents). How you get there can vary slightly, depending upon starting conditions. For this description, we assume a factory-fresh HSM and factory-fresh ('blank') PED Keys. Alternatively, you can run `hsm factoryReset` (at the local serial console) to place the HSM in a similar "like new" state.

Initializing

- Your SafeNet PED must be connected to the HSM, either locally/directly, or remotely via Remote PED connection (see ["About Remote PED" on page 270](#)), with "Awaiting command..." showing on its display.
- When you issue the `hsm init` command, the HSM passes control to the SafeNet PED, and the command line (`lunash:>`) directs you to attend to the PED prompts.
- A "default" login is preformed, just to get started (you don't need to supply any authentication for this step).
- SafeNet PED asks: "Do you wish to reuse an existing keyset?". If the answer is NO, the HSM creates a new secret which will reside on both the HSM and the key (or keys) that is (or are) about to be imprinted. If the answer is YES, then the HSM does not create a new secret and instead waits for one to be presented via the PED.
- SafeNet PED requests a blue PED Key. It could be blank to begin, or it could have a valid secret from another HSM (a secret that you wish to preserve), or it could have a secret that is no longer useful.
- SafeNet PED checks the key you provide. If the PED Key is not blank, and your answer to "...reuse an existing keyset" was "Yes", then SafeNet PED proceeds to copy the secret from the PED Key to the HSM.
- If the key is not blank, and your answer to "...reuse an existing keyset" was "No", then the PED inquires if you wish to overwrite its contents with a new HSM secret. If the current content of the key is of no value, you say "Yes". If the current content of the key is a valid secret from another HSM (or if you did not expect the key to hold any data) you can remove it from the PED and replace it with a blank key or a key containing non-useful data,

before you answer "Yes" to the 'overwrite' question. Even so, SafeNet PED asks "Are you sure...".

- Assuming that you are using a new secret, and not reusing an existing one, SafeNet PED asks if you wish to split the new HSM secret. It does this by asking for values of "M" and "N". You set those values to "1" and "1" respectively, unless you require MofN split-secret, multi-person access control for your HSM (See ["Using MofN" on page 243](#) for details).
- SafeNet PED asks if you wish to use a PED PIN (an additional secret; see ["What is a PED PIN?" on page 195](#) for more info).
- If you just press ENTER (effectively saying 'no' to the PED PIN option), then the secret generated by the HSM is imprinted on the PED Key, that same secret is retained as-is on the HSM, and the same secret becomes the piece needed to unlock the Security Officer/HSM Admin account on the HSM.
- If you press some digits on the PED keypad (saying 'yes' to the PED PIN option), then the PED combines the HSM-generated secret with your PED PIN and feeds the combined data blob to the HSM. The HSM throws away the original secret and takes on the new, combined secret as its SO/HSM Admin secret.
- The PED Key contains the original HSM-generated secret, but also contains the flag that tells the PED whether to demand a PED PIN (which is either no digits, or a set of digits that you supplied, and must supply at all future uses of that PED Key).
- SafeNet PED gives you the option to create some duplicates of this imprinted key. You should make at least one duplicate for backup purposes. Make additional duplicates if your security policy permits, and your procedures require them.
- Next, SafeNet PED requests a red Domain PED Key. The HSM provides a cloning Domain secret and the PED gives you the option to imprint the secret from the HSM, or to use a domain that might already be on the key. You choose appropriately. If you are imprinting a new Domain secret, you have the same opportunities to split the secret, and to apply a PED PIN "modifier" to the secret. Again, you are given the option to create duplicates of the key.
- At this point, the HSM is initialized and SafeNet PED passes control back to the appliance (lunash:>).
- Further actions are needed to prepare for use by your Clients, but you can now log in as SO/HSM Admin and perform HSM administrative actions.

Logging In, Once You Have Initialized

- The PED prompts for the blue SO PED Key. You insert that PED Key.
- If there was no PED PIN (you chose none at initialization time), then the PED combines the secret on the blue key with ... nothing... and the unchanged secret is passed to the HSM. The HSM recognizes the secret and logs you in. This assumes that you provided the correct blue PED Key.
- If there was a PED PIN (you added one at initialization time), then you type it on the PED keypad, SafeNet PED combines the secret from the key with the digits that you type, and the modified secret is passed to the HSM. The HSM recognizes the modified secret and logs you in. This assumes that you provided the correct blue PED Key **and** the correct PED PIN digits.
- If you type an incorrect PED PIN, what the HSM receives is much the same as if you presented a wrong PED Key. The HSM checks what it receives against what it expects, finds a mismatch and records a bad-login attempt against the bad-login counter. You have two more chances to present the correct SO authentication, or the SO is locked out and the HSM must be re-initialized (where it is zeroized and all contents are gone) before it can be used again.
- When you login successfully, the bad-login counter is reset to zero.

If you had also elected to split the login secret when you initialized, then the above sequence would need quantity M different blue keys from the set of quantity N, in order to reconstruct the needed secret (along with PED PINs, or not, for each partial-secret blue key).

Soft Initialization

The above description covers the situation where your HSM is new from the factory, or where you have recently run `hsm factoryReset` command. The result of `hsm init` is different if the HSM is **not** in factory reset state.

If you run `hsm init -l <hsmlabel>` on an HSM that is currently in initialized state, then you are performing a "re-initialization", or a soft init, and not a full, hard initialization.

In this situation, `hsm init -label <hsmlabel>` means remove any partitions (and their contents) and erase any token objects that reside in SO space on the HSM. The SO identity is preserved, as is the cloning domain. The HSM label can be any string - you do not need to retain the previous label - you can change the previous label with this command. You are prompted by SafeNet PED to insert the currently-valid blue SO PED Key (it is not changed by a soft init; it is needed only to validate your right to perform the soft initialization) and press [Enter] on the PED keypad. No other interaction is needed.

Why choose Hard Init or Soft Init?

A good example of a situation where you might generally prefer to perform **soft initialization** is when provisioning with Crypto Command Center for virtual clients. When a client (virtual or otherwise) is done with a SafeNet HSM resource - say, a partition or a group of partitions - the resource must be cleared (removed and re-created, re-deployed) for the next customer.

Either kind of initialization operation takes care of destroying the partitions and contents, but a **soft init** leaves the SO identity and the cloning domain intact. The HSM remains within its established environment, under control of the Crypto Command Center administrator, who has no need to change SO and domain, but who does wish to create new user partitions for the next deployment.

A **hard initialization** (factoryreset followed by init) prepares the HSM for any environment, since the factoryreset removes any traces of the previous environment (SO and domain) and makes the HSM ready to accept (or generate) new blue and red key data.

The **hard init** is always the safest approach to take, since you can always choose to use the existing blue and red PED Keys and imprint those onto the factoryreset HSM - emulating the end result of a soft init, but if you have security-policy reasons for not allowing the SO or domain to remain, the **hard init** addresses those reasons.

Similarly, if you had been validating an HSM in a laboratory environment before deploying it to a production HA environment, a **soft init** would leave the HSM with the lab domain in place (in a soft init, the PED does not prompt for insertion of the red PED Key, since the HSM already has a domain). The domain used by your production HA group of HSMs would not match, thereby preventing the new HSM from cloning with that group (so no HA, no synchronization). A **hard init** of the new HSM when introducing it to the HA group would ensure that it was initialized with the domain needed to participate in that HA group.

HSM Initialization and Zeroization

Ideally, the `hsm init` command is used once, when you first configure your SafeNet HSM for use with your application, then you place the unit in service and never initialize it again. However, unanticipated situations or requirements can arise that might cause you to initialize the HSM. A simple example is that you might perform trial setups in a laboratory environment before placing your SafeNet system into a "live" or "production" environment.

For further detail and for explanations of the concepts "hard" init and "soft" init, see ["Initialization Overview for PED-authenticated HSMs" on page 143](#) and ["Initialization Overview for Password-Authenticated HSMs" on page 141](#).

Additional Notes

The SafeNet shell command `hsm factoryReset` puts the HSM in a zeroized state. (See ["What Does Zeroized Mean?" on page 313](#).) To completely start over for configuration of the HSM, use `hsm factoryReset`, then `hsm init`.

It is not necessary to perform `hsm login` before `hsm factoryReset`. This is not considered a security issue because the command is accepted only via the local serial console. It is assumed that you provide sufficient physical security for your HSM appliance(s). An attacker who could interrupt or deny your use of the HSM by gaining access to your premises to make a serial connection and issue destructive commands could as easily steal or physically destroy the HSM while in your server room.

If you are taking a SafeNet Network HSM out of service, to go into storage, or to be shipped to another location (or back to SafeNet), then after you perform `hsm factoryReset`, perform `hsm init` to overwrite any labels or settings that you previously made.

View a table that compares and contrasts various "deny access" events or actions that are sometimes confused. ["Comparison of Destruction/Denial Actions" on page 309](#)

Re-initialize an HSM

To initialize (see ["Initializing a PED-Authenticated HSM" on page 1](#)) in the *Configuration Guide* or to re-initialize an HSM, use the command:

```
hsm init -label <new-HSM-label>
```



Note: Initializing/re-initializing an HSM destroys all HSM Partitions, and all contents are lost. This is not an action you would perform on a production SafeNet HSM. However, if you have made major changes in your system/deployment, or if you are moving a SafeNet HSM from a lab situation into production, you might wish to clear everything and restart with a "clean slate". In such cases, re-initialization might be appropriate. It would also be appropriate if you were so instructed by Customer Support.



Note: Also, some HSM policy changes are destructive of HSM contents (a security measure), and require re-initialization before you can continue to use the HSM. In the case where you intend to make a destructive HSM policy change, be sure to back up any important objects and keys so that they can be restored after the policy change and subsequent re-initialization.

Initialize an HSM With Existing Domain and Shared PED Keys

For two SafeNet HSMs, the following procedure assumes that you wish to have a set of PED Keys that will work with either HSM. One HSM is already initialized, so you have a full set of PED Keys, imprinted with the authentication data and the domain for that HSM. You want the second HSM to share the same domain (for backup, and the ability to

restore to either HSM from a Backup token), and both the old and the new PED Keys should work interchangeably with both HSMs.

For this example procedure, HSMs are designated:

- HSM 1, with its PED Keys Blue K1 and Red K1, and
 - HSM 2, with its PED Keys Blue K2 and Red K2.
1. Ensure that you can log in to HSM 1 as a Security Officer using Blue K1 (if not, then do not continue with the procedure).
 2. Log out.
 3. Begin initialization of HSM 2.
 4. Insert Blue K1 at the PED prompt, and when asked if you would "like to reuse an existing keyset", answer [YES] on the PED keypad.
 5. Duplicate Blue K1 to Blue K2 when prompted. (That is, when asked "Are you duplicating this keyset", answer [YES], then insert the target Blue K2).
 6. When "Generating a domain" appears, insert Red K1 at the prompt. When asked "Would you like to reuse an existing keyset", answer [YES].
 7. Duplicate Red K1 to Red K2.

The procedure to make a backup of the black PED Key (for HSM Partitions) would be similar to the procedure for the blue PED Key.



Note: You might receive a message that the key is blank, or that it contains valid data (for whatever type of key it was previously) and asking if you wish to overwrite. If the PED has indicated that the target PED Key is occupied and you are not certain that any authentication it contains is obsolete, then you should not allow it to be overwritten. Either remove the current, problematic key, insert another "blank" target key, and press [ENTER], or abort the operation. To abort, remove the PED Key and wait for PED time-out. Do NOT press [ENTER] at the "overwritten" message, if that is not your intent. Retry when you have sorted out your PED Keys and are confident that your target key is blank or contains truly obsolete authentication that can legitimately be overwritten.

If you wish to have a separate set of keys for each HSM, then instead of following the procedure as written you should use the Blue K2 and Black K2 for HSM2 and answer 'NO' to the question "Would you like to reuse an existing keyset?" This will imprint/overwrite the new blue or black keys making them specific to HSM2. For the Red key you should still insert Red K1 and answer 'YES' to the "Would you like to reuse an existing keyset?" question (the token/HSMs must share a common domain, or backup/restore cannot take place).

HSM Partitions

This chapter describes how to administer HSM administrative and application partitions on the HSM. It contains the following sections:

- ["HSM Partitions" below](#)
- ["Partition Creation - Notes" on page 1](#)
- ["Partition Creation with Policy Template Using LunaCM" on the next page](#)
- ["Partition Creation with Policy Template Using Lunash" on page 1](#)
- ["Separation of HSM Workspaces" on page 162](#)
- ["Configured and Registered Client Using an HSM Partition" on page 166](#)
- ["About Activation and Auto-Activation" on page 167](#)
- ["Removing Partitions" on page 170](#)
- ["Security of Your Partition Challenge" on page 170](#)
- ["Frequently Asked Questions" on page 172](#)

HSM Partitions

HSM Partitions are independent logical HSMs that reside within the SafeNet HSM inside, or attached to, your host computer or appliance. Each HSM Partition has its own data, access controls, security policies, and separate administration access for at least some roles, independent from other HSM partitions (if your HSM supports more than one). Depending on the product, the HSM can contain multiple HSM partitions, and each partition can be associated with one or more Clients. Each HSM Partition has a special administrative account or role, who manages it.

HSMs with firmware older than version 6.22.0 had three types of partitions:

- the HSM administrative partition, administered by the HSM SO
- the Auditor partition, accessible and administered by the HSM Auditor role, only
- application partition(s) administered at a high level by the HSM SO, but administered and operated at an operational level by the User or Crypto Officer role

HSMs with firmware 6.22.0 or newer can have three types of partitions:

- the HSM administrative partition, administered by the HSM SO (see Note below)
- legacy-style application partition(s) administered at a high level by the HSM SO, but administered and operated at an operational level by the User or Crypto Officer role (with optional Crypto User)

or

- PPSO application partitions (requires that the PPSO capability is installed) that are created by the HSM SO, but

are thereafter owned by their own local SOs, and administered and operated at an operational level by the Crypto Officer role (with optional Crypto User) - installing the PPSO capability is destructive, and requires that you re-create partitions, if you already had any.

Note: For HSMs with firmware 6.22.0 or newer, the Auditor role does not have an independent partition, but controls an area within the HSM administrative partition. The role and its objects are not seen or touched by the HSM SO.



Operationally, there is no difference from previous releases. The only caveat is that if you update an older HSM to firmware 6.22.0 or newer, the old Audit logging stops and you must initialize the Audit user again, and configure audit logging. It is perfectly acceptable to re-use the Auditor credentials (white PED Key).

HSM Partitions can be thought of as 'safe deposit boxes' that reside within the K6 Cryptographic Engine's 'vault'. The vault itself offers an extremely high level of security for all the contents inside; additionally, each safe deposit box also has its own security and access controls; while the bank managers might have access to the vault, they still cannot open the individual safe deposit boxes, because only the owner of the safe deposit box holds the key that opens it.

A legacy application partition was/is owned by the HSM SO, who assigns a User or Crypto Officer to handle day-to-day management of partition contents, creation, use, and destruction of keys and objects, and so on. PPSO application partitions (where HSM firmware is version 6.22.0 or newer, and the PPSO capability is applied) have their own partition SO, distinct from the HSM SO. The HSM SO initializes the HSM, sets HSM-wide policies, creates an empty application partition, and hands off complete control to whomever is to become the partition SO. Thereafter, the HSM has no oversight and can do nothing with the partition except to delete it, if that is ever required. The Partition SO then initializes the partition creating a Crypto Officer

Note: If you are both

- upgrading from an earlier firmware version to HSM firmware 6.22.0 (or newer)

AND

- applying the Per-Partition SO (PPSO) capability update,

be aware that the PPSO capability update is destructive. Therefore, there is no need to re-size partitions.



Instead, to avoid unnecessary duplication of effort, you should

- safeguard (archive) any existing partition contents,
 - then zeroize the HSM for a clean update,
 - then perform both the firmware AND capability updates,
 - and finally restore to new partitions.
-

Partition Creation with Policy Template Using LunaCM

Partition Policy Templates enable administrators to replicate configured application partitions, speeding the provisioning process and ensuring consistent policy assignments across partitions with similar security requirements. The Partition Policy Template feature enables scalable policy management across tens and hundreds of partitions while also simplifying future audit and compliance requirements.

Administrators can specify the initial value for each policy, as well as whether changes to the policy AFTER the partition is created will be destructive to existing user objects on the partition. This destructive or non-destructive behavior can be specified independently for the on-to-off and off-to-on transitions of the policy. Once the combined

initial values and destructiveness of each partition policy are configured as desired, they can be saved as a named policy template. Multiple such policy templates can be saved on the appliance, or *exported and imported between appliances*.

An administrator creating an application partition can optionally specify a previously saved policy template in order to create the partition with policy settings as configured in the template. If no policy template is specified during partition creation, the HSM uses built-in default partition policy values.

Partition policy templates can *not* be used to alter settings for an existing application partition. Once a partition has been created, with or without the use of policy templates, the administrator continues to use the **partition changePolicy** command to make changes to individual policy values.



Note: Policy destructiveness settings can *not* be altered on an existing application partition, as these can be specified only at the time the partition is created.

The examples on this page apply to manipulating application partitions via lunacm.

Process for a New Template

The general procedure is as follows:

- Create (and load for editing) a new, unnamed partition policy template. The possible policy codes, along with their default settings, are displayed.
- Make changes to those default values, one at a time, until you are satisfied. Each change is echoed back.
- Save the new partition policy template, applying a name that is unique and easily recognized, and also applying additional descriptive text to assist yourself and future users to recall the purpose of this specific template among any others you might create.
- Create an application partition, specifying a particular partition policy template by name. This creates the partition with policies applied to it, conforming to the selected template, different from the default set for the HSM.

Create and apply a new partition policy template

For this example, before starting, here are the policy values for a default partition that was created without using a template:

```
lunacm:> partition showpolicies
Partition Capabilities
  0: Enable private key cloning : 1
  1: Enable private key wrapping : 0
  2: Enable private key unwrapping : 1
  3: Enable private key masking : 0
  4: Enable secret key cloning : 1
  5: Enable secret key wrapping : 1
  6: Enable secret key unwrapping : 1
  7: Enable secret key masking : 0
 10: Enable multipurpose keys : 1
 11: Enable changing key attributes : 1
 15: Allow failed challenge responses : 1
 16: Enable operation without RSA blinding : 1
 17: Enable signing with non-local keys : 1
 18: Enable raw RSA operations : 1
 20: Max failed user logins allowed : 10
 21: Enable high availability recovery : 1
```

```
22: Enable activation : 1
23: Enable auto-activation : 1
25: Minimum pin length (inverted: 255 - min) : 248
26: Maximum pin length : 255
28: Enable Key Management Functions : 1
29: Enable RSA signing without confirmation : 1
30: Enable Remote Authentication : 1
31: Enable private key unmasking : 1
32: Enable secret key unmasking : 1
33: Enable RSA PKCS mechanism : 1
34: Enable CBC-PAD (un)wrap keys of any size : 1
35: Enable private key SFF backup/restore : 1
36: Enable secret key SFF backup/restore : 1
37: Enable Secure Trusted Channel : 1
```

Partition Policies

```
0: Allow private key cloning : 1
1: Allow private key wrapping : 0
2: Allow private key unwrapping : 1
3: Allow private key masking : 0
4: Allow secret key cloning : 1
5: Allow secret key wrapping : 1
6: Allow secret key unwrapping : 1
7: Allow secret key masking : 0
10: Allow multipurpose keys : 1
11: Allow changing key attributes : 1
15: Ignore failed challenge responses : 1
16: Operate without RSA blinding : 1
17: Allow signing with non-local keys : 1
18: Allow raw RSA operations : 1
20: Max failed user logins allowed : 10
21: Allow high availability recovery : 1
22: Allow activation : 0
23: Allow auto-activation : 0
25: Minimum pin length (inverted: 255 - min) : 248
26: Maximum pin length : 255
28: Allow Key Management Functions : 1
29: Perform RSA signing without confirmation : 1
30: Allow Remote Authentication : 1
31: Allow private key unmasking : 1
32: Allow secret key unmasking : 1
33: Allow RSA PKCS mechanism : 1
34: Allow CBC-PAD (un)wrap keys of any size : 1
35: Allow private key SFF backup/restore : 1
36: Allow secret key SFF backup/restore : 1
37: Force Secure Trusted Channel : 0
```

Command Result : No Error

Now, create a partition policy template and then create a new application partition using the new template.



Note: You must be in the administrative (HSM SO) slot in order to create a partition policy template.

1. Use command **partition policyTemplateCreate** to create a new partition policy template:

```
lunacm:> partition policytemplatecreate
```

Code	Description	Value	Destructive	
			Off-To-On	On-To-Off
0	Allow private key cloning	On	Yes	No
1	Allow private key wrapping	Off	Yes	No
2	Allow private key unwrapping	On	No	No
3	Allow private key masking	Off	Yes	No
4	Allow secret key cloning	On	Yes	No
5	Allow secret key wrapping	On	Yes	No
6	Allow secret key unwrapping	On	No	No
7	Allow secret key masking	Off	Yes	No
10	Allow multipurpose keys	On	Yes	No
11	Allow changing key attributes	On	Yes	No
15	Ignore failed challenge responses	On	Yes	No
16	Operate without RSA blinding	On	Yes	No
17	Allow signing with non-local keys	On	No	No
18	Allow raw RSA operations	On	Yes	No
20	Max failed user logins allowed	10	N/A	N/A
21	Allow high availability recovery	On	No	No
22	Allow activation	On	No	No
23	Allow auto-activation	On	No	No
24	Allow indirect login	Off	No	No
25	Minimum pin length (inverted: 255 - min)	248	N/A	N/A
26	Maximum pin length	255	N/A	N/A
28	Allow Key Management Functions	On	Yes	No
29	Perform RSA signing without confirmation	On	Yes	No
30	Allow Remote Authentication	On	No	No
31	Allow private key unmasking	On	No	No
32	Allow secret key unmasking	On	No	No
33	Allow RSA PKCS mechanism	On	Yes	No
34	Allow CBC-PAD (un)wrap keys of any size	On	Yes	No
35	Allow private key SFF backup/restore	Off	Yes	No
36	Allow secret key SFF backup/restore	Off	Yes	No
37	Force Secure Trusted Channel	Off	No	Yes

```
Type 'proceed' to continue, or 'quit'
to quit now.
> proceed
```

Successfully created and loaded the new partition policy template.

Use 'partition policyTemplateChange' to edit the template and 'partition policyTemplateSave' to save the template once you have applied all necessary changes.

Command Result : No Error

2. Use command **partition policyTemplateChange** to change some policy values in the new partition policy template:

```
lunacm:> partition policyTemplateChange -policy 25 -value 246
```

Code	Description	Value	Destructive	
			Off-To-On	On-To-Off
25	Minimum pin length (inverted: 255 - min)	246	N/A	N/A

Command Result : No Error

```
lunacm:> partition policyTemplateChange -policy 20 -value 9
```

Code	Description	Value	Destructive	
			Off-To-On	On-To-Off
20	Max failed user logins allowed	9	N/A	N/A

Command Result : No Error

```
lunacm:> partition policyTemplateChange -policy 7 -on non-destructive
```

Code	Description	Value	Destructive	
			Off-To-On	On-To-Off
7	Allow secret key masking	Off	No	No

Command Result : No Error

3. Use command **partition policyTemplateSave** to save the new partition policy template with its modified policy values:

```
lunacm:> partition policyTemplateSave -name sample01
```

```
sample01 successfully saved.
```

Command Result : No Error

```
lunacm:> partition policyTemplateList
```

Name	Description
sample01	

sample01

No partition policy template is currently loaded.

Command Result : No Error

4. Use command **partition create** with the **-policytemplate** option to create a new application partition, using the partition policy template that you previously created:

```
lunacm:> partition create -label parfortemplate -policyTemplate sample01
```

```
    Please attend to the PED.
```

```
Command Result : No Error
```

```
lunacm:> slot set slot 0
```

```
    Current Slot Id:    0      (Luna User Slot 6.24.0 (PED) Signing With Cloning Mode)
```

```
Command Result : No Error
```

```
lunacm:> partition showpolicies
```

```
    Partition Capabilities
```

```
    0: Enable private key cloning : 1
    1: Enable private key wrapping : 0
    2: Enable private key unwrapping : 1
    3: Enable private key masking : 0
    4: Enable secret key cloning : 1
    5: Enable secret key wrapping : 1
    6: Enable secret key unwrapping : 1
    7: Enable secret key masking : 0
    10: Enable multipurpose keys : 1
    11: Enable changing key attributes : 1
    15: Allow failed challenge responses : 1
    16: Enable operation without RSA blinding : 1
    17: Enable signing with non-local keys : 1
    18: Enable raw RSA operations : 1
    20: Max failed user logins allowed : 10
    21: Enable high availability recovery : 1
    22: Enable activation : 1
    23: Enable auto-activation : 1
    25: Minimum pin length (inverted: 255 - min) : 248
    26: Maximum pin length : 255
    28: Enable Key Management Functions : 1
    29: Enable RSA signing without confirmation : 1
    30: Enable Remote Authentication : 1
    31: Enable private key unmasking : 1
    32: Enable secret key unmasking : 1
    33: Enable RSA PKCS mechanism : 1
    34: Enable CBC-PAD (un)wrap keys of any size : 1
    35: Enable private key SFF backup/restore : 1
    36: Enable secret key SFF backup/restore : 1
    37: Enable Secure Trusted Channel : 1
```

```
    Partition Policies
```

```
    0: Allow private key cloning : 1
    1: Allow private key wrapping : 0
    2: Allow private key unwrapping : 1
    3: Allow private key masking : 0
    4: Allow secret key cloning : 1
    5: Allow secret key wrapping : 1
    6: Allow secret key unwrapping : 1
    7: Allow secret key masking : 0
    10: Allow multipurpose keys : 1
    11: Allow changing key attributes : 1
    15: Ignore failed challenge responses : 1
```

```

16: Operate without RSA blinding : 1
17: Allow signing with non-local keys : 1
18: Allow raw RSA operations : 1
20: Max failed user logins allowed : 9
21: Allow high availability recovery : 1
22: Allow activation : 0
23: Allow auto-activation : 0
25: Minimum pin length (inverted: 255 - min) : 246
26: Maximum pin length : 255
28: Allow Key Management Functions : 1
29: Perform RSA signing without confirmation : 1
30: Allow Remote Authentication : 1
31: Allow private key unmasking : 1
32: Allow secret key unmasking : 1
33: Allow RSA PKCS mechanism : 1
34: Allow CBC-PAD (un)wrap keys of any size : 1
35: Allow private key SFF backup/restore : 1
36: Allow secret key SFF backup/restore : 1
37: Force Secure Trusted Channel : 0

```

Command Result : No Error

Modify a partition template, then apply the modified partition template

For this example, we create an application using a partition template that has only one policy modified, then change the template to modify an additional policy, and create yet another partition to which we apply the modified partition template:



Note: You must be in the administrative (HSM SO) slot in order to create, load, and modify a partition policy template.

1. Create and save partition policy template Sample02 with policy 22 set to On, but policy 23 not set (see previous example for steps).
2. Use command **partition create** with the **-policytemplate** option to create a new application partition, using partition policy template Sample02 previously created:

```
lunacm:> partition create -label parfortemplateagain -policyTemplate Sample02
```

Please attend to the PED.

Command Result : No Error

3. Change to the slot of the newly-created partition and use command **partition showpolicies** to show the policies of the new partition:

```
lunacm:> slot set slot 0
```

Current Slot Id: 0 (Luna User Slot 6.24.0 (PED) Signing With Cloning Mode)

Command Result : No Error

```
lunacm:> partition showpolicies
```

Partition Capabilities

```
0: Enable private key cloning : 1
1: Enable private key wrapping : 0
2: Enable private key unwrapping : 1
3: Enable private key masking : 0
4: Enable secret key cloning : 1
5: Enable secret key wrapping : 1
6: Enable secret key unwrapping : 1
7: Enable secret key masking : 0
10: Enable multipurpose keys : 1
11: Enable changing key attributes : 1
15: Allow failed challenge responses : 1
16: Enable operation without RSA blinding : 1
17: Enable signing with non-local keys : 1
18: Enable raw RSA operations : 1
20: Max failed user logins allowed : 10
21: Enable high availability recovery : 1
22: Enable activation : 1
23: Enable auto-activation : 1
25: Minimum pin length (inverted: 255 - min) : 248
26: Maximum pin length : 255
28: Enable Key Management Functions : 1
29: Enable RSA signing without confirmation : 1
30: Enable Remote Authentication : 1
31: Enable private key unmasking : 1
32: Enable secret key unmasking : 1
33: Enable RSA PKCS mechanism : 1
34: Enable CBC-PAD (un)wrap keys of any size : 1
35: Enable private key SFF backup/restore : 1
36: Enable secret key SFF backup/restore : 1
37: Enable Secure Trusted Channel : 1
```

Partition Policies

```
0: Allow private key cloning : 1
1: Allow private key wrapping : 0
2: Allow private key unwrapping : 1
3: Allow private key masking : 0
4: Allow secret key cloning : 1
5: Allow secret key wrapping : 1
6: Allow secret key unwrapping : 1
7: Allow secret key masking : 0
10: Allow multipurpose keys : 1
11: Allow changing key attributes : 1
15: Ignore failed challenge responses : 1
16: Operate without RSA blinding : 1
17: Allow signing with non-local keys : 1
18: Allow raw RSA operations : 1
20: Max failed user logins allowed : 10
21: Allow high availability recovery : 1
22: Allow activation : 1
23: Allow auto-activation : 0
25: Minimum pin length (inverted: 255 - min) : 248
26: Maximum pin length : 255
28: Allow Key Management Functions : 1
29: Perform RSA signing without confirmation : 1
30: Allow Remote Authentication : 1
31: Allow private key unmasking : 1
```

```

32: Allow secret key unmasking : 1
33: Allow RSA PKCS mechanism : 1
34: Allow CBC-PAD (un)wrap keys of any size : 1
35: Allow private key SFF backup/restore : 1
36: Allow secret key SFF backup/restore : 1
37: Force Secure Trusted Channel : 0

```

Command Result : No Error

Observe that policy 22 is on; policy 23 is off, the result of creating the partition with partition policy template Sample02 as it exists at the moment.

- Use command **partition policyTemplateList** to show the available partition policy templates:

```
partition policyTemplate list
```

Name	Description
Sample02	Another template
sample01	Sample partition policyTemplate

No partition policy template is currently loaded.

Command Result : No Error

- Go back to the administrative slot if necessary, and use command **partition policyTemplateLoad** to load template Sample02 for modification:

```
lunacm:> partition policyTemplateLoad -name Sample02
```

Successfully loaded Sample02 partition policy template for editing.

Command Result : No Error

- Use command **partition policyTemplateChange** to change policy 23 in the loaded (for editing) partition policy template:

```
lunacm:> partition policyTemplateChange -policy 23 -value on
```

Code	Description	Destructive		
		Value	Off-To-On	On-To-Off
23	Allow auto-activation	On	No	No

Command Result : No Error

Observe that we can use the text string "On" or "Off" interchangeably with the numeric setting "1" or "0" to set a policy; both options are acceptable.

- Use command **partition policyTemplateSave** to save the newly modified partition policy template with its modified policy value. Do not provide a name; the loaded policy already has one (in this case, "Sample02"):

```
lunacm:> partition policyTemplateSave
```

Saving the modified settings will overwrite the existing template "Sample02".

Type 'proceed' to continue, or 'quit' to quit now -> proceed

Sample02 successfully saved.

Command Result : No Error

8. Delete the previously-created demonstration partition, if necessary to make room. Use command **partition create** with the **-policytemplate** option to create another new application partition, using partition policy template Sample02 previously created, and just now modified:

```
lunacm:> partition create -label parfortemplateyetagain -policyTemplate Sample02
```

Please attend to the PED.

Command Result : No Error

9. Use command **partition showpolicies** to show the policies of the new partition:

```
lunacm:> slot set slot 0
```

Current Slot Id: 0 (Luna User Slot 6.24.0 (PED) Signing With Cloning Mode)

Command Result : No Error

```
lunacm:> partition showpolicies
```

Partition Capabilities

```

0: Enable private key cloning : 1
1: Enable private key wrapping : 0
2: Enable private key unwrapping : 1
3: Enable private key masking : 0
4: Enable secret key cloning : 1
5: Enable secret key wrapping : 1
6: Enable secret key unwrapping : 1
7: Enable secret key masking : 0
10: Enable multipurpose keys : 1
11: Enable changing key attributes : 1
15: Allow failed challenge responses : 1
16: Enable operation without RSA blinding : 1
17: Enable signing with non-local keys : 1
18: Enable raw RSA operations : 1
20: Max failed user logins allowed : 10
21: Enable high availability recovery : 1
22: Enable activation : 1
23: Enable auto-activation : 1
25: Minimum pin length (inverted: 255 - min) : 248
26: Maximum pin length : 255
28: Enable Key Management Functions : 1
29: Enable RSA signing without confirmation : 1
30: Enable Remote Authentication : 1
31: Enable private key unmasking : 1
32: Enable secret key unmasking : 1
33: Enable RSA PKCS mechanism : 1
34: Enable CBC-PAD (un)wrap keys of any size : 1
35: Enable private key SFF backup/restore : 1

```

```
36: Enable secret key SFF backup/restore : 1
37: Enable Secure Trusted Channel : 1
```

Partition Policies

```
0: Allow private key cloning : 1
1: Allow private key wrapping : 0
2: Allow private key unwrapping : 1
3: Allow private key masking : 0
4: Allow secret key cloning : 1
5: Allow secret key wrapping : 1
6: Allow secret key unwrapping : 1
7: Allow secret key masking : 0
10: Allow multipurpose keys : 1
11: Allow changing key attributes : 1
15: Ignore failed challenge responses : 1
16: Operate without RSA blinding : 1
17: Allow signing with non-local keys : 1
18: Allow raw RSA operations : 1
20: Max failed user logins allowed : 10
21: Allow high availability recovery : 1
22: Allow activation : 1
23: Allow auto-activation : 1
25: Minimum pin length (inverted: 255 - min) : 248
26: Maximum pin length : 255
28: Allow Key Management Functions : 1
29: Perform RSA signing without confirmation : 1
30: Allow Remote Authentication : 1
31: Allow private key unmasking : 1
32: Allow secret key unmasking : 1
33: Allow RSA PKCS mechanism : 1
34: Allow CBC-PAD (un)wrap keys of any size : 1
35: Allow private key SFF backup/restore : 1
36: Allow secret key SFF backup/restore : 1
37: Force Secure Trusted Channel : 0
```

Command Result : No Error

Observe that both policy 22 and policy 23 are on (value = 1), as soon as the partition `parfortemplateyetagain` is created, using the recently-modified partition policy template "Sample02". For more information about those frequently-used policies, see ["About Activation and Auto-Activation" on page 167](#).

Note: The chosen partition affects the policies of a partition only when a partition is created.



In the examples on this page, partition `parfortemplateagain` was created when policy template `Sample02` was set to modify only partition policy 22. Therefore, partition `parfortemplateagain` does not have partition policy 23 set. The change to the policy template does not affect a partition that was already in existence. It has effect only for partitions that are created with that template after the template was modified.

Partition `parfortemplateyetagain` was created with the template after that modification, so it shows both policies changed.

You can change a policy manually, using **partition changepolicy** command.

Delete a partition policy template

If a partition policy template is no longer useful, use command **partition policyTemplate delete** to remove that template from the list.



Note: You must be in the administrative (HSM SO) slot in order to delete a partition policy template.

```
lunacm:> slot list
```

```

Slot Id -> 0
Tunnel Slot Id -> 2
Label ->
Serial Number -> 349297122742
Model -> K6 Base
Firmware Version -> 6.24.0
Configuration -> Luna User Partition With SO (PED) Signing With Cloning
Mode
Slot Description -> User Token Slot

Slot Id -> 1
Tunnel Slot Id -> 2
Label -> mypcie6
Serial Number -> 150022
Model -> K6 Base
Firmware Version -> 6.24.0
Configuration -> Luna HSM Admin Partition (PED) Signing With Cloning Mode
Slot Description -> Admin Token Slot
HSM Configuration -> Luna HSM Admin Partition (PED)
HSM Status -> OK

```

```
Current Slot Id: 1
```

```
Command Result : No Error
```

```
lunacm:> slot set slot 1
```

```
Current Slot Id: 1 (Luna Admin Slot 6.24.0 (PED) Signing With Cloning Mode)
```

Command Result : No Error

```
lunacm:> partition policyTemplateList
```

Name	Description
Sample02	Another template
sample01	

No partition policy template is currently loaded.

Command Result : No Error

```
lunacm:> partition policyTemplateDelete -name sample01
```

Are you sure you wish to delete partition policy template: sample01

Type 'proceed' to continue, or 'quit' to quit now -> proceed

Successfully deleted partition policy template: sample01

Command Result : No Error

```
lunacm:> slot set slot 1
```

Current Slot Id: 1 (Luna Admin Slot 6.24.0 (PED) Signing With Cloning Mode)

Command Result : No Error

```
lunacm:> partition policyTemplateList
```

Name	Description
Sample02	Another template

No partition policy template is currently loaded.

Command Result : No Error

Separation of HSM Workspaces

Depending on the SafeNet HSM and its configuration, the HSM can have three, or more, logical partitions.

- One for the Security Officer (SO)
- One for the Auditor, and
- One (or more) for applications and Clients.

In rare circumstance, the Security Officer might create and keep cryptographic objects, Normally it is not used for "production" cryptographic operations - the SO space is intended for overall HSM-level administration.

The Auditor partition is used to enable and manage secure audit logging activities, and generally has no other function in the HSM.

Legacy Application Partitions

The application partition (or partitions, depending upon HSM type and configuration) is enabled (Activated) and managed by the partition User Owner in some regimes), and is then used by client applications to create and use cryptographic objects, and to perform cryptographic operations.

The ordinary partition User entity can be further sub-divided into Crypto Officer and Crypto-User in cryptographic security regimes that require this distinction. Legacy partitions are under the administrative control of the HSM SO, and do not have their own separate SO. The User or the Crypto Officer entity is created by the HSM SO.

PPSO Application Partitions

Either type (legacy or PPSO) can be created on an HSM with firmware 6.22.0 or newer and with the PPSO capability update installed. On HSMs that support multiple application partitions it is possible to create both types on the same HSM. A PPSO partition has its own SO. The Partition SO manages what happens inside its partition. The HSM SO creates the PPSO partition, and deletes it when necessary, but has no other oversight or control of the PPSO partition. This distinction is particularly important in cloud scenarios, but is a significant element in separation of roles for any use of an HSM.

Operation

Crypto operations are normally performed from a logged-in session on the HSM. It is possible to create objects without logging in, so long as the CKA_PRIVATE attribute is set to 0 - that is, public objects. You can also delete any object that has CKA_PRIVATE=0. This is as defined in PKCS#11, and is not a security issue.

The restrictions that you expect come into play for objects that are created with CKA_PRIVATE=1, where only the owner is able to delete (or the SO could delete the entire partition containing the objects).

These distinctions can be demonstrated with CKDEMO commands 1) Open Session, and 3) Login.

The "Open Session" prompt has three options, to choose the partition that you wish to use:

Enter your choice (99 or 'FULL' for full help): 1

SO[0], normal user[1], or audit user[2]?

If you select "normal user [1]", when opening a session, you are telling the library that you choose to use the user partition which is owned by the partition User (or is shared by the Crypto-Officer and Crypto-User if the partition User has been separated into those two sub-entities).

The session is started, but you have not yet authenticated, and so cannot perform most operations in the session.

The Login prompt has four options, to perform the needed authentication (log into the session that you started above):

Enter your choice (99 or 'FULL' for full help): 3

Security Officer[0]

Crypto-Officer [1]

Crypto-User [2]:

Audit-User [3]:

Enter PIN :

If you have chosen the "normal user [1]" partition, when opening the session, then the valid login authentication options are:

- Crypto-Officer (which is the same as partition User (the black PED Key for PED-authenticated HSMs) if the Crypto-Officer/Crypto-User distinction is not in force) or
- Crypto User (which is the limited user when the Crypto-Officer/Crypto-User distinction has been invoked).

If you attempt one of the other two authentications, "Security Officer [0]" or "Audit-User [3]", an error message is returned because those are not applicable to the session type (therefore, the partition type) that you selected earlier.

If certificates are created as private objects (CKA_PRIVATE=1), the Crypto User cannot delete them. Also, the Crypto User cannot create fake private objects with CKA_PRIVATE=1. The Crypto User limitations are focused on restricting access to sensitive and/or private keys and objects.

Key Management Commands

LUNA_CREATE_OBJECT:

LUNA_COPY_OBJECT:

LUNA_DESTROY_OBJECT:

LUNA_MODIFY_OBJECT:

LUNA_DESTROY_MULTIPLE_OBJECTS:

LUNA_GENERATE_KEY:

LUNA_GENERATE_KEY_W_VALUE:

LUNA_GENERATE_KEY_PAIR:

LUNA_WRAP_KEY:

LUNA_UNWRAP_KEY:

LUNA_UNWRAP_KEY_W_VALUE:

LUNA_DERIVE_KEY:

LUNA_DERIVE_KEY_W_VALUE:

LUNA_MODIFY_USAGE_COUNT:

Normal Usage Commands

LUNA_ENCRYPT_INIT:

LUNA_ENCRYPT:

LUNA_ENCRYPT_END:

LUNA_ENCRYPT_SINGLEPART:

LUNA_DECRYPT_INIT:

LUNA_DECRYPT:

LUNA_DECRYPT_END:

LUNA_DECRYPT_RAW_RSA:
LUNA_DECRYPT_SINGLEPART:
LUNA_DIGEST_INIT:
LUNA_DIGEST:
LUNA_DIGEST_KEY:
LUNA_DIGEST_END:
LUNA_SIGN_INIT:
LUNA_SIGN:
LUNA_SIGN_END:
LUNA_SIGN_SINGLEPART:
LUNA_VERIFY_INIT:
LUNA_VERIFY:
LUNA_VERIFY_END:
LUNA_VERIFY_SINGLEPART:
LUNA_GET_OBJECT_SIZE:
LUNA_SEED_RANDOM:

Unauthenticated Commands

LUNA_GET:
LUNA_GET_CONTAINER_LIST:
LUNA_GET_CONTAINER_NAME:
LUNA_LOGIN:
LUNA_OPEN_SESSION:
LUNA_PARTITION_SERNUM_GET:
LUNA_FIND_OBJECTS:
LUNA_GET_RANDOM:
LUNA_OPEN_ACCESS:
LUNA_GET_MECH_LIST:
LUNA_GET_MECH_INFO:
LUNA_SELF_TEST:
LUNA_GET_HSM_CAPABILITY_SET:
LUNA_GET_HSM_POLICY_SET:
LUNA_GET_CONTAINER_CAPABILITY_SET:
LUNA_GET_CONTAINER_POLICY_SET:
LUNA_GET_CONFIGURATION_ELEMENT_DESCRIPTION:
LUNA_RETRIEVE_LICENSE_LIST:

LUNA_QUERY_LICENSE:
LUNA_GET_CONTAINER_STATUS:
LUNA_GET_OUID:
LUNA_GET_CONTAINER_STORAGE_INFO:
LUNA_GET_ATTRIBUTE_VALUE:
LUNA_GET_ATTRIBUTE_SIZE:
LUNA_GET_HANDLE:
LUNA_INIT_TOKEN:
LUNA_PARTITION_INIT:
LUNA_CLOSE_ACCESS:
LUNA_DEACTIVATE:
LUNA_MTK_GET_STATE:
LUNA_MTK_RESPLIT:
LUNA_MTK_RESTORE:
LUNA_MTK_UNLOCK_CHALLENGE:
LUNA_MTK_UNLOCK_RESPONSE:
LUNA_MTK_ZEROIZE:
LUNA_CLEAN_ACCESS:
LUNA_PED_GET_SET_RAW_DATA:
LUNA_ZEROIZE:
LUNA_FACTORY_RESET:
LUNA_HA_LOGIN:
LUNA_CONFIGURE_SP:
LUNA_LOG_POLL_HOST:
LUNA_LOG_EXTERNAL:
LUNA_ROLE_STATE_GET:

Commands That are Valid Only in a Session, But Require Special Handling

LUNA_LOGOUT:
LUNA_CLOSE_ALL_SESSIONS:
LUNA_CLOSE_SESSION:
LUNA_GET_SESSION_INFO:

Configured and Registered Client Using an HSM Partition

Following the instructions in the previous sections, you have already:

- registered and assigned a Client to a SafeNet Network HSM Partition.

All that is required for a Client application to begin using a SafeNet Network HSM Partition (to which the Client has been assigned) is the standard handshake sequence:

- the **client establishes** a Network Trust Link connection with the SafeNet Network HSM (port 1792)
- the **client requests** a list of available Partitions (if not already known)
- **SafeNet Network HSM responds** with a list of only those Partitions to which the requesting Client has been assigned
- the **client chooses** a Partition from the available, assigned Partitions
- **SafeNet Network HSM demands** the password for the selected Partition
- the **Client** (which may also be called Crypto User if you are using the Crypto Officer / Crypto User authentication and access model) **provides** the appropriate password
- **SafeNet Network HSM grants** access, and the **Client application begins using** the Partition.

Your application should be capable of performing the above actions.

Simple Troubleshooting

If your Client application is having difficulty using SafeNet Network HSM for Client tasks, and if you have already verified the connection and the configuration (using multitoken and CMU utilities - see ["The Multitoken Utility"](#) or see ["About CMU Functions"](#)), then there may be a problem with the configuration of your Client application. Try the following suggestions before calling for support.

If your SafeNet Network HSM is a Password Authentication model, then you should look to your application setup for the source of the problem. It might require special configuration to use a Hardware Security Module (HSM). Or, if SafeNet Network HSM has replaced another HSM product (including a SafeNet product) you will need to modify the application to recognize the new device.



Note: Refer to the *SDK Reference Guide* and to the application integration documents provided by SafeNet Technical Support for information and advice on integrating many popular applications and services with SafeNet Network HSM.

However, if your SafeNet Network HSM is a PED Authenticated model, then be aware that having the Client application present the Partition Password is not sufficient to access the HSM Partition. The HSM Partition must also be in a special login state called activation (see), meaning that the Partition Owner (or Crypto Officer) must have logged in (with the correct black Partition Owner (or Crypto Officer) PED Key), and not logged out again before your application tried to connect. To ensure that the HSM Partition is always in the desired state, we recommend that you autoActivate (see ["About Activation and Auto-Activation" below](#)) the Partition, so that it can accept Client authentication and access at any time without human intervention at the SafeNet Network HSM appliance.

If you wish minute-by-minute control of a client's ability to access the HSM, without need for human presence at the appliance location, you could use the Remote PED feature (see ["About Remote PED" on page 270](#)).

About Activation and Auto-Activation

Client access to Partitions, on an HSM with PED Authentication, needs to be as efficient and convenient as Client access to a Password Authenticated HSM . Activation and auto-Activation are ways to manage the additional layer of authentication - the PED and PED Keys - so that Clients can reliably connect using just their passwords.

Activation

Activation is just a login with explicit caching of the login data, on the HSM.

- For legacy partitions, the cached authentication data is referred to as partition login data, handled by partition commands.
- For PPSO partitions, the cached authentication data is referred to as role login data, handled by role commands.

Login caching, or Activation, is convenient so that you can remove the black or gray PED key (perhaps to allow other uses of the PED, such as administrative logins by the HSM SO, or moving the PED to another HSM), while ensuring that access by Clients is not stopped, and that nobody is required to be present to press [ENTER] on the keypad on behalf of Clients.

To use Activation, you must first allow it by setting Partition Policy 22 (Allow Activation) to *on*, for each partition that you create. This is done by the HSM SO for legacy application partitions, and by the Partition SO for PPSO application partitions. If the Policy (22, Allow Activation) is *on*, then the partition Crypto Officer) can issue the `partition activate` command for legacy partitions. For PPSO partitions, once the policy is active it requires just **role login** to activate. The PED prompts for the black PED Key(s) and PED PIN if appropriate. Once you provide a black PED Key (Crypto Officer) or gray PED Key (Crypto User), the HSM appliance caches that authentication and the partition remains in a login state (Activated) until:

- you explicitly deactivate (with lunash command `partition deactivate`, or lunacm command **partition deactivate** or **role deactivate**, as appropriate)
- power is lost to the HSM.

You can remove the black PED Key (or gray PED Key) and keep it in your pocket or in safe storage. Activation remains on, and any registered Client with the Partition challenge password is able to connect and perform operations on the partition.

Activation is not a big advantage for Clients that connect and remain connected. It is an indispensable advantage in cases where Clients repeatedly connect to perform a task and then disconnect or close the cryptographic session following completion of each task.

To activate an application partition for use by registered Clients

1. Ensure that the partition policy "Allow activation" has been switched on.

For SafeNet PCIe HSM or SafeNet USB HSM legacy application partition, type:

partition changepolicy -policy 22 -value 1

For SafeNet PCIe HSM or SafeNet USB HSM or SafeNet Network HSM PPSO application partition, type:

partition changepolicy -slot <slot number> -policy 22 -value 1

2. To start activation of the desired partition, type:

partition activate -par <partitionname>

for legacy application partitions, or type:

role login -name <name of role to log in>

for PPSO application partitions.

Respond to the PED prompts.

AutoActivation

AutoActivation is supported for SafeNet Network HSM and for SafeNet PCIe HSM, but not for SafeNet USB HSM.

AutoActivation extends the Activation feature, and allows automatic re-activation of the partition or the role, using the cached Crypto Officer or Crypto User authentication data, in the event of a restart or a short power outage (up to 2 hours). That is, the Activated state can recover to allow Clients to re-connect and continue using the application partition, without need for human intervention to insert the black PED Key (or gray PED Key) and press [ENTER] on the PED keypad.

AutoActivation, which you set by the `partition changePolicy` command, requires that Partition Policy 23 (Allow AutoActivation) be *on*, for the affected partition.

When you run the `partition activate` command for legacy partitions, or when you simply **role login** for PPSO partitions, autoactivation is set as well (if you set policy 23 for that partition). You are directed to the PED, depending upon the current status of cached data.

If the authentication data requires refreshing, then the PED prompts you to insert the appropriate black or gray PED Key (that is, a PED Key that was imprinted with the partition authentication data for the particular partition [legacy] or role [PPSO]) and press [ENTER]. Once control returns to the command line, and the system announces success, you can remove the black PED Key and store it away. Clients can begin connecting and using the application partition.

We anticipate that most customers will set Partition Policy 23 Allow auto-activation (battery-backed caching of partition authentication) to *on* for their partitions, to ensure the convenience (uptime) of their clients.

Customers who prefer to not set auto-activation *on*, but who keep their SafeNet appliances located remotely from their administrative staff, might prefer to 'manually' resume partition activation by means of Remote PED. These options are entirely a matter of your preference and of your security policy.

To auto-activate an application partition for use by registered Clients

1. Ensure that Activation is switched on (see previous section).
2. Log in as the partition's administrator (HSM SO for legacy partition, Partition SO for PPSO partition).
3. Ensure that the partition policy "Allow auto-activation" has been switched on.

For SafeNet PCIe HSM or SafeNet USB HSM legacy application partition, type:

```
partition changepolicy -policy 23 -value 1
```

For SafeNet PCIe HSM or SafeNet USB HSM or SafeNet Network HSM PPSO application partition, type:

```
partition changepolicy -slot <slot number> -policy 23 -value 1
```

De-Activate a Partition

You can turn off Activation for an HSM Partition by issuing the deactivate command.

To deactivate an application partition

1. To deactivate, do one of the following.

For SafeNet PCIe HSM or SafeNet USB HSM legacy application partition, type:

partition deactivate

For PPSO application partition on SafeNet PCIe HSM or SafeNet USB HSM or SafeNet Network HSM, type:

role deactivate -name <name of role>

2. This turns off activation until the next time a login or activation is performed, at which time the authentication data

is re-cached. Deactivation is temporary un-caching.

If you wish to turn off caching more permanently, so that it does not re-assert at the next login, change the Activation policy.

For SafeNet PCIe HSM or SafeNet USB HSM legacy application partition, type:

partition changepolicy -policy 22 -value 0

For SafeNet PCIe HSM or SafeNet USB HSM or SafeNet Network HSM PPSO application partition, type:

partition changepolicy -slot <slot number> -policy 22 -value 0

3. If Activation is turned off, then autoActivation is also turned off (cached authentication data is cleared) at the same time, but when the SafeNet HSM is Activated again, autoActivation resumes.

To turn off autoActivation, you must turn off the policy.

For SafeNet PCIe HSM or SafeNet USB HSM legacy application partition, type:

partition changepolicy -policy 23 -value 0

For SafeNet PCIe HSM or SafeNet USB HSM or SafeNet Network HSM PPSO application partition, type:

partition changepolicy -slot <slot number> -policy 23 -value 0

Removing Partitions

To remove an HSM Partition, you must be logged in (as admin) to the SafeNet appliance command shell (`lunash`) and you must be logged in to the on-board HSM as HSM Admin. Do the following:

```
partition delete -partition <HSM-Partition-name>
```

Replace `<HSM-Partition-name>` with the name of the Partition that you wish to delete (do not include the angle brackets "`<>`"). If you are not sure of the Partition name, use the `partition list` command.

When a Partition is **deleted**, the Partition is cleared from the HSM and any contents are deleted. This also implies that the Partition is revoked from any Clients that were registered to it.

By contrast, when a Partition is **revoked**, it still exists, but is no longer registered to (and available to) the Client from which it has been revoked. The Partition and its contents could still be used by other Clients, or could be re-assigned to the Client from which it was revoked.

Security of Your Partition Challenge

For SafeNet HSMs with Password Authentication, the partition password used for administrative access by the Partition Owner is also the partition challenge secret or password used by client applications.

For SafeNet HSMs with PED Authentication, authentication is in two parts:

- The partition authentication used for administrative access by the Partition Owner is the secret on the black PED Key(s) for that partition.
- The partition challenge secret or password used by client applications is a separate random character string generated by SafeNet PED at the time the partition is created. This is one way in which we implement separation of roles in the SafeNet HSM security paradigm. The Partition Owner (holder of the black PED Key) can change the challenge secret dispensed by SafeNet PED for one that:

- is more "human-friendly", or
- is compliant with your organization's security policy (or is simply a different password/challenge in compliance with a mandated password-change interval)

How Secure Is the Challenge Secret or Password?

How secure do you want it to be?

When the question is asked, the concern is usually that a password harvesting attack of some sort might eventually crack the secret that protects the partition. Layers of protection are in place, to minimize or eliminate such a risk.

First, such an attack must be run from a SafeNet Client computer. That is not just any old computer.

- For interaction with HSM partitions on a SafeNet Network HSM appliance, like SafeNet Network HSM, a SafeNet Client computer is one with SafeNet software installed, on which you have performed the exchange of certificates to create a Network Trusted Link (NTL). That exchange requires the knowledge and participation of the appliance administrator and the Partition Owner (who might, or might not, be the same person). That is, it is not possible to secretly turn a computer into a Client of a SafeNet HSM partition - an authorized person within your organization must participate.
- For interaction with HSM partitions on a USB-connected or a PCI-e installed HSM, the access to the computer is as restricted as your physical and procedural security makes it. Or, at least, the computer and the contained/connected HSM are as secure as your security regime is consistently enforced.

Second, for SafeNet HSMs with Password Authentication, you set the partition password directly

- when you create the partition, for legacy partitions, or
- when the PSO initializes the partition and creates roles for Per-Partition SO partitions.

Therefore, you can make it as secure as you wish (for an example of guidance on password strength, see <http://howsecureismypassword.net/> or <http://xkcd.com/936/>)

For SafeNet HSMs with PED Authentication, the partition password (challenge secret) is generated randomly, and displayed by the PED at partition creation and is therefore a very secure 16-character alphanumeric string that includes special characters.

Using the `lunash:>` command-line interface, for SafeNet Network HSMs, or the `lunacm:>` interface for SafeNet USB HSMs or SafeNet PCI-E HSMs, you can change the partition password (or challenge secret) if you suspect it has been compromised, or if you are complying with a security policy that dictates regular password changes.

As long as you replace any password / challenge-secret with one that is at least as secure, the possible vulnerability is extremely small.

Conversely, you can choose to replace a secure, random password/challenge-secret with one that is shorter or more memorable, but less secure - you assume the risks inherent in such a tradeoff.

Third, SafeNet HSM Partition Policy number 15 "Ignore failed challenge responses" can be set to "Off" (a value of zero). When that policy is off, the HSM stops ignoring bad challenge responses (that is, attempts to submit the partition secret) and begins treating them as failed login attempts. Each bad login attempt is counted.

Partition Policy number 20, "Max failed user logins allowed" determines how high that count can go before the partition is locked out.

Once a partition is locked by bad login attempts, it cannot be accessed until the HSM Security Officer (SO) unlocks it. So, for example, if you had set "Max failed user logins allowed" to 10, and if you had set "Ignore failed challenge responses" to Off, then an attacker on a client computer would have ten tries before the HSM stopped responding to his attempts. The attacker would then need to wait while the SO intervened to unlock the target partition. At that point,

the attacker would have only ten more attempts until the cycle repeated. This defeats an automated harvesting attack that relies on millions of attempts occurring at computer-generated speeds.

For SafeNet Network HSM, after one or two such lockout cycles, the HSM SO would realize that an attack was under way and would rescind the NTL registration of the attacking computer. That computer would no longer exist as far as the HSM partition was concerned. The SO or your security organization would then investigate how the client computer had been compromised, and would correct the problem before allowing any new NTL registration from that source.

For SafeNet USB HSMs or SafeNet PCI-E HSMs, the administrator of the host computer would be present to reset, or would be remotely managing the host computer and would need to verify the status and view logs on a timely basis. Either way, you cannot get the partition back in service without being aware (acknowledging) that too many attempts have been made.

Summary

The above discussion illustrates that the degree and level of partition security is within your control. As the owner/administrator of the HSM, you can weigh any tradeoffs respecting security, convenience and other operational parameters. Via your security policies and procedures, you decide how much effort an attacker must expend; you control your response and your system's response to any potential attack.

Frequently Asked Questions

This section provides additional information by answering questions that are frequently asked by our customers.

Why do I get an error when I attempt to set the partition policies for activation (22) and auto-activation (23) on my password authenticated SafeNet Network HSM?

Those policies apply to PED-authenticated SafeNet Network HSM, only.

For both PED-authenticated and Password authenticated HSMs, your client authenticates to a partition with a challenge password.

For PED-authenticated HSMs, the application partition must be in a state where it is able to accept that challenge password. That is, the extra layer of authentication - the partition Crypto Officer's black PED Key or the Crypto User's gray PED Key - must have been presented first before the partition can be receptive to the challenge/password.

Password-authenticated HSMs have only the single layer of authentication - the challenge/password is all that is needed. The password is both the client authentication and the partition administrator (Crypto Officer / Crypto User) authentication.

For PED-authenticated HSMs, activation and auto-activation enable caching of the first layer of authentication to provide a level of operational convenience similar to that of the Password-authenticated HSMs.

So, what is the difference in security, once Activation and Auto-activation are started?

From the convenience point of view, none. But, whereas the Password-authenticated partition is "open for business" to anybody with that partition's password, as soon as the partition is created, a PED-authenticated partition is not. One implication is that all partitions of a multi-partition password-authenticated HSM are available whenever any of them are available, which is essentially whenever the HSM is powered on.

The owner of a PED-authenticated HSM partition can disable client access to just one partition by deactivating (de-caching) just that one partition's PED Key authentication, so that the challenge/password is not accepted. Any other partitions on that HSM that are not deactivated (i.e., still have their black PED Key or gray PED Key authentication cached) are still able to accept challenge/password from their clients.

You are not required to cache the PED Key data in order to use a partition. You could, if you preferred, simply leave the PED Key for that partition inserted in a connected SafeNet PED, and press keypad keys on the PED whenever first-level authentication for partition access was required. Since this would defeat much of the reason for having a powerful networked HSM server, generally nobody does this with SafeNet Network HSM in a production environment. (For the kind of application where that scenario might be appropriate, we recommend a host-installed SafeNet PCIe HSM or a USB-connected SafeNet USB HSM.) As well, if you have created both a Crypto Officer and a Crypto User for your partition, you would need to switch out the black PED Key or the gray PED Key, whenever the other entity needed to PED-authenticate, while the PED Key authentications are not cached.

You also have the option of partially engaging the PED Key caching feature by enabling Activation without enabling Auto-activation. In that case, you present your PED Key to activate the partition - which allows it to accept its partition challenge/password from clients - and the cached black PED Key or gray PED Key authentication data is retained while the HSM has power (or until you explicitly de-cache). But the cached authentication does not survive a power outage or an intentional power cycle (because you chose to Activate, but not to AutoActivate as well). Thus, by applying different policy settings, you could have some partitions on your PED-authenticated HSM able to return to client availability immediately following a power-cycle/outage (no human intervention needed), while others would wait for your intervention, with a black PED Key (Crypto Officer) or a gray PED Key (Crypto User), before becoming client-available.

Finally, Activation and Auto-activation are partition-level policy settings, not role-level. Therefore, if the policy is on, it is on for all roles. If the policy is off, it is off for all roles. You cannot individually cache authentication data from a gray PED Key, but not from a black PED Key (or the opposite) within a single partition.

HSM Status Values

Each HSM administrative slot shown in a slot listing includes an HSM status¹. Here are the possible values and what they mean, and what is required to recover from each one.

Indicated Status of HSM	Meaning	Recovery
OK	The HSM is in a good state, working properly.	n/a
Zeroized	The HSM is in zeroized state. All objects and roles are unusable.	HSM initialization is required before the HSM can be used again. "Hard init" - HSM SO and domain are gone, no authentication required. (see Note1)
Transport Mode	The HSM is in Secure Transport Mode.	STM must be disabled, by providing the correct purple PED Key, before the HSM can be used. (see Note2)
Transport Mode, zeroized	The HSM is in Secure Transport Mode, and is also zeroized.	STM must be disabled, by providing the correct purple PED Key (see Note2).and then HSM ("hard") initialization is required before the HSM can be used.
Hardware Tamper	The HSM has been tampered. (MTK is destroyed and must be rebuilt from recovery vectors.)	Reboot the host or restart the HSM (vreset for Thales Luna PCIe HSM, or ureset for Thales Luna USB HSM). The event is logged. If one of the recovery vectors is external (on a purple PED Key) then you are prompted to provide it before the HSM can recover from tamper. Resume using the HSM. (see Note2)
Hardware Tamper,	The HSM has been tampered. (MTK is destroyed and must be rebuilt from recovery vectors.)	Reboot the host or restart the HSM (vreset for Thales Luna

¹The state or condition of a device, as reported in the user interface.

Indicated Status of HSM	Meaning	Recovery
Zeroized	The HSM is also in zeroized state. All objects and roles are unusable.	<p>PCIe HSM, or ureset for Thales Luna USB HSM). The event is logged. If one of the recovery vectors is external (on a purple PED Key) then you are prompted to provide it before the HSM can recover from tamper. (see Note2)</p> <p>HSM initialization is required before the HSM can be used again. "Hard init" - HSM SO and domain are gone, no authentication required. (see Note1)</p>
HSM Tamper	The HSM has been tampered. (MTK is destroyed and must be rebuilt from recovery vectors.)	<p>Reboot the host or restart the HSM (vreset for Thales Luna PCIe HSM, or ureset for Thales Luna USB HSM). The event is logged. If one of the recovery vectors is external (on a purple PED Key) then you are prompted to provide it before the HSM can recover from tamper. (see Note2)</p> <p>HSM initialization is required before the HSM can be used again. "Hard init" - HSM SO and domain are gone, no authentication required. (see Note1)</p>

NOTE1: A condition, not reported above, preserves the HSM SO and the associated Domain, while SO objects and all application partitions and contents are destroyed. HSM SO login is required to perform the "soft init". See ["Initialization Overview for PED-authenticated HSMs" on page 143](#) for more information.

NOTE2: If the HSM is placed in Secure Transport Mode, or if the HSM experiences a Hardware Tamper event while a recovery vector is external to the HSM, and you are unable to provide the requested purple PED Key (with that external recovery vector), then the HSM is unrecoverable. Contact Thales Luna to obtain an RMA and ship the HSM back for re-manufacture.
(Applies to PED-authenticated HSMs only.)

Indicated Status of HSM	Meaning	Recovery
-------------------------	---------	----------

If your HSM is Password-authenticated, or if your PED-authenticated HSM has both recovery vectors internal (no purple PED Key was created), then if a tamper event destroys the MTK, the HSM recreates the MTK after being restarted, and no further intervention is required.

The above scenarios assume that a tamper event is transient, and the cause is corrected. If the HSM remains in tamper, or immediately returns to tamper, then contact Thales Luna Technical Support.

For comparison and detailed explanation of "hard init" vs "soft init", see ["Initialization Overview for Password-Authenticated HSMs" on page 141](#) and ["Initialization Overview for PED-authenticated HSMs" on page 143](#).

For a comparison of various destruction or denial actions on the HSM, see ["Comparison of Destruction/Denial Actions" on page 309](#).

Key Migration

This chapter describes how to migrate key material from one HSM to another. It contains the following sections:

- ["Key Migration Procedures" below](#)
- ["Migrating Key Material from Older \(2U\) to New \(1U\) Appliances" below](#)
- ["Frequently Asked Questions" below](#)

Key Migration Procedures

If you have other SafeNet HSMs on which you have important keys or data, you can securely migrate that material to another SafeNet HSM. Contact SafeNet Technical Support and ask for the following document:

- 007-011528-001 *SafeNet HSM Key Migration Guide*

Migrating Key Material from Older (2U) to New (1U) Appliances

Contact SafeNet Technical Support at support@safenet-inc.com or www.safenet-inc.com/Support

Frequently Asked Questions

We want to generate keys on one HSM and copy them to other HSMs. Can they have the same object handles?

No. You can clone keys between HSMs that share a domain, but each HSM assigns its own object handles to incoming - or generated - objects.

Good PKCS#11 applications **never** make assumptions about the object handle number.

Typically, an application will find an object prior to use; for example, find by CKA_LABEL is the most common.

The label either is known to the user or is published somewhere application-specific; for example, Microsoft uses the certstore to store the label (a.k.a. container name).

Possible workarounds:

If your application already uses handles to access/identify keys, consider identifying keys by fingerprint (and possibly label) and devising your own mapping to the new handles for objects that you import (clone) into the HSM.

HOWEVER, that approach might not be feasible if you are not in a position to make API changes - such as, if you are using a third-party application, or if you are locked in by internal compliance/audit or by external compliance/audit. Then, perhaps you could consider using multiple HSMs in an HA group.

If you are accessing via an HA group, then the HA group has a single virtual handle for each object that your application would see, regardless of the "real" object handle on each HSM.

We want to migrate from a Microsoft Certificate Authority to a Linux CA while keeping the same private key. Does the SafeNet HSM offer any barriers to doing this?

This is not recommended. It is not an issue of the HSM. Rather it is an issue of the software that you use to run your CA. When you generate a key in the HSM, it is stored internally in a partition. The key can be used by any application that has appropriate access and can successfully authenticate to the partition. That application could be the Microsoft CA, a Linux CA, or both, or other.

Most applications expect to generate and control their own key material. If your Linux CA allows you to point at an HSM and say "use that key", then the SafeNet HSM does not prevent it from doing so. However, as an example, when Microsoft CA creates the root key, it embeds a representation of the machine name in the key, to enforce that only that machine can have access to it. In the Microsoft world it is possible to get around this obstacle by using "clusterkey", but it is not clear how the Linux CA would react, as we have not tested such configurations.

Generally, the "best practice" that we recommend for switching from one PKI platform to another, or from one HSM vendor to another, is to implement the new PKI with a new root and issuing CA key, while leaving the original PKI in place. All new certificates are issued from the new PKI, and the original PKI (no longer used to issue certificates) is allowed to phase out over time, as each certificate that was issued from it expires.

We want to migrate from a Microsoft IIS with existing keys stored in software to CSP with keys stored in the SafeNet HSM. We have followed the steps to import using ms2luna utility, but it appears that IIS is still using the keys in software.

The migration from Microsoft CSP to SafeNet CSP does a "copy" of the keys, while a SafeNet KSP migration (for Microsoft CNG) does a "move" of the keys. So if you migrated to SafeNet CSP, the application might still be looking for keys (and finding them) at the original location; it needs to be told to use the keys that are now on the HSM.

Check the value of your key in registry location: HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\services\CertSvc\Configuration\[CA NAME]\CSP\Provider

PED Authentication

This chapter describes PED-based HSM authentication. It contains the following sections:

- ["About the SafeNet PED" below](#)
- ["Using the PED" on the next page](#)
- ["About PED Keys" on page 186](#)
- ["What is a PED PIN?" on page 195](#)
- ["How to Use a SafeNet PED" on page 204](#)
- ["Interaction between the HSM and the PED" on page 206](#)
- ["Lost PED Keys or PED PINs, or passwords" on page 214](#)

About the SafeNet PED

SafeNet PED is a PIN Entry Device, where PIN stands for Personal Identification Number. The PED works in conjunction with HSMs and backup tokens from SafeNet. It provides PIN entry to SafeNet HSMs and to backup tokens via a secure data port, as part of FIPS 140-2 level 3 security (the Trusted Path). PED 2.x is the current generation. A migration path is available if you have the legacy SafeNet PED 1.x - contact Gemalto Technical Support.

The PED is shipped separately from your HSM product, because one PED can be used with any Trusted Path HSM. A PED with firmware version of 2.0 or later is also RoHS-compliant. The version is displayed on the PED display panel, each time the PED is powered on.

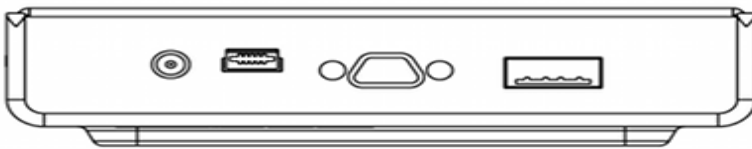
As well, you require a set of at least three PED Keys. For PED 2.0 and later, the PED Keys are in the form of hardware identification tokens, SafeNet iKey model 1000 (RoHS-compliant) or possibly other SafeNet iKey models, to be introduced at a later date. For most applications, you would want an additional set to make duplicates for backup purposes (and, optionally, several more PED Keys, if you intend to use the MofN authentication option with a SafeNet HSM product that supports MofN).

PED Features

The figure below shows a front view of the PED, with some important features indicated.



1. On the lower front face is the keypad for command and data entry.
2. On the upper front face is the 8-line liquid crystal display (LCD).



3. At the top on the far left is a DC power-adaptor connector (not used when PED is connected directly to an HSM - local PED).
4. At the top, second from the left is a USB mini-B connector, reserved for file transfer to/from the PED.
5. At the top in the middle is a micro-D subminiature (MDSM) connector for the cable to the HSM (data and power).
6. At the top, on the far right, is the USB A-type connector for iKey-style PED Keys.
7. Also shown is an iKey PED Key, for insertion in the PED Key connector, and described in these pages.

The visible difference between the standard (local-only) PED 2 and the Remote Capable PED 2 is a sticker on the upper right-front panel, either local

PED local

or remote

PED remote

Using the PED

A SafeNet PED is required to authenticate to an HSM that requires PED (Trusted Path) Authentication.

The requirement for Trusted Path Authentication, as opposed to Password Authentication, is part of the specific model of HSM, as configured at the factory (the one exception is the SafeNet Backup HSM, which configures itself, at

backup time, as either Password-authenticated or PED-authenticated, depending on the type of primary HSM it is backing up).

Figure 1: PED (2.x) front view

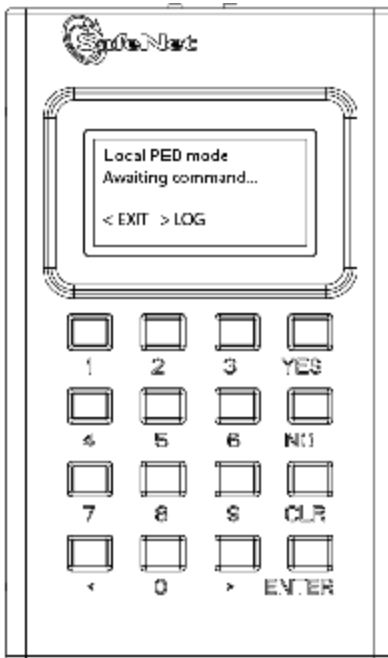
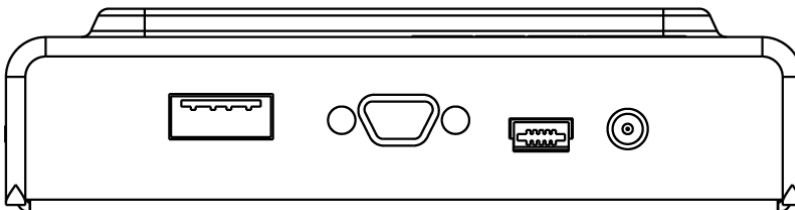


Figure 2: PED top view



Interaction with Other Operations

HSM firmware version 6.24.0 introduces a change in how ongoing PED operations interact with cryptographic operations requested simultaneously.

Behavior before HSM firmware version 6.24.0

PED operations interrupt other operations occurring at the same time on the HSM. The HSM waits for a PED operation to complete before processing requests for other operations. This can cause delays in production.

Behavior after HSM firmware version 6.24.0

PED operations no longer interrupt other operations occurring at the same time on the HSM in most cases. The most beneficial effect is that PED operations acting on a partition no longer block operations occurring on other partitions on the same HSM. For example, you can now create new partitions or backups while running cryptographic

operations on a separate partition. In this way, you can perform maintenance and configuration on your HSM without interrupting important client applications. PED operations might still block cryptographic operations occurring on the same partition, especially high volumes of write object requests.

Versions

PEDs are generally unit-interchangeable (with limitations within the version range, PED 2.x, see table), and more specifically interchangeable within the same PED-firmware version. That is, if a SafeNet PED with a given firmware supports your current operation with your current HSM version, then any SafeNet PED with the same, or newer, firmware can replace it.



Note: Exception - If you are using the Remote PED feature, only another PED with Remote capability can support that operation, regardless of firmware version.

Newer PED firmware versions are compatible with HSM versions shown in their row in the table, and backward compatible with any earlier HSM that requires a version 2.x PED.

PED firmware version	Local PED operation and Remote PED capable	PED-mediated MofN per secret (with HSM f/w 6.x) also SRK (purple PED Key) and Secure Transport Mode	Field update-able	Audit User (white PED Key)	Small Form-factor Backup	PED version is feature-compatible with SafeNet HSM firmware version(s)
2.2.0	Yes	No	No	No	No	SafeNet HSM 4, f/w 4.x
2.4.0-3	Yes	Yes	To 2.5.0	No	No	SafeNet 5.0, f/w 6.0.8 SafeNet 5.1.x, f/w 6.2.1
2.5.0-3	Yes	Yes	To 2.6.0	Yes	No	SafeNet 5.2, f/w 6.10.2 SafeNet 5.3.1 f/w 6.20.0
2.6.0-6	Yes	Yes	Yes	Yes	Yes	SafeNet 5.4, f/w 6.21.0 SafeNet 6.0, f/w 6.22.0

PED firmware 2.2.0 is mentioned in the table above because many customers who first bought SafeNet HSM 5.0 were already in possession of older PEDs since they already had earlier SafeNet HSMs (f/w 4.x). SafeNet HSM 5.0 needed PED f/w 2.4.0-3 to access all functions.

PED firmware 2.5.0-3 or newer is suitable for all local and remote authentication and is required for some PED-mediated features added since SafeNet HSM 5.0.

PED firmware 2.6.0-x, available as a field upgrade or on newly-purchased PEDs, supports SafeNet Small Form-Factor Backup - a completely separate function mediated by SafeNet PED, and using different USB tokens - and also supports all previous PED 2.x authentication functions.

Authentication

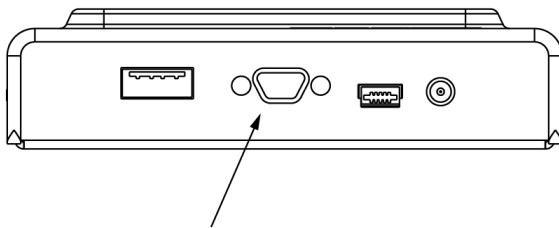
In this current discussion, we ignore SFF Backup, and focus on the HSM authentication function of SafeNet PED. The authentication information for your HSM roles is contained on the PED Key, and SafeNet PED is the device that provides the interface so that authentication data can pass between PED Key and HSM (see ["About PED Keys" on page 186](#)).



The keypad on the PED is used to acknowledge prompts (on the PED screen) and to optionally input a "something you know" additional secret, called a PED PIN (see ["What is a PED PIN?" on page 195](#)) to augment the "something you have" secret contained in the PED Key.

Local and Remote

A locally-connected PED is powered by its connection to the HSM.



That connection - directly between the PED and the HSM card inside the host - bypasses your computer bus and the computer bus of the HSM host (if separate). It is the only data path between the HSM and the PED and therefore is considered much more secure (trusted) than any authentication path that passes through the appliance's computer data paths. The Trusted Path cannot be monitored by any software (whether authorized by you or not) on your administrative or client computer. Also, because you use the PED Keypad to input the optional PED PIN password (to unlock the secret that, in turn, unlocks your HSM see ["What is a PED PIN?" on page 195](#)), nothing is typed on a computer keyboard. No virus, trojan, spyware, remote-session software or other method can be used to acquire those secrets, because they never pass through the normal computer data pathways, never reside in computer memory or on disk.

With HSM appliances and host computers often tucked away in server farms, which are frequently run as "lights-off" facilities with the minimum possible human intervention, the PED cannot always be conveniently connected directly to the HSM. Instead, a callback server arrangement (Remote PED) uses a SafeNet PED connected to a separate computer, at a convenient location, to serve PED interactions over a network connection. The connection is strongly secured and, like the direct connection, prevents unauthorized persons from gaining access to the authentication data. A Remote PED does not have the direct connection to an HSM to provide power; it uses a USB connection for data exchange, which might not provide sufficient stable power for operation. Therefore a PED used in Remote mode also needs a dedicated power connection via the provided power block.

For both local and remote PED use, the only way for another person to discover a PED PIN password while you are inputting it is if you allow that person to observe while you use the PED keypad.

When Do I Need A PED?

You need to use the PED whenever you perform any action with the HSM that causes it to look for authentication (with some exceptions, see below). For example, using the shell (lunash on SafeNet Network HSM) or Lunacm (for any SafeNet HSM) you might login as Security Officer, login as User, or initialize the HSM. When the HSM receives such a command, it requests the appropriate data from the PED - or in the case of initialization, the HSM might send data to the PED.

Therefore, you should have the PED connected and in its ready state ("Awaiting command...") when you issue a command that invokes the PED. One MDSM connector attaches to the matching connector on the HSM or appliance, and the other MDSM (Micro-D Sub-Miniature) connector attaches to its matching connector on the top of the PED (tighten the connector screws if you intend to leave the PED connected; this makes the most reliable connection and provides strain relief to the cable-connector junction during handling of the device).

If you are using the Activation/autoActivation feature then, after authentication, the data is cached on the HSM, where it remains until you deactivate or you remove power to the HSM. In that case, once the authentication is performed, you can disconnect the PED and store it until the next time it is required.

If you are not using activation, then authentication data is not cached and every time you or your client application needs access to the HSM, the HSM will look to the PED, which must remain connected.

For Remote PED connections, the MDSM connector is not used, and power and USB connections are used instead.

What Do I Do?

As soon as it receives power from a connection to a powered HSM, or from the supplied power block if you are using Remote PED, the PED performs its start-up and self-test routines and then goes to its normal operating mode, displaying the prompt "Awaiting command...". The PED is ready for use in Local mode, by default.

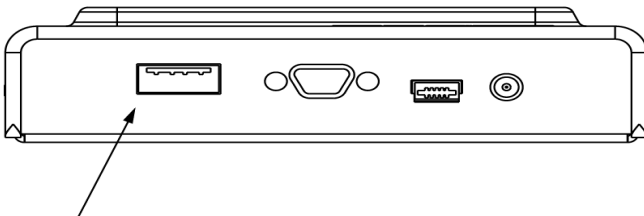
There are three things that you can do with the PED at this point:

- Wait for a prompt, which results when a program has caused the HSM to request authentication
- Change to the Remote Mode (which expects encrypted commands from a computer USB connection, where you would be running Pedserver, rather than from a direct PED-HSM connection)
- Perform standalone PED operations.

To perform prompted actions, just do what is asked on the PED screen. Normally the prompts are:

- Insert a PED Key
- Press "YES", "NO" or "ENTER" on the keypad

Insert and remove appropriate PED Keys, type numeric passwords (PED PINs) when requested, and so on. The particular sequence depends upon the operation that the HSM needs at the time, which in turn depends on the command-line administrative operations that you are performing (with lunacm, lunadiag, multitoken, or other SafeNet utilities), or operations triggered by your applications.



The operations "[Initializing a PED-Authenticated HSM](#)" on page 1 and "[Prepare to Create a Partition \(PED Authenticated\)](#)" on page 1 are described elsewhere in this documentation.

In normal practice, you would perform initial configuration operations one time before placing the unit in service, then perform only monitoring and occasional maintenance thereafter. See the table below for a simple breakdown of the normal tasks and if/how the PED and PED Keys might apply.

Situation	Needs	Action with PED and PED Keys
Setup/configuration	Blue, red and black PED Keys and PED. Optionally a purple PED Key, if you or someone invoked Secure Transport Mode, and an orange PED Key if an RPK was already created, and you are performing these actions remotely.	You perform the HSM initialization, possibly create Partition Groups with other HSMs, set up a redundant, load-sharing HA group with other SafeNet HSMs. This is the kind of chore you must perform before first putting the unit into "production", and then might never need to do again. The PED Keys are required at several stages, as well as the PED.
Occasional Maintenance of HSM	Appliance admin password, blue and black PED Keys, possibly the red if you need to initialize a new cluster member, and the PED. Network connection to the appliance.	Add and remove cluster members, modify number and assignment of Partitions/Groups, enable and disable... you might need some or all PED Keys for authentication, depending on the activity.
Client access to their assigned HA group partitions	Clients need their own authentication that is set up before client applications are launched to use a partition or an HA group virtual partition; no PED Key or PED required, but you need the Crypto Officer and/or Crypto User challenge secrets.	None. You would normally have activated the individual HSM or (in case of HA) the HA-group members (in other sections of this table), and put the PED and PED Keys away in safe storage. They aren't needed in ongoing operation.
PED Key administration	A PED and whichever PED Keys you wish. You can connect to any SafeNet HSM that has the proper connector - this is to power the PED only. Alternatively, you can use the PED power supply kit provided with Remote PED, and not need any HSM connection.	While you can perform some PED Key admin during HSM operations (mentioned elsewhere), you can also just power up the PED, go to Admin mode (instead of the default "Local PED" mode), and perform actions like creating duplicates of your existing, imprinted PED Keys. No HSM access is required. See the next section on this page (below) for more detail.

Standalone or local or off-line PED operations

You can perform some operations on PED Keys without going through the HSM.

To perform standalone operations:

1. Press the "<" key to exit from Local PED mode.
2. Press "4" to enter Admin mode.

3. In Admin mode, options are 1 PED Key or 7 Software Update. (The software update function is rarely used and requires that you be sent a PED software file from SafeNet, along with directions about how to use it. Therefore, we'll assume that you are selecting "1 PED Key", which brings the PED to PED Key mode.)
Press "1".
4. To perform an operation on a particular PED Key, insert that PED Key into the PED Key connector on top of the PED.
5. PED Key mode has an option "1" to login to that PED Key, which applies to models other than iKey 1000 PED Keys - just press "1" to get to the next menu, if you are using iKey 1000 PED Keys, which do not need login.
6. At the PED Key Mode menu you have options to Login (which you have just done, but the prompt is available in case you might wish to login to a different PED Key) , Logout, or Duplicate the PED Key. Only the "Duplicate" option is meaningful for your iKey 1000 PED Key. To **duplicate** the contents of the currently connected PED Key to another (blank or re-used) PED Key, press "7" on the PED keypad.
7. When prompted, insert a blank target PED Key, or a non-blank whose data is no longer needed, and press ENTER.
8. If data already exists on the target PED Key, you are warned and required to press YES two times, to confirm that you really do wish to overwrite whatever is on the PED Key that is currently connected to the PED.
If the source PED Key had an optional PED PIN assigned, then that PED PIN is automatically applied to the duplicate during this process.
9. Remove the newly imprinted PED Key and press ENTER. The PED goes back to "PED Key mode" awaiting further commands. If you wish to duplicate another PED Key, repeat the above steps. Otherwise, press "<" to go back to "Admin mode", and press "<" again to reach the main menu, and finally press "1" to resume "SCP mode", which is the normal operating mode of the PED, awaiting commands from the connected HSM.
10. Identify the new PED Key with a tag or other marker, and record a PED PIN (if any) in secure fashion, according to your security policies.

EXCEPTION: Secure Recovery

The PED will not perform a standalone copy operation (that is, without an HSM) of a purple PED Key. This is a security feature. You can copy a purple PED Key (just like any other PED Key for any other HSM role or function) during an imprinting operation controlled by a SafeNet HSM. Because purple PED Keys are specific to a single HSM, no other HSM can share a purple key or make a copy. The PED refusal to make standalone copies of purple keys is just an additional barrier to anyone wanting to attack an HSM that has been placed in Secure Transport Mode.

EXCEPTION: Remote PED

The Remote PED 2 functions as described earlier, when it is in Local or Admin mode. However, when it is placed in Remote mode, it is capable of setting up a secure connection, via a specially-configured computer workstation, to a remotely located HSM. The remote functionality is described separately at ["About Remote PED" on page 270](#).

About PED Keys

A PED Key is an electrically-programmed device, with USB interface, embedded in a molded plastic body for ease of handling. Specifically, a PED Key is a SafeNet iKey authentication device model 1000(must be firmware version 1.1 or later - the PED checks the firmware version of a presented iKey, and displays an error message if the version is too old) with FIPS configuration. In conjunction with PED 2 or PED 2 Remote, a PED Key can be electronically imprinted with identifying information, which it retains until deliberately changed.

A PED Key holds a generated secret that might unlock one or more HSMs. That secret is created by initializing the first HSM. The secret can then be copied (using PED 2.x) to other PED Keys, for purposes of backup, or to allow more than one person to have access to HSMs that are protected by that particular secret. The secret can also be copied to other HSMs (when those HSMs are initialized), so that one HSM secret is able to unlock multiple HSMs.

The HSM-related secret might be the access control for one or more HSMs, the access control for Partitions within HSMs, or the Domain key that permits secure moving/copying/sharing of secrets among HSMs that share a domain.

The PED comes in two versions:

- the standard PED 2 is designed for local connection, only, to a SafeNet HSM
- the Remote PED 2 has all the function of the standard PED 2 and can also be used remotely from an HSM, when used with PEDServer.exe workstation software.

Why do you need PED Keys?

The PED and PED Keys are the only means of authenticating, and permitting access to the administrative interface of the PED-authenticated HSM, and are the first part of the two-part Client authentication of the FIPS 140-2 level 3 (FIPS is the Federal Information Processing Standards of the United States government's National Institute of Standards and Technology -- FIPS 140-2 is an internationally recognized standard regarding security requirements for cryptographic modules, and level 3 is its second-highest level of security features/assurance) compliant SafeNet HSM with Trusted Path Authentication. See "[About FIPS Validation](#)" on page 363 for more information.

The use of PED and PED Keys prevents key-logging exploits on the host HSM, because the authentication information is delivered directly from the hand-held PED into the HSM via the independent, trusted-path interface. You do not type the authentication information at a computer keyboard, and the authentication information does not pass through the internals of the computer, where it could possibly be intercepted by spy software.

The PED does not hold the HSM authentication secrets. The PED facilitates the creation and communication of those secrets, but the secrets themselves reside on the portable PED Keys. This means that an imprinted PED Key can be used only with HSMs that share the particular secret, but PEDs are interchangeable (at least, within compatible versions - you can replace any PED 2.x with any other [unless otherwise indicated], but you cannot use a PED 1.x where a 2.x version is needed, or vice-versa).

Types of PED Key

The current-model PED uses iKey USB-fob type PED Keys of no particular color (the standard issue is black) for all functions. You can visually differentiate your PED Keys by attaching tags or labels. A set of sticky labels in appropriate colors (see below) is supplied with your PED Keys.

The roles and uses of the PED Keys employed with SafeNet HSMs and the PED are as follows:



SO



Security Officer (SO)'s (also sometimes called HSM Admin) PED Key. The first actions with a new SafeNet HSM involve creating an SO PIN and imprinting an SO PED Key. The SO identity is used for further administrative actions on the HSMs, such as creating HSM Partition Users and changing passwords, backing up HSM objects, controlling HSM Policy settings.

A PED PIN (an additional, optional password typed on the PED keypad) can be added. SO PED Keys can be duplicated¹ for backup, and can be [shared among HSMs](#) by imprinting subsequent HSMs with an SO PIN already on a PED Key.

Partition User or Crypto Officer



Application Partition User key or Crypto Officer key. For HSM SO-controlled application partitions (with no SO local to the partition) this PED Key is required to log in as HSM Partition Owner or Crypto Officer. Needed for Partition maintenance, creation and destruction of key objects, etc. Creates the optional Crypto User.



Note: Creation of a challenge secret is forced for an application partition owned by the HSM SO (a.k.a. "Administrator").

For application partitions that have their own SO, this PED Key is required to log in as Crypto Officer, and is needed for Partition maintenance, creation and destruction of key objects, etc. The Crypto Officer creates the Crypto User.

The challenge secret generated in conjunction with the Crypto Officer can grant client applications access to create, delete, and manipulate partition objects.



Note: Creation of a challenge secret is not forced for an application partition with its own SO.

A PED PIN (an additional, optional password typed on the PED keypad) can be added. Black Crypto Officer PED Keys can be duplicated² for backup, and can be shared among application Partitions using the "Group PED Key" option.

¹a PED Key can be copied so that two or more PED Keys contain the same secret - this is useful and necessary in order to have backups of each of your PED Keys, and for other operational purposes, but you must maintain rigorous control of all duplicates to prevent unauthorized persons from accessing your HSM(s), and for tracking of the "paper trail" of possession, to satisfy your security auditors.

²a PED Key can be copied so that two or more PED Keys contain the same secret - this is useful and necessary in order to have backups of each of your PED Keys, and for other operational purposes, but you must maintain rigorous control of all duplicates to prevent unauthorized persons from accessing your HSM(s), and for tracking of the "paper trail" of possession, to satisfy your security auditors.

Crypto User



The Crypto User has restricted read-only administrative access to application partition objects. The challenge secret generated in conjunction with the Crypto User can grant client applications restricted, sign-verify access to partition objects.

Note: Creation of a challenge secret is forced for an application partition owned by the HSM SO. Creation of a challenge secret is not forced for an application partition with its own SO.

A PED PIN (an additional, optional password typed on the PED keypad) can be added. Gray User PED Keys can be duplicated¹ for backup, and can be shared among application Partitions using the "Group PED Key" option.

Domain



Key Cloning Vector (KCV) or Domain ID key. This PED Key carries the domain² identifier for any group of HSMs for which key-cloning/backup is to be used. The red PED Key is created/imprinted upon HSM initialization. Another (or could reuse the same domain) is created/imprinted with each HSM Partition. A cloning domain key carries the domain (via PED) to other HSMs or HSM partitions which are to be initialized with the same domain, thus permitting backup and restore among (only) those containers and tokens. The red Domain PED Key receives a domain identifier the first time it is used, at which time a random domain is generated by the HSM and sent to both the red Domain key and the current HSM Partition. Once imprinted, that domain identifier is intended to be permanent on the red Domain PED Key – and on any HSM Partitions or tokens that share its domain. Any future operations with that red Domain PED Key should simply copy that domain onto future HSM Partitions or backup tokens (via PED) so that they may participate in backup and restore operations (see [What is a Domain PED Key?](#) later in this section, for a more detailed explanation). A PED PIN (an additional, optional password typed on the PED keypad) can be added at the time the PED Key is created/imprinted. Red PED Keys can be duplicated³ for backup or multiple copies of the key.

The red PED Key can be considered the most important PED Key to protect from access by unauthorized persons. An unauthorized person who is able to access the HSM host could see and manipulate objects on a logged-in or activated partition, but would be able to copy those objects to another HSM only if he had possession of the partition domain secret. Without the proper red PED Key, an attacker cannot copy/clone HSM partition contents to other HSMs.

¹a PED Key can be copied so that two or more PED Keys contain the same secret - this is useful and necessary in order to have backups of each of your PED Keys, and for other operational purposes, but you must maintain rigorous control of all duplicates to prevent unauthorized persons from accessing your HSM(s), and for tracking of the "paper trail" of possession, to satisfy your security auditors.

²(Also referred to as KCV – Key Cloning Vector) A domain is a shared identifier, common to a group of Luna cryptographic modules, with access controlled by a red PED Key (for Trusted Path Authentication) or by a domain string (for Password Authentication). Cloning (secure duplication) of token objects is possible among tokens/HSMs that share a particular domain. Cloning is not possible across different domains, and is not possible where the tokens lack a domain. A domain must be declared and imprinted at the time a token is initialized.

³a PED Key can be copied so that two or more PED Keys contain the same secret - this is useful and necessary in order to have backups of each of your PED Keys, and for other operational purposes, but you must maintain rigorous control of all duplicates to prevent unauthorized persons from accessing your HSM(s), and for tracking of the "paper trail" of possession, to satisfy your security auditors.

Remote PED



SafeNet HSM
Remote PED

This PED Key is required when you need to perform PED operations at a distance. The orange RPK carries the Remote PED Vector (RPV) and allows a SafeNet PED connected to a properly configured computer to substitute for a PED connected directly to the SafeNet appliance/HSM, when that local connection is not convenient.

The RPV is created/imprinted by a SafeNet HSM with a suitable PED connected (version 2.4.0 or later, having the Remote PED feature installed). A Remote PED can be connected to the USB port of a networked computer that has the PED driver installed and is running the PEDserver.exe program. A SafeNet HSM (that has been initialized with a Remote PED vector) can initiate a secure connection to the Remote PED Server computer, and that connection can be validated by an orange Remote PED Key that carries the same vector as the SafeNet HSM. For the duration of that session, HSM commands can be run at that appliance with all the needed PED Keys (SO, User, Domain, even SRK) being supplied via the PED connected to the computer. There is no need to be present at the remotely located SafeNet appliance/HSM with PED Keys and PED. Orange PED Keys can be duplicated¹ for backup or multiple copies of the key.

Secure Recovery



SafeNet HSM
Recovery

The purple Secure Recovery Key contains the external split of the SRV (secure recovery vector), to recreate the HSM's Master Tamper Key with which all HSM contents are encrypted. The master key is destroyed whenever a tamper event occurs, or when the HSM is deliberately set to Secure Transport Mode. For Secure Transport Mode, the purple PED Key is then shipped via a separate channel from the HSM shipment so that no attacker could obtain access to both the HSM and the SRV while they are in transit. Upon receipt, the administrator brings both the HSM and the purple key together, and invokes the "hsm srk recover" command. This brings the internal (in the HSM) and external (on the purple SRK) components of the SRV together and recreates the HSM Master Tamper Key, allowing the HSM to be used.

A PED PIN (an additional, optional password typed on the PED keypad) can be added. Purple PED Keys can be duplicated² for backup or multiple copies of the key, but only during the creation/imprinting of the key - SafeNet PED cannot duplicate purple keys via the standalone PED admin menu. Purple PED Keys are unique to each HSM, and cannot be shared.

Audit



SafeNet HSM
Audit

Audit is an HSM role that takes care of audit logging, under independent control. The audit role is initialized and imprints a white PED Key, without need for the SO or other role. The Auditor configures and maintains the audit logging feature, determining what HSM activity is logged, as well as other logging parameters, such as rollover period, etc. The purpose of the separate Audit role is to satisfy certain security requirements, while ensuring that no one else - including the HSM SO - can modify the logs or hide any actions on the HSM. The Audit role is optional until initialized.

For SafeNet USB HSM and SafeNet PCIe HSM see the audit commands in lunacm:>.

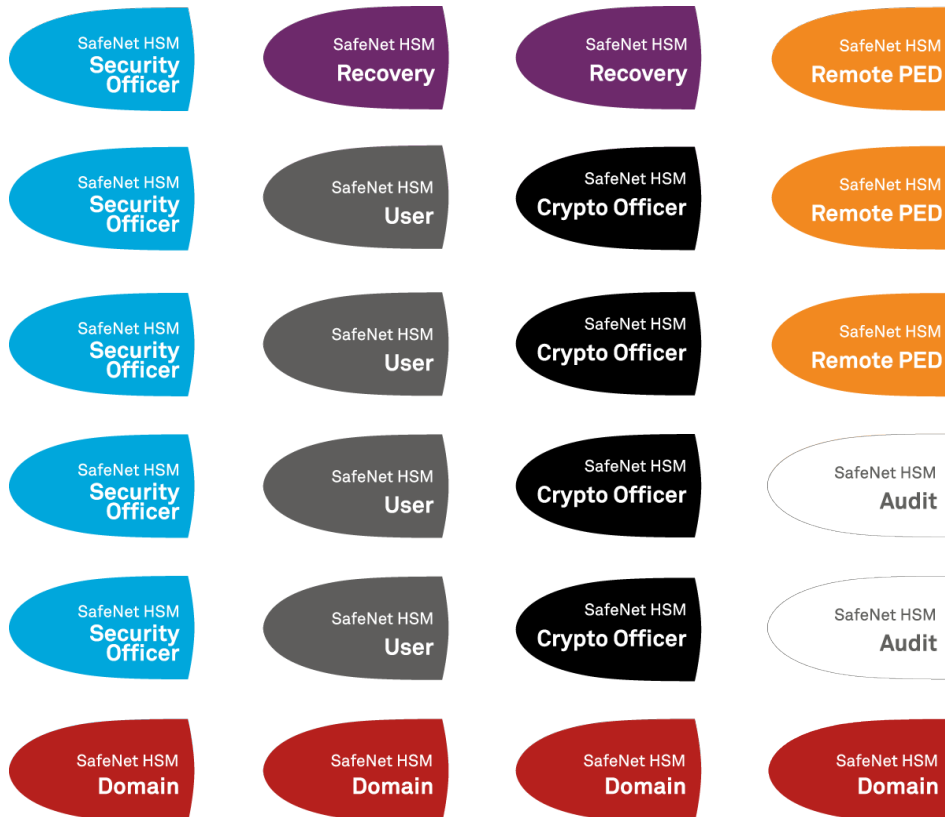
¹a PED Key can be copied so that two or more PED Keys contain the same secret - this is useful and necessary in order to have backups of each of your PED Keys, and for other operational purposes, but you must maintain rigorous control of all duplicates to prevent unauthorized persons from accessing your HSM(s), and for tracking of the "paper trail" of possession, to satisfy your security auditors.

²a PED Key can be copied so that two or more PED Keys contain the same secret - this is useful and necessary in order to have backups of each of your PED Keys, and for other operational purposes, but you must maintain rigorous control of all duplicates to prevent unauthorized persons from accessing your HSM(s), and for tracking of the "paper trail" of possession, to satisfy your security auditors.

A PED PIN (an additional, optional password typed on the PED keypad) can be added. White Audit PED Keys can be duplicated¹ for backup, and can be shared among HSMs using the "Group PED Key" option.

What is a Set of PED Keys?

A nominal set of **blank** PED Keys, as purchased with a SafeNet HSM with PED (Trusted Path) Authentication, consists of ten black USB-token PED Keys, along with colored stickers to identify them (several each of blue, red, black, orange, white, and purple), which allows some spares or backups.



The stickers (above) are just visual labels to attach to your PED Keys. They are provided for your convenience, and you can use them, or not, at your discretion.

The set of stickers/labels does not indicate a requirement. The quantity of each color on a sheet is merely an average distribution according to customer practices that we have seen.

You are not required to have a PED Key of each color, above. The ones that are mandatory for use of your PED-authenticated HSM are the blue SO key, the red domain key, and the black Crypto Officer key.

- Operation of the HSM does not require that you create a Crypto User - it is optional, in case you need a limited-capability client.
- Operation of the HSM does not require that you create an Audit user - if your situation does not require audit logging, that is, if standard, unverified HSM logging is sufficient, then the Audit role is superfluous to your needs.

¹a PED Key can be copied so that two or more PED Keys contain the same secret - this is useful and necessary in order to have backups of each of your PED Keys, and for other operational purposes, but you must maintain rigorous control of all duplicates to prevent unauthorized persons from accessing your HSM(s), and for tracking of the "paper trail" of possession, to satisfy your security auditors.

On the other hand, if your implementation is likely to be audited against security standards, then creating, assigning, and using the HSM's Audit role would be recommended.

- Operation of the HSM does not require that you imprint a Remote PED Key, as long as you have local access for PED interactions, when needed. If you expect to administer the HSM remotely, then a Remote PED Key will be required.
- Operation of the HSM does not require that you imprint a Secure Recovery PED Key. If no external Secure Recovery Vector is generated, then the HSM can recover from tamper events with only a login, and the HSM cannot be placed in Secure Transport Mode. If you wish to invoke (and later recover from) Secure Transport Mode, or if you require that the HSM halt in case of a tamper event, and require intervention (presentation of a purple PED Key) to acknowledge and recover from such tamper, then enabling SRK would be necessary.

Imprinting

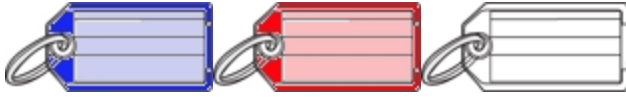
- A PED Key can contain only one HSM authentication secret at a time.
- PED Keys are completely interchangeable before they are imprinted by your action - the PED checks for an existing authentication secret, and tells you if the currently presented key is blank.
- PED Keys are imprinted by SafeNet PED during HSM initialization and Partition creation and other HSM actions that create HSM roles or invoke certain HSM functions.
- PED Keys can be re-imprinted with new HSM authentication secrets. Imprinting a new secret overwrites (destroys) any HSM authentication secret that was already present on a PED Key when it was presented for new imprinting. The PED always warns you if the presented key contains an authentication secret. The PED has no way to know if an authentication secret that it finds on a key is useful and valid for some role on the current HSM (or on some other HSM), or if a contained authentication secret is outdated and no longer valid, and therefore safe to overwrite - so the PED simply tells you what it has found and lets you make the decision.
- PED Keys are completely interchangeable for re-imprinting (except see Note, below) - that is, you can turn any PED Key into a different PED Key during an initialization/imprinting operation; the PED warns you that the key you have presented already contains an authentication secret (and tells you what kind), but you can choose to overwrite if you think the currently imprinted secret is no longer useful.

Note: The exception is the purple SRK PED Key. If you attempt an operation that creates and imprints a new SRK value, the PED will accept any blank key or any non-purple, previously-imprinted PED Key to accept the new external recovery vector for the currently-connected HSM. That means, it will imprint a new SRV onto any blank key, and it will also imprint a new SRV onto any key that currently contains authentication for SO, CO, CU, Domain, Auditor, RPK, if you tell it to overwrite.



But, it will not overwrite a purple PED Key that contains the currently valid SRV for the current HSM. This is a safety feature.

Keep in mind that the PED has no way to determine if a discovered SRV secret on a key is currently valid for some other HSM. It can check only with the currently-connected HSM.



We recommend that you use some system of visually identifying the role of each PED Key once it is imprinted. Ordinary key-chain tags are handy and can be acquired anywhere; they provide room for written information that is important to you, and they do not interfere with the operation of the PED Keys.

We strongly suggest that you use our supplied self-stick PED Key labels, or that you otherwise maintain the color associations that are referenced throughout the documentation and also in the HSM utilities and in SafeNet PED's own dialogs.

- Security Officer (SO) key - blue
- domain key - red
- Crypto Officer key (or partition Owner/ key, old scheme) - black
- Crypto User key - gray
- Remote PED - orange
- Secure Recovery (SRK) - purple
- Audit role - white

The others are spares for each role. The SO, Domain, and User roles are the minimum that you need to operate the HSM.

For purposes of backup redundancy, you would normally have at least a second full set for keeping in safe storage, once they have been imprinted. Imprinting takes place when an HSM is initialized (a backup token is initialized/re-initialized whenever a backup is performed onto it). Initialization is also an opportunity to make more duplicates of any PED Key, if you require them. Imprinting of Partition PED Keys takes place when an HSM Partition is created (on a SafeNet HSM it is always possible to create at least one Partition -- more may be possible, depending upon the configuration that you initially purchased, or upon licensing/capability update packages that you might later choose to purchase and apply). Again, Partition backup is an opportunity to create more duplicate black PED Keys, or to cause a newly-created Partition to share an authentication secret that is already used on other HSMs' Partitions.

You will also require additional PED Keys if you decide to use the MofN security feature.

The numbers of PED Keys that you will need for your situation are discussed in much greater detail at ["How Many PED Keys Do I Need?" on page 232](#).

Physical Identification of PED Keys

This section is a few suggestions for your handling of PED Keys. Naturally you should be guided primarily by your organization's security policies.

As indicated above, you might wish to physically mark your PED Keys, in order to help keep track of them. Colored, blank tags are suggested, in addition to the provided stickers, though you could use any identifier that does not interfere with the operation of the PED Key. At a minimum, in an operational environment, you should have at least one working set and one full backup set, and a way to tell them apart.

If multiple personnel will need access to the HSM, you might provide duplicates of some or all PED Keys that are associated with a particular HSM. It would be helpful to number them, or to write the name or title of the person who will hold each duplicate, to ease tracking. Your organization's security policy might have requirements in that regard.

If you have multiple HSMs or groups of HSMs in your organization, a thoughtful labeling convention can ease the task of tracking and differentiating the various PED Keys and key-holder personnel.

If you invoke the optional MofN security feature (see ["Using MofN" on page 243](#)), you could have multiple sets of several PED Keys (containing the secret splits for SO, or for the Partition Owner, or for Domain, etc.) that might require unique visible identification. Possibly one person might be the designated holder of MofN secret shares belonging to more than one HSM in your company. If that person is carrying several PED Keys, it would be convenient to see, at a glance, which PED Key belonged to which MofN set so as to avoid making accidental bad login attempts due to mix-ups of PED Keys.

For example, if each department in your company had a SafeNet HSM, and you were using MofN feature, your key tags might be labeled something like:



So this would be Security Officer (SO) key-share number 4, of a 5-key MofN set that requires at least three key-holders to be present to unlock the administration functions of that HSM in the Accounts Receivable department. You might prefer to not mention the "N" quantity, so that an attacker would not know how many more he/she needed to acquire.

Alternatively, you might use something obscure like:



which could be a code representing a more descriptive entry that you would keep in a log book or in a database. Either way, by looking at the tag you can quickly find out which of various PED Keys you are currently holding.

Obviously, these are just basic suggestions, and you can use any identifying scheme that works for you.

Using PED Keys

This is described in detail at ["How to Use a SafeNet PED" on page 204](#).

Briefly, when you perform an HSM operation that requires a PED Key, you should already have the PED connected to the HSM or appliance.

When the command is issued, the system tells you when to look to the PED.

The PED prompts you when to insert various PED Keys, appropriate to the task. When prompted, insert the indicated PED Key into the connector at the top of the PED, immediately to the right of the PED cable connection, then respond to further instructions on the PED display, until control is returned to the administrative command-line.

Compare Password and PED Authentication


	Password-authenticated HSM	PED-authenticated HSM
Ability to restrict access to cryptographic keys	<ul style="list-style-type: none"> knowledge of Partition Password is sufficient for backup/restore, knowledge of partition domain password is sufficient 	<ul style="list-style-type: none"> ownership of the black PED Key is mandatory for backup/restore, ownership of both black and red PED Keys is necessary the Crypto User role is available to restrict access to usage of keys, with no key

	Password-authenticated HSM	PED-authenticated HSM
		management <ul style="list-style-type: none"> option to associate a PED PIN (something-you-know) with any PED Key (something you have), imposing a two-factor authentication requirement on any role
Dual Control	<ul style="list-style-type: none"> not available 	<ul style="list-style-type: none"> Mof N (split-knowledge secret sharing) requires "M" different holders of portions of the role secret, in order to authenticate to an HSM role - can be applied to any, all, or none of the administrative and management operations required on the HSM
Key-custodian responsibility	<ul style="list-style-type: none"> linked to password knowledge, only 	<ul style="list-style-type: none"> linked to partition password knowledge, linked to black PED Key(s) ownership
Role-based Access Control (RBAC) - ability to confer the least privileges necessary to perform a role	roles limited to: <ul style="list-style-type: none"> Auditor HSM Admin (SO) Partition Owner 	available roles: <ul style="list-style-type: none"> Auditor HSM Admin (Security Officer) Domain (Cloning / Token-Backup) Secure Recovery Remote PED Partition Owner (or Crypto Officer) Crypto User (usage of keys only, no key management) for all roles, two-factor authentication (selectable option) and MofN (selectable option)
Two-factor authentication for remote access	<ul style="list-style-type: none"> not available 	<ul style="list-style-type: none"> Remote PED and orange (Remote PED Vector) PED Key deliver highly secure remote management of HSM, including remote backup

What is a PED PIN?

For three-factor authentication, a PED PIN is "something you know", and is associated with "something you have", the PED Key (this is termed "three-factor" because you must

- login to the password-protected [1st factor] admin session before you can invoke the HSM SO or User,

- provide a physical PED Key [2nd factor]  and

- input the optional PED PIN [3rd factor]).

A PED PIN is an optional additional authentication layer (It is optional only for the first PED Key imprinted at initialization time - if you choose to have some duplicates made of that PED Key, then they all get the flag for PED PIN [or no PED PIN if that's what you chose] that you gave for the first key) for any of:

- the HSM Admin or SO (blue PED Key) or,
- the Partition Owner or Crypto Officer (black PED Key)
- the cloning Domain (red PED Key)
- the Remote PED Key (orange PED Key)
- the Secure Recovery Key (purple PED Key)
- the Audit key (white PED Key).

The secret that unlocks the HSM is the PinKey.

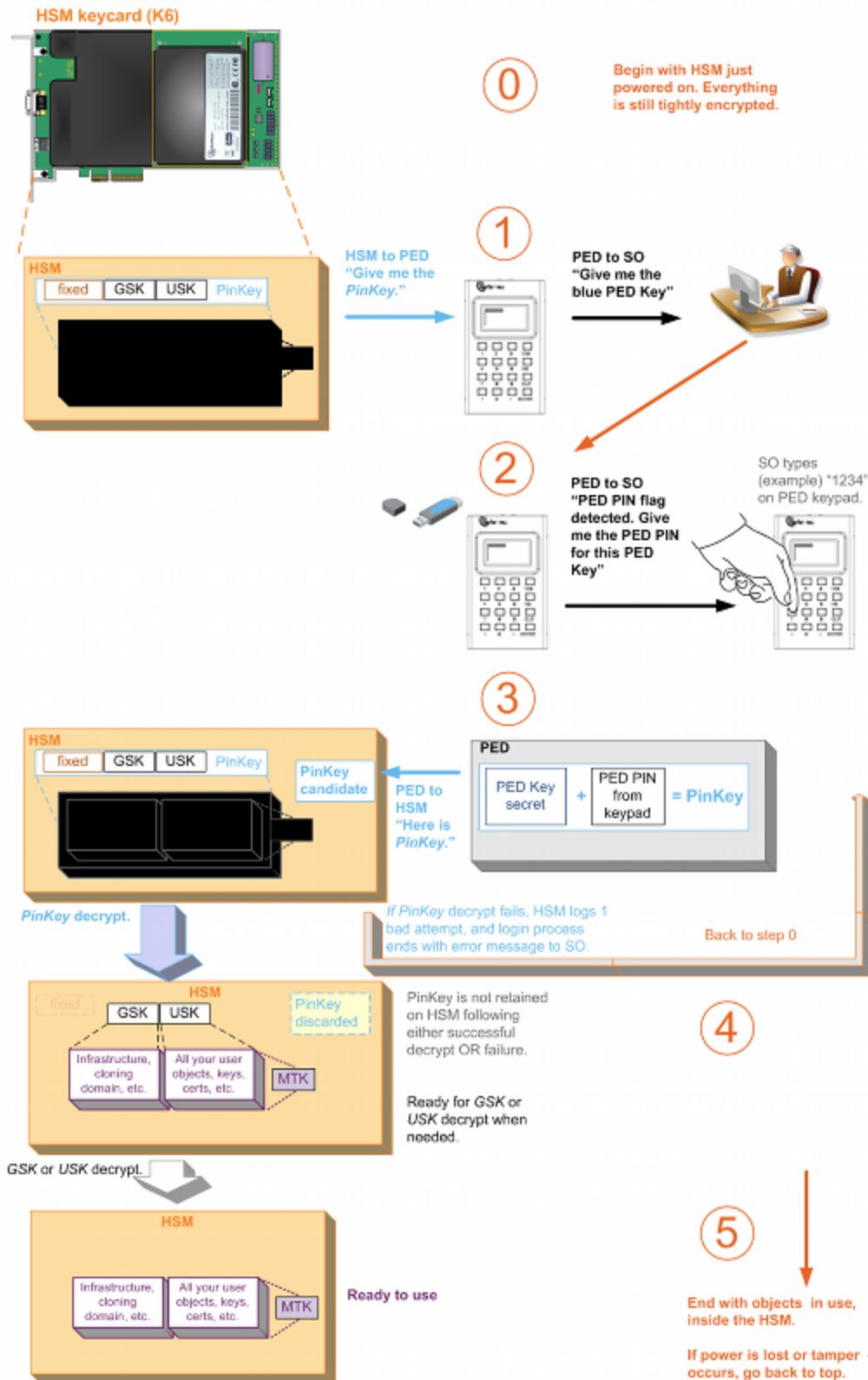
In Password authenticated HSMs, the PinKey is the text password that you type at a keyboard.

In PED authenticated HSMs, the PinKey is the secret that the HSM receives from the PED when the HSM calls for authentication.

But what is it?

A PED PIN is a sequence of digits that you type in, at the PED keypad, which is combined with the secret stored on the key, and the resulting combined PinKey is sent to the HSM. The combined secret-and-PED-PIN is what the HSM recognizes as its unlocking secret.

HSM – PED interaction, one PED PIN



If, for example, you are initializing an HSM and not re-using any existing secret on the PED Key that you present (or it's a blank key), then during the process, the SafeNet PED prompts you to provide a PED PIN. (see below)

How to invoke/require a PED PIN with an HSM

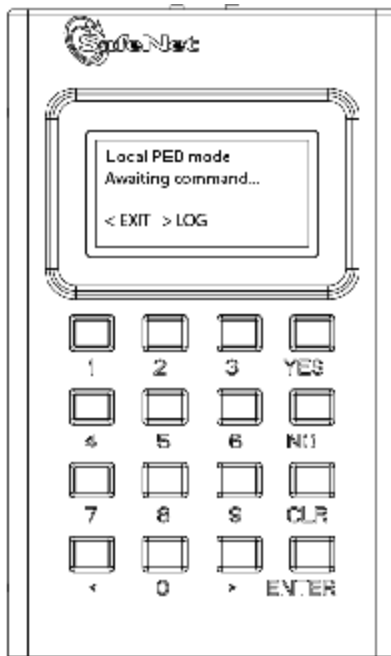
At the SafeNet PED prompt:

Enter a new PED PIN

If you want a PED PIN:

- enter 4 to 48 digits via the SafeNet [PED keypad](#) and press [Enter] (you are prompted to enter the PED PIN again, to confirm)

Note: **do not use zero for the first digit**



(When the leading digit is zero, the PED ignores any digits following the exact PED PIN. Thus an attacker attempting to guess the PED PIN must get the first digits correct, but does not need to know the exact length of the PED PIN. If the PED PIN is started with any digit other than zero, extra digits are detected as an incorrect attempt. This is not considered a real vulnerability since any attacker

- must have physical possession of the PED KEY,
- must have physical access to the HSM and PED, and
- gets only three tries to guess correctly, before the HSM is zeroized.

However, since we noticed it, we thought we should mention the slightly different function when the first PED PIN digit is zero.)

- the PED PIN must be the same across multiple HSMs
- SafeNet PED combines your PED PIN with the random PIN from the (blue or black) PED Key and presents that combination to the token as the authentication for HSM Admin or the Partition Owner (or Crypto Officer) respectively.
- PED PIN digits are not echoed to the PED screen; instead, whatever you type is masked by asterisk (*) characters.

If you don't want a PED PIN:

- just press [Enter] on the SafeNet PED keypad (signifying 0 digits); you are prompted again, to confirm.

The PinKey is the secret on the PED Key, combined with the PED PIN. The PED PIN is not recorded - it is a transformation that you perform on the PED Key secret to convert it into the PinKey. Therefore, the PED PIN is separate and distinct from the HSM SO authentication secret (or the User/Owner/Crypto Officer authentication, etc.) contained on the PED Key. It is optional to create a PED PIN (as an extra layer of authentication security) when you initialize an HSM, but once a PED PIN is invoked, it is then required every time you authenticate to the HSM. That is, if you opt to not create a PED PIN at initialization time (or Partition creation time for the black PED Key), then you never use PED PINs, but if you do create a PED PIN at initialization time, then you are "stuck" with the requirement until the next time you wipe the contents (zeroize) and re-initialize. The point to make is that the PED PIN option is there if your policy and situation require the additional security, but you don't need to invoke the extra layer if you don't require it.

The choice to invoke PED PIN for a particular PED Key function [blue SO key, black User/Owner key, red Cloning Domain key, orange Remote PED key, white Audit key, or purple Secure Recovery key] is independent of the other types of PED Key.

For example, if (at initialization time) you decide to have a PED PIN for the blue (SO) PED Key, then that PED PIN is thereafter required when you use blue PED Keys with that HSM (until you initialize again), but you do not need to use PED PINs with the black and red PED Keys if you don't wish to do so. Similarly, you might choose to invoke PED PIN for the red PED Key, but not for the blue or black PED Keys.

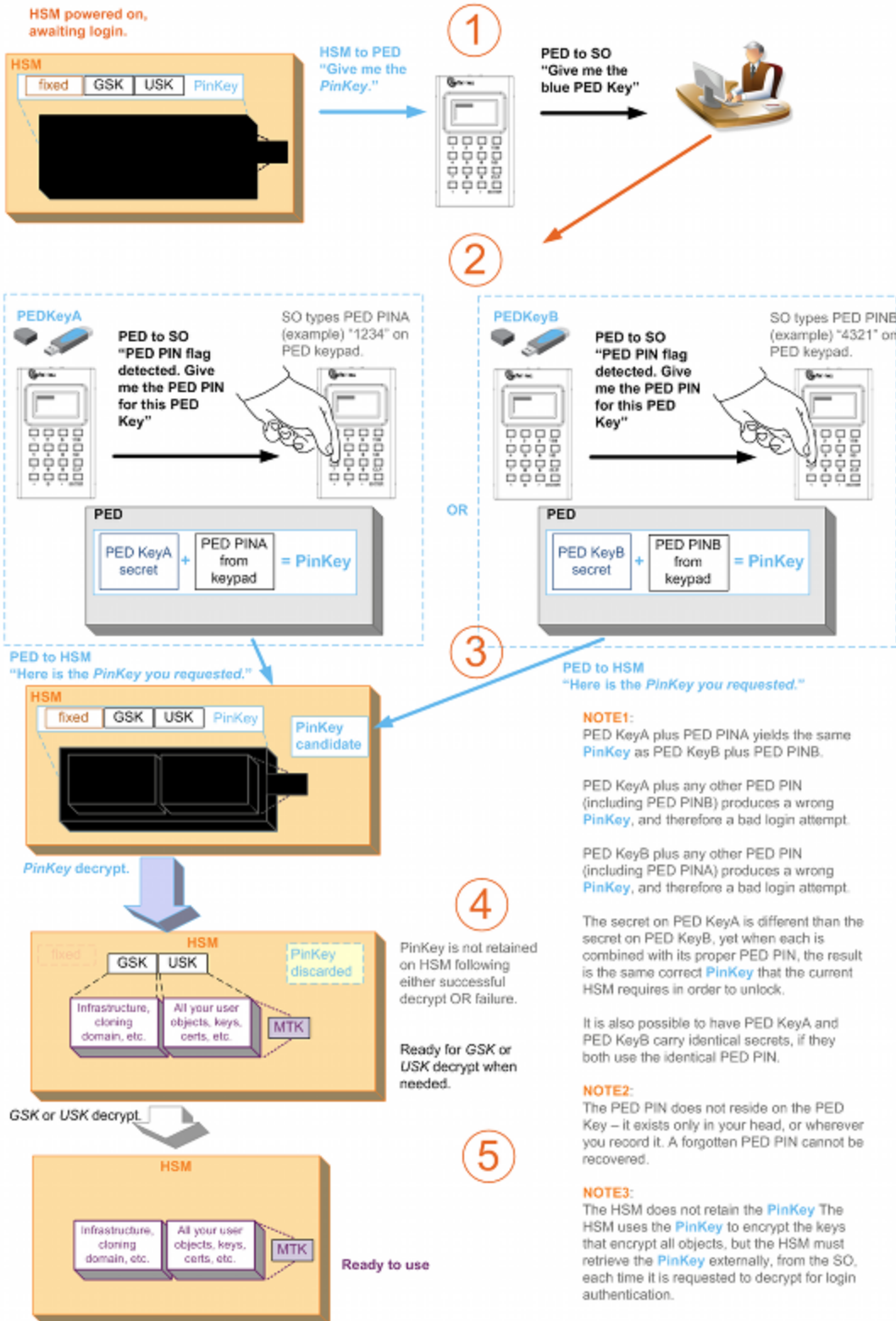
Here are possible combinations if you have two HSMs H1 and H2, and any of several initialization-time choices regarding PED PIN. What is important to unlock the HSM is the secret that is imprinted on the HSM, so in the following table we will call that secret H1SO or H2SO. We will call the secret contained on the PED Key K1SO or K2SO.

Configuration	SO Authent Secret on HSM	What You Need to Unlock HSM	PED Keys Interchangeable?
Different blue PED Key Pinkeys H1SO and H2SO K1SO does not equal K2SO	H1SO not same as H2SO	The correct PED Key for the current HSM	No
Two identical blue PED Keys, no PED PINs, so PED Key secret is the PinKey secret, which is the same on both K1SO = K2SO and H1SO = H2SO	H1SO secret identical to H2SO	Either PED Key; both are correct for either HSM	Yes
Two identical blue PED Key(s), same PED PINs so PED Key secret is the same on both (K1SO = K2SO) and therefore PinKey secret is the same for both, to yield H1SO = H2SO	H1SO secret identical to H2SO	Either PED Key plus the one PED PIN; both are correct for either HSM	Yes
Two blue PED Key(s), different PED PINs for both HSMs, but PED Key secrets are also different (K1SO does not equal K2SO) such that PED Key1 plus PED	H1SO secret identical to H2SO	Either PED Key plus the correct PED PIN for just that PED Key; both are correct for either HSM.	Yes, but the PED PINs are not.

Configuration	SO Authent Secret on HSM	What You Need to Unlock HSM	PED Keys Interchangeable?
PIN1 together generate the same PinKey secret as PED Key2 plus PED PIN2 - H1SO = H2SO		Either PED Key with the PED PIN for the OTHER PED Key is a bad login attempt.	

Here is a drawing of HSM PED authentication with two PED PINs.

HSM – PED interaction, duplicate PED Keys, two different PED PINs



Must I Use a PED PIN?

If a PED PIN has been set for a PED Key and an HSM, then you must always provide that PED PIN when using that key (or any duplicate of it) to login to that HSM. If you duplicate a PED Key, what you are duplicating is the secret that was originally imprinted on the PED Key, plus the state of a flag. The flag is an instruction to the PED to "prompt for a PED PIN"... or not.

If you choose, at initialization, not to invoke a PED PIN (that is, if you just press [Enter] without typing any digits on the keypad), then the flag is not set on the PED Key, and the secret on the PED Key matches exactly the secret in the HSM. Any duplicates that you make of the first PED Key will also have the flag unset. Whenever you use any of those PED Keys (original or duplicates) the PED checks for the state of the flag, finds it not set, and simply decrypts and sends the unmodified stored secret to the HSM, without prompting for PED PIN.

Should I Use a PED PIN?

That is up to you and your organization's security policy, but security procedures should never be more complicated than your requirements dictate.

Consider also if your security policy requires regular changes to passwords and other authentication. Your personnel would need to remember new PED PINs with each change cycle. If people are asked to remember too many passwords/PINs or asked to change them too often, they begin writing them down, which is itself a potential security issue.

What If I Change My Mind?

You can remove the requirement for a PED PIN by using the 'hsm changePw' command. A new secret is generated on the HSM, and is imprinted onto the PED Key (you are asked if you want to overwrite the existing data and you say YES). You are given the opportunity to add a PED PIN and you just press ENTER on the PED keypad to decline a PED PIN.

During the PED operation, you are given the opportunity to imprint additional keys with the new secret that doesn't include a PED PIN. You can use that opportunity to imprint additional new, blank PED Keys, or to overwrite PED Keys that are already imprinted with the old secret¹.

Note: This action must be performed on all the PED Keys [duplicate PED Keys] associated with that HSM.



If you have a group of HSMs that share the same authentication secret (meaning they can all be unlocked by the same PED Keys [group PED Keys, see below]) then you must keep one unchanged PED Key until you have logged in and performed the 'hsm changePw' command on all the HSMs in that group.

Similarly, if you decide to increase the stringency of your security, you can use the 'hsm changePw' command to change the secret on your PED Keys and HSM(s) and at the same time, add PED PINs. Again, if you make such a change, consider doing it on all copies [duplicates] of the PED Key, and on all HSMs that shared the old PED Key authentication data.

Alternatively, you could leave some PED Keys with the old secret and leave some HSMs with that same secret. The result would be two groups of HSMs and associated PED Keys that could not be interchanged (for authentication). In other words, you could use that technique to split a group of HSMs.

¹[which is now invalid for the current HSM]

Does that apply to the other PED Key colors?

Not all.

- It does apply to the black PED Key - use the lunash command `partition changePw`. This change is non-destructive to the HSM partition or its contents.
- For the purple PED Key, you must generate a new SRK (lunash command `hsm srk keys resplit`). This requires that you have the old/current SRK to begin, and that you provide a different PED Key to receive the new Secure Recovery Vector. The PED does not allow you to overwrite the current purple PED Key. This change is non-destructive to the HSM or its contents.
- For the orange PED Key, you can use the lunash command `hsm ped vector init` to create a new Remote PED vector on the HSM and on the current orange PED Key, or you can import a different RPV from a different orange SRK and imprint that RPV onto the HSM in place of the current one. This change is non-destructive to the HSM or its contents.
- However, you **cannot** change an HSM Domain without a hard initialization of the HSM (destroys all contents), and you cannot change a partition Domain without deleting the current partition and creating a new partition, which deletes all contents of the current partition.

What is a Shared or Group PED Key?

Visit this topic for an additional, interesting concept that might be important to you when imprinting and using PED Keys:

["Shared or Group PED Keys" on page 226](#)

What else do I need to know?

Here is a re-cap of what happens when you initialize.

The HSM, when told to initialize, turns over control to the PED, which immediately asks "Do you wish to reuse an existing keyset?". If the answer is NO, the HSM creates a new secret which will reside on both the HSM and the key (or keys) that is (or are) about to be imprinted. If the answer is YES, then the HSM does not create a new secret and instead waits for one to be presented via the PED.

The secret (whether from the current HSM or from an inserted PED Key, previously imprinted by another HSM) is presented to the PED.

If you are using a new secret [you answered "NO" to the "...reuse..." question], the PED prompts for a PED PIN, and you provide either a string of digits via the keypad (a PED PIN), or no digits and just a press of "Enter" (no PED PIN).

If you are reusing an existing secret, then the PED takes that from the presented PED Key (including any PED PIN, which you must know and provide when prompted) and presents that to the HSM.

At this point, either the secret from the HSM is written to the PED Key, or the secret from the PED Key (possibly combined with a PED PIN is written to the HSM. If a PED PIN exists, then the secret on the PED Key is modified from the original by combination with the PED PIN, and that modified secret is imprinted upon the HSM - only the unmodified secret on the PED Key, combined with the PED PIN can reproduce the secret that the HSM expects.

The PED asks if you will be duplicating this key. Each duplicate can have a different PED PIN (or no PED PIN).

The same pattern applies for any of the secrets - SO (blue), User/Owner (black), Domain (red), RPK (orange), SRK (purple).

Best Practice

When you initialize a PED-authenticated HSM (or create a partition, or perform any action that imprints a PED Key), and you choose to associate a PED PIN with the PED Key secret, you must ensure that the PED PIN will be remembered when it is needed. That normally means writing it down on paper or recording it electronically. This, of course represents a security risk. But it would equally be a security risk to not record the PED PIN and then be unable to remember it.

Before you tuck that yellow-sticky with the PED PIN into your safe, TRY it once, to verify that you did set the PED PIN that you think you set (or that you correctly recorded what you actually set).

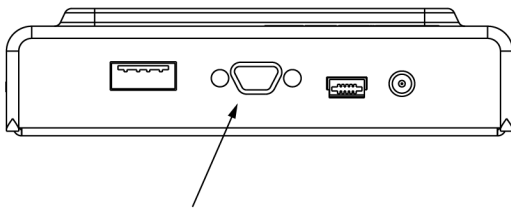
In the case of a red key, that would mean you would need to attempt a cloning or backup/restore operation before storing your record of the PED PIN.

How to Use a SafeNet PED

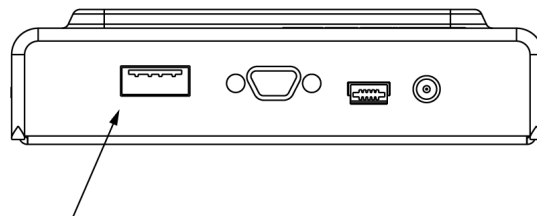
The PED, when used locally, derives its power from its connection to a SafeNet HSM.

To use the SafeNet PED:

1. Connect the SafeNet PED to the PED connector on the SafeNet HSM, using the supplied cable.



2. SafeNet PED performs its self-test and briefly displays its firmware version. When the display shows "SCP mode" and "Awaiting command..." Luna PED is ready to use with your Luna HSM.
3. When an activity on the server requires SafeNet PED operation, the SafeNet PED display changes, to prompt you to insert a PED Key, or to perform some other action.
4. If a PED Key is requested, remove any Key that is currently inserted (if any), and insert the requested PED Key



into the USB connector slot

on the right-hand top side of the SafeNet PED (immediately to the right of the cable connection).

5. When the Key is fully inserted, the LED in the key housing comes on.
6. Press [ENTER] on the keypad, and watch for further prompts on the display.



Note: The SafeNet PED display returns to “Awaiting command..” when the current sequence of PED operations is finished.
 “Awaiting command..” on the SafeNet PED means that control has been transferred back to the HSM.

SafeNet PED Keypad Functions

Key	Function
[CLR] or [Clear]	- Clear the current entry, such as when inputting a PED PIN - wipes the entire entry. - * Reset the PED - the key is held down for five seconds. Useful if a PED operation has timed out.
[<]	- Backspace; clear the most recent digit that you have typed on the PED, such as when inputting a PED PIN. - "Back"; navigate to a higher-level menu in the PED.
[>]	- Shows most recent PED actions (since being in Local or Remote Mode)
Numeric keys	- Select numbered menu items. - Input PED PINs.
[Yes] and [No]	- Respond to Yes-or-No questions from the PED.
[Enter]	- Confirm an action



Note: * Pressing (and holding) [CLR] causes reset only if the PED is engaged in an operation or is actively prompting you for action.
 Pressing [CLR] has no effect in the main menu, in the Admin Mode menu, or when "Awaiting command..."

PED Display and "Slot 03"

Following a successful startup sequence, the display screen of the PED shows a menu (use the keypad keys to navigate), or it shows an activity screen and prompt. The activity screen usually has the word "Slot" and a number on the first line.

If you connect the PED locally to an HSM, the PED shows Slot 03. This is normal behavior due to legacy compatibility where two PKCS#11 slots were originally reserved for removable PCMCIA-card HSMs. So a directly connected PCI-E HSM or USB HSM is recognized as slot 3.

Remote PED mode does not show a number, because there is no physical slot to enumerate when connected to a Remote PED server instance on the host.

SafeNet PED Interaction

Go to ["Interaction between the HSM and the PED"](#) below to read about using the SafeNet PED with your HSM.

Remote PED

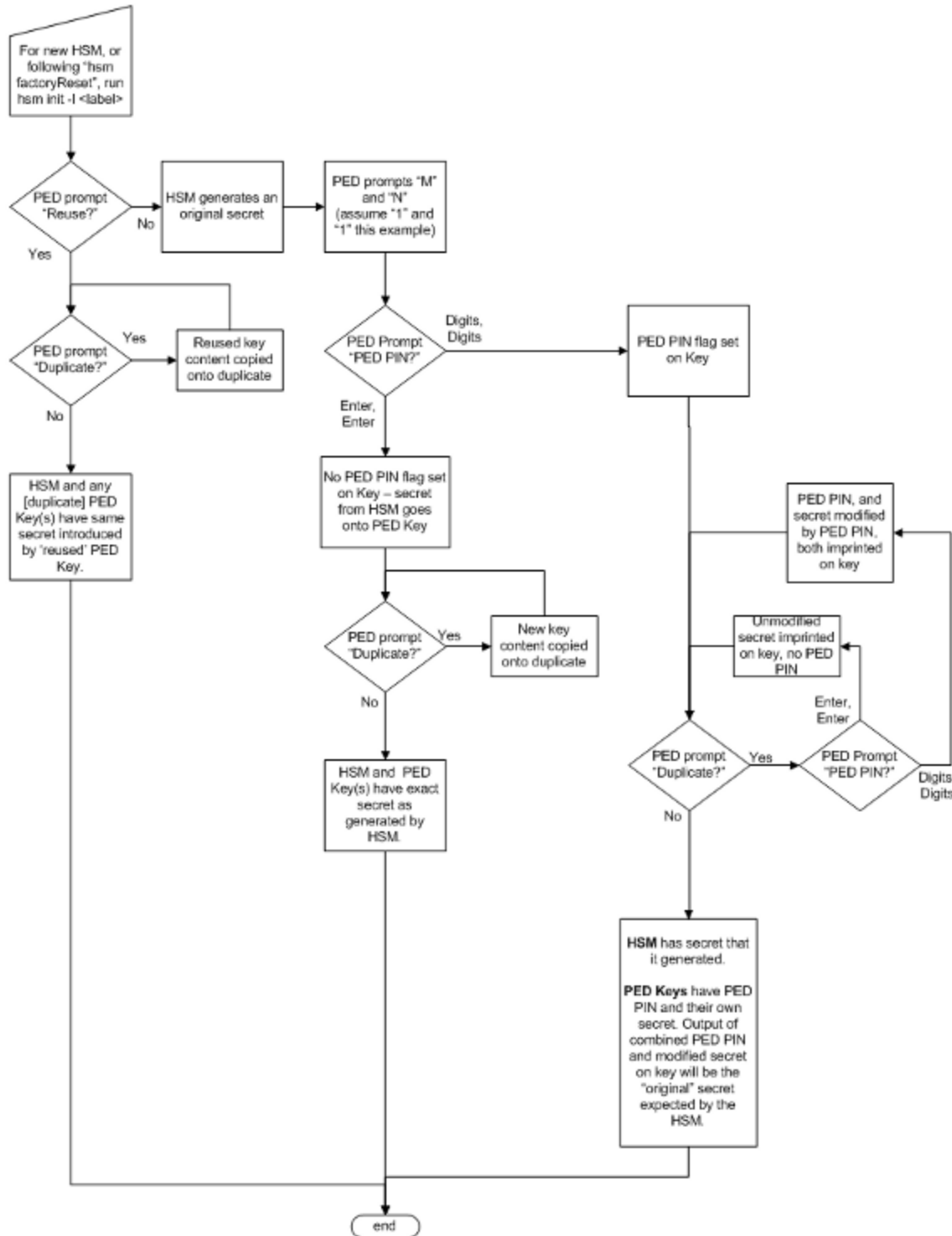
Got to "[About Remote PED](#)" on [page 270](#) to read about using a SafeNet PED remotely from your SafeNet HSM, via PedServer and PedClient.

Interaction between the HSM and the PED

(This page is background information that might help make some operations more obvious.)

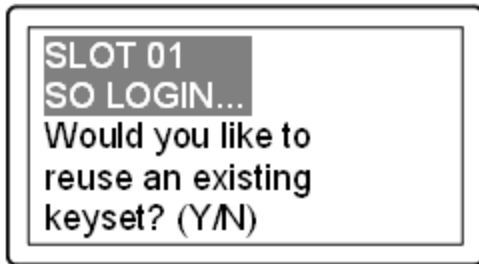
After the first-ever SafeNet HSM, all succeeding generations have included both password-authenticated and PED-authenticated variants. This page describes how the current-generation PED-authenticated HSMs (firmware 6.x) interact with SafeNet PED and PED Keys, particularly during initialization - a time when important decisions are made. Other pages describe the PED and PED Keys. This is more about flow.

The diagram shows how the components are affected as you make choices during an initialization [this sequence depicts events and choices if you initialize a new, factory-fresh HSM, or one on which you have recently run "hsm factoryReset"; as well the process would be very similar for creation of a partition]. This flow depicts the SO / HSM Admin secret, but the interactions for other secrets follow the same pattern.



When you issue the "hsm init" command at the command-line, the HSM generates a secret, then turns over control to the SafeNet PED.

Reuse? (a.k.a. Group PED Key)



The first question from the PED is whether you wish to "Reuse" an existing SO / HSM Admin authentication secret (the same logic applies to the other PED Keys, so we use just the blue key in this example). This means that you have a blue PED Key from another HSM, or you have a blue PED Key from a previous initialization of this HSM. The PED is asking if you wish to import the secret from that key onto the HSM. The options at this point are:

- a) you have only fresh blank PED Keys that have not been used previously with any HSM (No - do not reuse)
- b) you have a previously used PED Key, but the secret it contains is not one you wish to preserve or re-use (No - do not reuse)
- c) you have a previously used PED Key, with a secret from this HSM, and you don't mind reusing it (Yes - reuse)
- d) you have a previously used PED Key, from another HSM, and you wish to reuse it so that the blue key can unlock both the current HSM and the other HSM. (Yes - reuse)

These options also apply to any other key color when they are being imprinted. If you elect to reuse the content of an existing key, then the secret that the current HSM generated is discarded, and the secret from the reused PED Key overwrites onto the HSM. This ensures that the PED Key and the HSM have the same authentication secret, and the key can unlock the HSM. If the secret on the key was from another HSM that is still operational, then the PED Key has become a "group PED Key" that unlocks the equivalent aspects of both HSMs. In this manner, you can include as many HSMs as you wish in a group. [Note that this "group" of HSMs is related only by the convenience of being able to use one PED Key to unlock any of them. This "group" concept is not the same as (say) the HA Group concept for high availability.

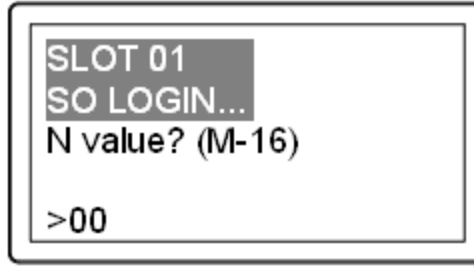
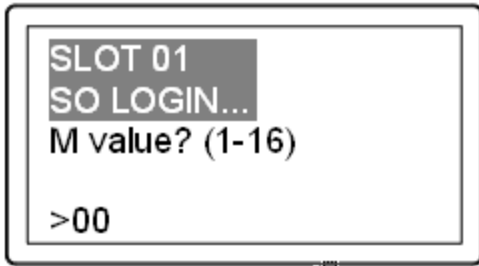
The HSM slots that form an HA group interact with their client(s) via a virtual HSM slot, such that any of the real HSM slots behind the HA group is interchangeable and can be swapped in and out as needed. But members of an HA group do not need to be members of a PED Key group. In an HA group, any or all of the members could have the same or different authentication secrets, without affecting the HA function. Only the cloning domain must be identical across all HA group members.]

If you choose to **not** reuse the content of an existing key, then the secret that the current HSM generated is copied onto the key that is currently inserted into the PED (after the PED verifies multiple times that this is what you wish to do). This ensures that the PED Key and the HSM have the same authentication secret, and the key can unlock the HSM. If the PED Key previously had a secret for another HSM, it no longer does. The PED Key can now unlock the current HSM but is useless with the previous HSM.

Note also that your organization's security policies govern whether you can allow multiple HSMs or HSM partitions to be unlockable by the same PED Key.

MofN?

The second question from the PED would ask for M and N values,



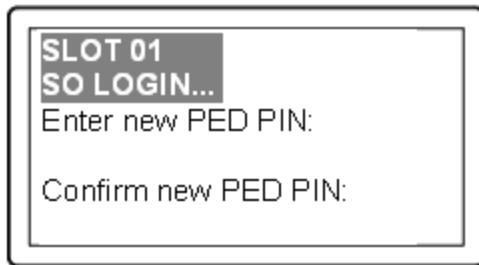
so that you could set

up MofN split-secret, multi-person access control. However, that option would greatly complicate this explanation, so we will assume that you choose M=1 and N=1, which means "no MofN invoked".

If you wish to see an explanation of how MofN works on the HSM, go to "[HSM Authentication Model with MofN and No PED PINs](#)".

PED PIN?

The PED provides the opportunity to add an additional layer of authentication security to the handling of the current secret.

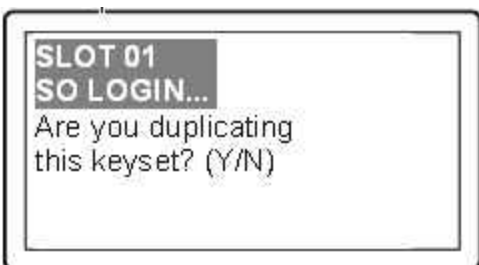


A PED PIN is a numeric secret typed on the PED keypad. If you just press enter, no PED PIN is created, and therefore no PED-PIN flag is set on the current PED Key. If you do type in some digits on the PED's keypad, then that sequence becomes a PED PIN, a numeric password that must be typed whenever you wish to use that key in future. Whatever your response, the PED asks you to confirm by typing it in again, before proceeding to the next question.

If you wish to see an explanation of how PED PINs work on the HSM go to "[HSM Authentication Model with a PED PIN](#)" or "[HSM Authentication Model with Multiple PED PINs](#)".

Duplicate? (make backups)

The next question from the PED is whether you wish to duplicate the current PED keyset.



[The word "keyset" is used because you could have chosen to invoke MofN, splitting the (in this case) HSM secret across several blue keys, rather than just the one in this example. That is, a "keyset" can consist of one key, containing a complete secret, or multiple keys, each containing a portion of that secret.]

In general, it is a good idea to have several PED Keys with the HSM secret duplicated, so that you can have on-site and off-site backups, and to meet your other operational considerations.

The first opportunity to make copies is at initialization time, as the PED always asks this question during the process. Your answer to the "duplicate" question determines the end of the process for the current PED Key secret.

Again, your security policies dictate how many backup copies - or other operational copies - of a PED Key should be made, and how they should be handled and maintained.

How it was - versus - how it is today

Customers who are familiar with our legacy HSM products, and who are now preparing to use SafeNet Network HSM 6.x (a firmware 6.x HSM) would observe that much of the concept and action is similar to the previous generation, but with a few important differences, described below. This would be especially important for customers who are migrating keys and HSM contents from older HSMs to the current generation.

Differences in function are driven to a considerable extent by the updating of the (optional) MofN, split-secret, multi-person access control model.

Historical

In HSM firmware 4, the MofN concept was of a separate, self-contained single secret (on green keys, and no PED PIN), so all the other PED Key colors were just one secret each, which was a simple model that allowed certain possibilities and precluded others.

In that older model, if a PED PIN was created, it existed only in your head ("something you know"), and was a transformation that you applied to the secret on the key ("something you have"), to make it into the secret on the HSM. In that model, it was *not* possible to have more than one PED PIN for (say) the SO secret on HSM1. However, it was possible to use that same key for another HSM (2) with a different PED PIN, because the secrets on the two HSMs didn't have to match.

All that was needed was that whatever was on the blue key could be reliably transformed into what HSM1 wanted, and could also be transformed into whatever HSM2 wanted, by typing something on the keypad.

You could minimize the number of blue keys, while still ensuring that HSM1 and HSM2 had effectively different secrets – as long as you trusted that HSM1's SO and HSM2's SO were not going to talk to each other. But any duplicate blue keys were, indeed, exact duplicates. It was the HSMs in a group that had different secrets, not the keys. The same idea applied to the black keys.

(Key #1 + PED PIN #1) or (Key #2 + PED PIN #1) = Success on HSM1,

(Key #1 + PED PIN #2) or (Key #2 + PED PIN #2) = Success on HSM2

(Key #1 + PED PIN #1) or (Key #2 + PED PIN #1) = Failure on HSM2

(Key #1 + PED PIN #2) or (Key #2 + PED PIN #2) = Failure on HSM1

Modern

In HSM firmware 6 (SafeNet PCIe HSM 5, SafeNet Network HSM 5, SafeNet USB HSM), the new PED-mediated MofN-per-key-color model required some re-engineering. Additional infrastructure was needed, which makes this model incompatible with the previous method.

Functionally, in the current model, it doesn't matter whether you choose M and N to be one (feels like no MofN) or you choose M and N to be greater than one (invoking secret-splitting) – the infrastructure is there, regardless.

One result is that the HSM takes on additional responsibility for validating splits (even if there's only the one...), and the PED Key data now has a direct relationship to the PED PIN (which is part of the validation done when the PED Key is entered). Therefore, a "duplicate" is now a slightly fuzzier concept. Each duplicate PED Key can be given a different PED PIN (or none), and can still unlock the same HSM1. BUT, if you now make a group of HSMs by initializing a second HSM (HSM2) with the same basic secret (by imprinting the new HSM from one of the duplicate PED Keys), you must use the correct PED PIN for the Key used – any other choice will fail validation. The result is that the second HSM uses the same secret as the first - which is different from the firmware 4 case.

You can optionally have split each secret (M and N greater than 1 when you initialized HSM1), which just makes the combinations more interesting to track without a good set of notes, but that doesn't change the concept... merely adds a layer.

In the following table, we illustrate your interactions with the PED as you initialize an HSM or create a partition, with a fresh secret (not reused), and then create two duplicates of the PED Key, each with a PED PIN different from the original and from each other, yet all three will unlock that HSM or that partition - to simplify this exercise, we ignore MofN. Assume that all keys are fresh blanks.

HSM1 PED prompt	Original key, No PED PIN (your action)	First duplicate key PED PIN "1234" (your action)	Second duplicate key PED PIN "4321" (your action)
"Do you wish to reuse an existing keyset" (creating new PED Keys during initialization)	Press [No]	n/a	n/a
Insert...	(insert a new key)	-	-
Enter new PED PIN / Confirm new PED PIN	Press [Enter]	n/a	n/a
"Are you duplicating this keyset?"	Press [Yes]	-	-
Insert...	-	(insert a new key)	-
Enter new PED PIN / Confirm new PED PIN	-	Type "1234" and press [Enter]	
"Are you duplicating this keyset?"	-	Press [Yes]	-
Insert...	-	-	(insert a new key)
Enter new PED PIN / Confirm new PED PIN	-	-	Type "4321" and press [Enter]
"Are you duplicating this keyset?"			Press [No]

All three PED Keys have different PED PINs, but any one of them can unlock this HSM. The combination of any of those PED Keys, with its own PED PIN will produce the same secret for the HSM.

To round out the parallel concept that finished the firmware 4 discussion above, any duplicate blue keys are not necessarily exact duplicates, they just all contain a way (PED PIN secret) to get back to the same output secret. But in this model (firmware 6), if you want to use the same blue keys for several HSMs, all the HSMs must have exactly the same blue (SO) secret, because a duplicate of any blue key CAN have whatever PED PIN you choose (or none) but must still be able to generate the correct secret.

(Key #1 + PED PIN #1) or (Key #2 + PED PIN #2) = Success on HSM1

(Key #1 + PED PIN #1) or (Key #2 + PED PIN #2) = Success on HSM2

(Key #1 + PED PIN #2) or (Key #2 + PED PIN #1) = Failure on HSM1

(Key #1 + PED PIN #2) or (Key #2 + PED PIN #1) = Failure on HSM2

Restating the "obvious"?

Some important implications of the above explanations deserve restating.

- If you choose to NOT reuse a secret from an existing PED Key, then the HSM and the new set of PED keys being created by initialization all receive secrets based on the secret that is newly generated by the HSM. This is how you ensure that no other HSM can be unlocked by the PED Key(s) that you are now associating with the current HSM. This exclusivity lasts as long as nobody initializes yet another HSM using the PED Key(s) that you just created for this current HSM. Reusing, or not, is chosen on a per-role basis, so that **some** PED Key secrets on an HSM could be shared with other HSMs, while others are not.
- It is crucially important to always control your PED Keys. Know where they are, know how many there are, and know who is handling them.
- If you choose to reuse a pre-existing secret, then the secret that the HSM generates at the start of initialization is discarded, in favor of the imported secret [the secret that you accept from an existing imprinted PED Key when you say [Yes] to the PED question "Would you like to reuse an existing keyset?"]. This is how you make group PED Keys that can unlock more than one HSM.
- The PED PIN, if you invoke one, exists only in your head¹ not on the PED Key - it is the combination of the secret on the key, plus the PED PIN for that key, that produces the secret that the HSM sees (and requires).

An additional question that is sometimes asked, about reuse and duplicates...

- You can "reuse" an existing secret only for the same type of secret that is currently being requested by the HSM and the PED. That is, if you say [Yes] to "Would you like to reuse an existing keyset" while preparing to set the HSM's Security Officer (SO) secret, then you must present a valid, imprinted blue PED Key. Any other color, or a blank key, is rejected as a source to reuse. A Crypto Officer (black key) secret cannot be "reused" as an HSM SO (blue key) secret. Nor can a Domain (red), or RPK (orange), or SRK (purple key) secret. "Reuse" is the opposite of overwrite. For the "reuse" option, with any PED Key secret, the matching kind of pre-existing secret is needed.

SRKs, the purple key secrets, are unique per HSM and are not reused, ever.

Duplicating PED keys

SafeNet PED has the ability to make copies of PED Keys, without the intervention of an HSM. All the PED needs is power.

¹[or wherever you write it down]

Duplicating PED Keys / Copying PED Keys

Insert any PED Key containing a secret that you wish to duplicate. The PED defaults to the local mode menu.

Press "<" to get to the Select Mode menu.

Press "4" for the Admin menu.

Press "1" for PED Key.

Press "1" again, for Login.

Press "7" for Duplicate. The PED reads the key that you already inserted, then prompts you:

Duplicate PED Key...

Insert target

PED Key.

Press ENTER.

When you press ENTER, the key in the slot gets the data that was read from the first key.

You can imprint as many new PED Keys as you wish.

Note that the PED does NOT prompt you for a PED PIN.

If the PED PIN flag was not set on the source key (the first key you inserted before invoking the Duplicate function), then the new copy also has that flag unset.

If the PED PIN flag was set on the original key, then that setting is automatically recorded on the duplicate. No HSM is involved in this PED-only transaction, so entering a PED PIN would have no effect in this case. Yet the correct PED PIN will be requested when you later use one of these duplicates to access the HSM.

This DIFFERS from the sequence when you are initializing and choose to make duplicates at that time - in that case you are prompted for PED PIN and can make several "duplicate" keys that have different PED PINS and yet unlock the same HSM. This method is called a "raw" duplication and works for every type of PED Key except a purple SRK.

Compare Duplication via PED Admin menu - versus - "Duplication" when initializing

	Requires HSM	Launched from command line	Prompt (option) to set PED PIN	"Copies" are identical	"Copies" unlock same HSM
"Duplicating" (creating new PED Keys during initialization)	Yes	Yes	Yes	Only if no PED PIN or if same PED PIN is repeatedly entered	Yes, as long as you know the correct PED PIN for the key you have
Duplicating "raw" key content via PED menu	No (only a power connection needed) Note: does not work for purple PED Key.	No	No	Yes	Yes, PED PIN is the same for all raw duplicates

Lost PED Keys or PED PINs, or passwords

Help! I have lost my blue/black/red/orange/purple/white PED Key or I have forgotten the password!

ANSWER-general (Passwords): Go to the secure lockup (a safe, an off-site secure deposit box, other) where you sensibly keep such important information, read and memorize the password. Return to the HSM and resume using your HSM(s).

ANSWER-general (PED Keys): Retrieve one of the copies that we (and your security advisor/consultant) always advise you to make, from your on-site secure storage, or from your off-site [disaster-recovery] secure storage, make any necessary replacement copies, using SafeNet PED, and resume using your HSM(s).

If you have lost a blue PED Key, someone else might have found it. Consider `lunacm:>changePw` or `lunash:>hsm changePw`, as appropriate to invalidate the current blue key secret, which might be compromised, and to safeguard your HSM with a new SO secret, going forward. HSM and partition contents are preserved.

But I don't have keys or secrets in secure on-site or off-site storage! What do I do?

ANSWER - blue PED Key or SO password : If you truly have not kept a securely stored written backup of your HSM SO Password, or for PED-authenticated HSM, your blue SO PED Key, then you are out of luck. If you **do** have access to your partition(s), then immediately make backups of all partitions that have important content. When you have done what you can to safeguard partition contents, then perform `hsm factoryReset`, followed by `hsm init` - this is a "hard initialization" that wipes your HSM (destroying all partitions on it) and creates a new HSM SO password or blue PED Key. You can then create new partitions and restore contents from backup. Any object that was in HSM SO space (rather than within a partition) is irretrievably lost.

ANSWER - black PED Key or Partition User password - legacy partitions: If you truly have not kept a secured written backup of your partition User Password, or for PED-authenticated HSM, your black partition User PED Key, then log into your HSM as SO, and perform `partition resetPw`. The `partition changePw` action is done by a partition owner who has the current credential and wishes to change it, so that one is not available to you now. The `partition resetPw` is done by the HSM SO when the current partition secret has been lost, or is compromised (perhaps by the unplanned departure of personnel). Select option 4 when you run the command.

```
lunash:> partition resetpw -partition mypar
```

Which part of the partition password do you wish to change?

1. change User or Partition Owner (black) PED key data
2. generate new random password for partition owner
3. generate new random password for crypto-user
4. both options 1 and 2
0. abort command

Please select one of the above options: 4

Luna PED operation required to reset partition PED key data - use User or Partition Owner (black) PED key.

'partition resetPw' successful.

Command Result : (Success)

```
lunash:>
```

**** Follow the PED prompts:

- a. press [No] when asked “Would you like to reuse an existing keyset? (y/n)”
- b. provide the M and N values of your choice ([1] and [1] if you don't want MofN)
- c. press [Yes] to overwrite the user key
- d. provide your choice of PED key PIN when prompted (or just press [Enter] if you do not wish to impose a PED PIN)
- e. press [Yes] when asked “Do you want to duplicate the keyset? (y/n)”
- f. write down the new random challenge from the PED screen (for best legibility, type it)

Now that you have the new partition authentication, you can change the PED-generated text challenge to something more to your liking via the partition **changePw** command, choosing option 3.

```
lunash:> partition changePw -partition mypar1

Which part of the partition password do you wish to change?

1. change partition owner (black) PED key data
2. generate new random password for partition owner
3. specify a new password for the partition owner
4. both options 1 and 2

0. abort command

Please select one of the above options: 3
> *****

Please enter the password for the partition:
>*****

Please enter a new password for the partition:
>*****

'partition -changePw' successful.

Command Result : 0 (Success)
lunash:>
```

ANSWER - red PED Key or HSM-or-Partition domain secret: If you have the red PED Key or the HSM-or-Partition domain secret for another HSM or Partition that is capable of cloning (or backup/restore) with the current HSM or Partition, then you have the domain that you need - just make a copy. Cloning or backup/restore can take place only between entities that have identical domains, so that other domain must be the same as the one you "lost".

If you truly have not kept a secured written backup of your HSM or partition cloning domain, or for PED-authenticated HSM, your domain PED Key(s), then you are out of luck. Any keys or objects that exist under that domain can still be used, but cannot be cloned or backed-up or restored. You have no fall-back, in case of accident. Begin immediately to phase in new/replacement keys/objects on another HSM, for which you DO have the relevant domain secret(s) or red PED Key(s). Ensure that you have copies of the red PED Keys, or that you have a written record of any text domain string, in secure on-site and off-site backup locations. Phase out the use of the old keys/objects, as you have no way to protect them against a damaged or lost HSM.

ANSWER - orange PED Key : You will need to generate a new Remote PED Vector on one affected HSM with `lunacm:>ped vector init` or `lunash:>hsm ped vector init` to have that HSM and an orange key (plus

backups) imprinted with the new RPV. Then you must physically go to all other HSMs that had the previous (lost) RPV and do the same, except you must say "Yes" to the PED's "Do you wish to reuse an existing keyset?..." question, in order to bring the new RPV to all HSMs that are intended to use Remote PED with the new orange PED Key(s). If you forget and say "No" to the PED's "...reuse..." question, then you are starting over.

ANSWER - white Audit PED Key : You will need to initialize the audit role on any affected HSM. This creates a new Audit identity for that HSM, which orphans all records and files previously created under the old, lost audit role. The audit files that were previously created can still be viewed, but they can no longer be cryptographically verified. Only records and files that are created under the new audit role can be verified, in future. Remember, when performing Audit init on the first HSM, you can say "Yes" or "No" to SafeNet PED's "Do you wish to reuse an existing keyset?..." question, as appropriate, but for any additional HSMs that must share that audit role, you must answer "Yes" to "Do you wish to reuse an existing keyset?..."

ANSWER - purple PED Key : If SRK was not enabled, this is not a problem - any purple PED Keys you had for that HSM are invalid anyway. If SRK was enabled, then your options depend on whether the HSM is currently in a tamper condition or Secure Transport mode... or not. There is no way to recover from a tamper or from Secure Transport Mode if the external split of the Master Tamper Key (the SRK) is not available. If you haven't got a backup purple key, your HSM is locked the moment it experiences a tamper event, or if it was placed in Secure Transport Mode. The same applies if you do have the key, but have forgotten/lost a numeric PED PIN that you [optionally] applied when the purple key was imprinted with the Secure Recovery Vector (the external split of the MTK). Either way, you must obtain an RMA and return the HSM to SafeNet for remanufacture. All HSM contents are lost.

If the purple key is lost, BUT the HSM is still in working mode - that is, it has not experienced a tamper event, and you have not placed it in Secure Transport Mode - then you should immediately rescue any important HSM or partition contents by backing them up, and restoring onto another HSM (that does NOT have SRK enabled, or for which SRK is enabled, but you DO still have the purple key). Once that is accomplished, obtain an RMA, and ship the HSM back to SafeNet for re-manufacture. It is not safe to continue using an HSM that has SRK enabled, but for which you have lost the purple PED Key. Any tamper event would render contents irretrievable. Avoid putting yourself in such a situation.

I have my PED Key, but I forgot my PED PIN! What can I do?

Forgetting a PED PIN is the same as not having the correct PED Key. See above, for your options in each situation. A PED PIN is an [OPTION] that you decide, at the time a role is created. If your security regime/protocol demands that your HSM access must enforce multi-factor authentication, then a PED PIN is a useful/necessary option for you. If your security protocol does NOT demand such measures, then you should seriously consider whether it is justified.

Once a PED PIN is imposed, it is a required component of role authentication, until/unless you arrange otherwise. You can remove the requirement for a PED PIN on a given HSM role only if you are currently able to authenticate (log in) to that role. For black PED Keys, you can have the SO reset your authentication. For other roles... not.

Thus, for blue or purple PED Keys, forgetting a PED PIN, like losing the PED Key (with no backups) is fatal.

For red PED Keys, forgetting the PED PIN is eventually fatal, but you can work in the meantime while you phase out your orphaned keys and objects.

Forgetting PED PINs for other roles, like losing their PED Keys is just more-or-less inconvenient, but normally not fatal.

I have my PED Keys and my PED PINS, but I can't remember which one goes with which HSM (or partition)!

See your options, above. The most serious one is the blue PED Key or the PED PIN for the SO role. You have only three tries to get it right. On the third wrong attempt, the HSM contents are lost. Wrong attempts are counted if you present the wrong blue PED Key, or if you type the wrong PED PIN with the right PED Key.

For black User PED Keys, and their PED PINS (if applicable) you have ten tries to get the right key or the right combination, unless the SO has changed from the default number of retries. If you are getting close to that maximum number of bad attempts, stop, and ask the SO to reset your partition PW.

For other PED Keys, there is no restriction on re-tries. Good luck. Try to be better organized in future.

PED Key Management

This chapter describes how to manage your PED keys. It contains the following sections:

- "PED Key Management Overview" below
- "PED Keys and Operational Roles" on page 221
- "Actions That Require a PED Key" on page 223
- "Shared or Group PED Keys" on page 226
- "Domain PED Keys" on page 228
- "Duplicating PED Keys" on page 230
- "How Many PED Keys Do I Need?" on page 232
- "Using MofN" on page 243
- "Complexity When Managing PED Keys" on page 247
- "General Advice on PED Key Handling" on page 248
- "Updating PED Keys – Example" on page 249
- "Updating PED Key for a Backup Token" on page 253
- "Frequently Asked Questions" on page 172

PED Key Management Overview

This section applies to SafeNet HSMs with PED (Trusted Path) Authentication, only.

As indicated elsewhere, the capability to imprint "group-User" PED Keys and "duplicate-User" PED Keys, permits considerable flexibility in the use, archiving and general management of PED Keys. For any role on the HSM, options like "group"/reuse, MofN, or the use of PED PINs (second factor of two-factor authentication) are imposed, or not, at the time the role is created.

The following pages address the ongoing management of PED Keys (which would normally include at least one "working" or "production" set, and at least one backup set, possibly stored off-site).

"Possible" Does Not Mean "Necessary"

When you initialize an HSM or create a Partition, SafeNet PED prompts you for various PED Keys and actions. Some are mandatory, some are advisable, and some are optional, depending upon your situation and requirements. Here is a quick summary:

Imprint a Blue PED Key

When an HSM is initialized, it sets up a blue Security Officer (SO) or HSM Administrator authentication PED Key (two names for the same function, depending upon the industry you are in). This is the key that you will need in future, to

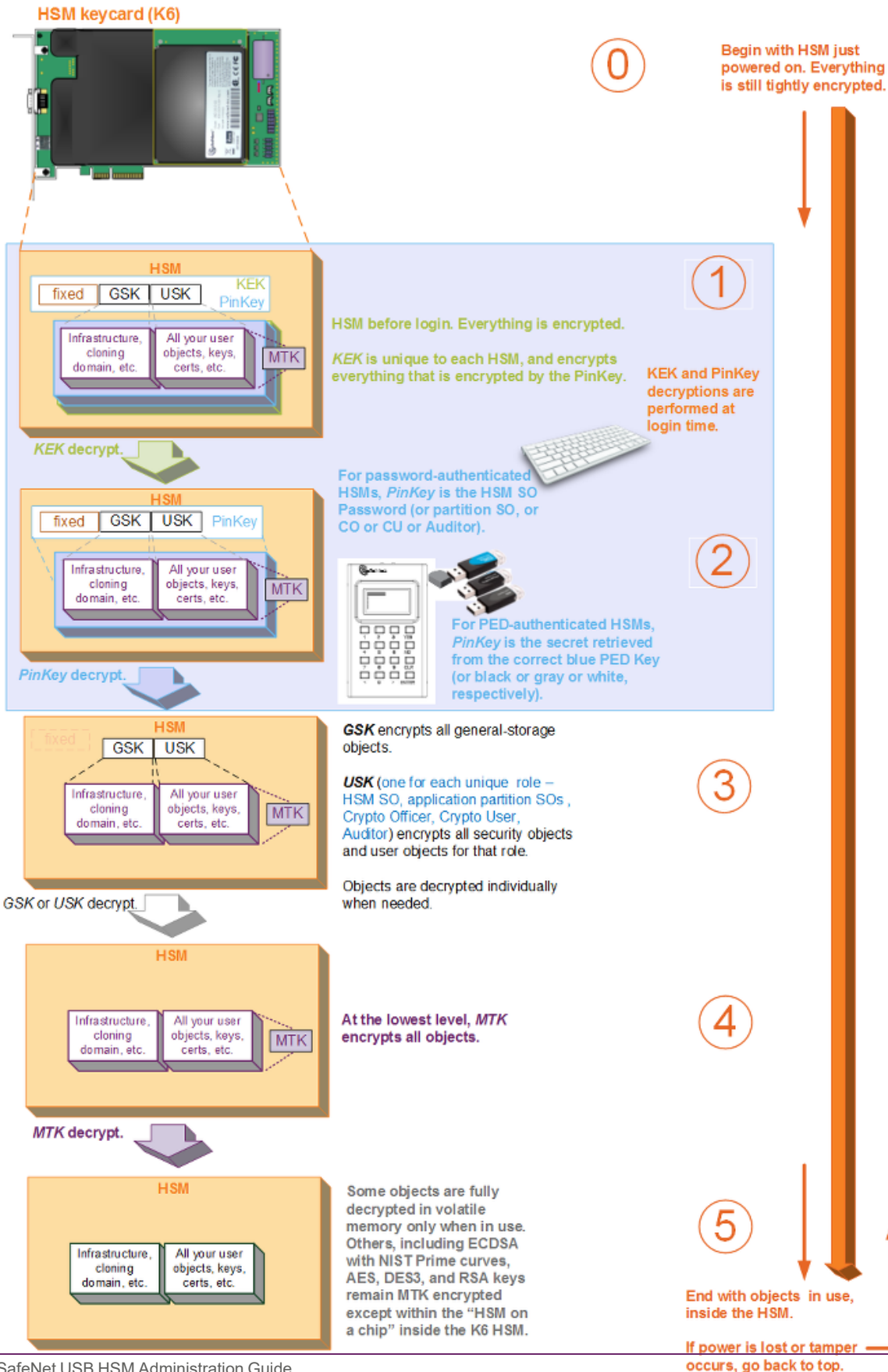
access that HSM. This can be done in one of two ways:

- the HSM can generate new, unique, random authentication data and imprint it onto a blue PED Key -- the resulting blue PED Key will now unlock that HSM, but no other (you do this when you answer "NO" to the "reuse an existing keyset (roughly equivalent to the "Group PED Key" question on the old PED 1.x)" question from the SafeNet PED)

OR

- the HSM can read the authentication from a blue PED Key that was already imprinted by another HSM, and accept that data as its own -- the blue PED Key can now unlock two (or more) different SafeNet HSMs (you do this when you answer "YES" to the "Reuse an existing keyset" question from SafeNet PED)

HSM Layered Encryption - the General Case



During initialization of an HSM, the HSM determines which blue PED Key will "unlock" the HSM in future. The HSM can create new, random authentication data and imprint that data onto a blue PED Key, **or** the HSM can scan an existing (previously imprinted) blue PED Key from another HSM and set the data from that older blue key as the new HSMs own "unlocking" data.

- For your very first HSM, you **must** initialize a blue PED Key for the HSM Admin.
- If this HSM is not the first; if you are creating a group of HSMs that are related in some way, then you CAN initialize a new blue PED Key for it, or you can re-use the authentication data on another blue PED Key (by deciding it will be a Group PED Key). This is your option. The HSM requires an imprinted blue PED Key when you access it, but you decide (at HSM initialization) whether that blue PED Key should be unique to this particular HSM, or shared among two or more HSMs.
- Whenever you perform an initialization, the SafeNet PED also gives you the option to make duplicates of your important PED Keys. If you already have enough (at least one primary and at least one backup), then you can just answer "NO" to the "Are you duplicating this key" prompt. If you need more of the current type of PED Key (in this case, the blue HSM Admin PED Key), then say "YES" and continue supplying additional blank keys until you have enough duplicates.




Note: If you are new to using PED keys and your security policy allows it, you should make a duplicate copy of the blue Security Officer and red cloning domain PED Keys as backups. See "[General Advice on PED Key Handling](#)" on page 248 for more information.










Note: The person or persons charged with ownership of the HSM, are responsible to safeguard the authentication secrets, ensuring that no unrecorded duplicates are made. Similarly, for application partitions with their own SO, the SO of each partition is responsible for securing the authentication secrets and copies.

PED Keys and Operational Roles

Below are some suggested holders of PED Keys by role.

Lifecycle	PED Key [Note 1]	Operational Role	Function	Custodian
<i>PED keys enforce division of operational roles and prevent unilateral action by key holders</i>				
HSM Admin		Security Officer	Manages provisioning activities and global security policies for the HSM : - HSM initialization, - partition provisioning, - global policy for the HSM and the partitions within it.	CSO CIO
		Domain Cloning Token Backup	Cryptographically defines the set of	Domain Administrator

Lifecycle	PED Key [Note 1]	Operational Role	Function	Custodian
			HSMs or partitions that can participate in cloning for the purposes of backup and high-availability.	WAN Administrator
		Secure Recovery	Restores an HSM after a Secure Transport or tamper event	CSO
		Remote PED	Establish a Remote PED connection	System Administrator
Application Partition Admin		Security Officer	Manages provisioning activities and global security policies for the partition : - partition initialization, - role setting, - policy setting.	
Daily Operation		Crypto Officer	This is the full user role associated with a partition. This role can perform both cryptographic services and key management functions on keys within the partition.	System Administrator
		Crypto User	This is a restricted user role on a partition. This role can perform cryptographic services using keys already existing within the partition,	System Administrator

Lifecycle	PED Key [Note 1]	Operational Role	Function	Custodian
			only. (See Note 2, below.)	
Ongoing Auditing		Audit User	An independent role responsible for audit log management. This role has no access to other HSM services.	Auditor
<p>[Note 1: This table implies a single PED Key for each HSM role or functional secret. For any role or PED Key secret, you can elect to invoke the MofN split-knowledge shared secret option, to spread the responsibility for that role or function over multiple persons. That is, you can require that a predetermined number of responsible persons, greater than one, must be present to unlock/access the particular HSM role or function. Choose MofN for a role or function when it is important that no single person have unsupervised access. See "About MofN" on page 1 and "Using MofN" on page 243.</p>				
<p>[Note 2: Functionally, the Crypto User (gray¹) PED Key is just another "black PED Key". The PED does not distinguish gray from black. The gray label is provided only for your convenience, so that CO and CU PED Keys are easy to visually identify and manage.</p> <p>It is useful to have two separate PED Keys (one for each of CO and CU) for separation of those administrative roles, in which case two different color labels are helpful for physical identification and handling. But if that administrative separation is not important in your setting, you can use just a single black key that authenticates to both roles, and still have two separate challenge secrets to give to applications:</p> <ul style="list-style-type: none"> - one for applications that need read-write crypto access to your partition, and - one for applications that are allowed only read-use access.] 				

Actions That Require a PED Key

It can, occasionally, be less than obvious why a certain action requires that you authenticate to the HSM or to the Partition, while another action does not.

Such questions have been carefully considered, from a crypto-security perspective, and we believe that we have consistently made the correct determination in all cases.

An example might be:

Question

If I activate an existing partition - make a service available to customers - I must have the black PED Key for that partition

But if I want to deactivate the same partition - withdraw/deny the service - I do not need the black key).

¹An alternate spelling of "grey". If you see either "gray" or "grey" throughout these documents, they refer to the same concept.

Is making a service available considered more dangerous than taking it away?

Answer

The rationale behind this behavior is that when you activate a partition you are making crypto services available to applications (that have the correct challenge password, of course). From a crypto module perspective, making crypto services available is a big thing and requires proper authentication. Removing that availability might be an operational issue but it is not a crypto security issue and, therefore, did not require the Black key.

As well, if an attacker wishing to deny service is given physical or command access to your HSM, he or she can do a lot more damage than simply issuing a `partition deactivate`. In other words, if you have let them get that deeply inside your security perimeter, then you have far worse problems than a "partition deactivate".

If you ever discover a situation where our implementation seems inconsistent with that philosophy, please let us know by contacting support@safenet-inc.com. We will either fix the problem or explain why it is not considered a problem.

Question

If I have a service running, can I force my application's administrator to provide the partition's password each time the service is restarted and/or each time the application server is restarted?

Answer

It depends on the setup, and it depends on what you mean by password. But, in limited circumstances, yes, although you probably would not want to do that.

It doesn't really matter whether your application accesses the HSM directly, or whether a service or some other provider is between application and HSM.

If an application directly accesses the HSM partition, then when it first does so, the application must initialize the library, and open a session on the HSM. Then the application provides the partition authentication when the application needs to perform actions on partition contents. For either Password-authenticated or PED-authenticated HSMs, the partition password (or partition challenge) secret must be available to the application so that it can provide that secret when access to partition objects is needed. The application provides the partition secret to say that it (the application) has the right to perform partition-object actions. This usually means that the secret is stored somewhere on the host's file system or registry (most likely encrypted) for retrieval when needed .

This means that the application is already providing the partition password (or partition challenge secret) string whenever it is demanded by the HSM. So, in that sense, the answer to your question is already "yes".

But if you meant something stronger than the password-presenting action that the application must already perform when accessing partition objects, you probably need to consider PED authenticated HSMs.

For PED-authenticated HSMs, the PED Key data for that partition must be provided to the HSM before the partition secret string is provided.

In almost all cases, for PED-authenticated HSMs, customers would Activate the partition when first setting up, so the PED Key data for that partition would be cached. That is, the customer would be making an authenticated administrative declaration (activating the partition) that the partition was "open for business", and an application with the partition challenge secret could then access partition objects at any time.

Note: Authentication differences - Password-authenticated vs PED-authenticated:

- When the HSM is PED-authenticated,

- the *administrative role secret* contained on a black or gray PED Key is one secret, used only by administrative personnel, while
- the *challenge-secret or password* is a second secret (plain text, initially presented on the PED screen, but you can change it), which is the application-authentication secret, that allows the HSM verify that the presenting application is entitled to perform cryptographic operations on the particular application partition.



The application can submit its own authentication (that second secret) only after the PED Key secret has "opened" the HSM partition for operation (by Activating) - that is, there are two levels of protection, one administrative, and the other operational, where the operational level is gated by the administrative level.

- When the HSM is Password-authenticated,

- the *administrative role secret is also the application-authentication secret*, one plain-text secret used for two purposes; the application that knows that secret declares the application partition open-for-business while in the act of accessing it with that single secret as its authentication - a single level of protection that is both administrative and operational. On a Password-authenticated HSM, once the administrator (Crypto Officer or Crypto User) has distributed the secret to the application(s), the only way to restrict access by applications (or personnel) that have come into possession of that secret is to change the password - which also changes the authentication for the associated administrative role.
-

Activation and autoActivation of Partitions

With the partition unlocked by Activation and the application in possession of the challenge secret or password, the application could open and close sessions at will, and whenever it needed to manipulate partition objects, the application would provide the partition (challenge) secret. Once the Partition PED Key data is available, the action of accessing and using partition objects is identical for PED-authenticated or Password-authenticated HSMs.

If the partition is autoActivated, then the black PED Key data is cached in the HSM, just as for Activation, except it is now protected against power failure for as much as two hours.

So, for the direct application-to-HSM scenario, if you want to force the application owner to perform an authentication beyond what the application already performs with each access to partition objects, then you would need:

- a. to use a PED-authenticated HSM, and
- b. ensure that the PED Key data for that partition was NOT cached - therefore, no Activation or autoActivation (Partition Policies 22 and 23 would be set to "off").

The application still has its access to the partition challenge secret, but the partition is not "open for business". A PED Key must be provided (possibly a PED PIN as well, if you set one), every time authentication is needed.

The drawback, in that scenario, is that *every* access of partition objects now requires PED Key authentication, in addition to the partition challenge secret. The PED would remain connected to the HSM, the key for that partition would remain inserted, and somebody would have her-or-his finger poised to press the PED's [Enter] key every time the application needed to manipulate a partition object.

The above is the situation for direct access to the HSM by an application; it is only for very specialized situations where the partition is rarely accessed, and extremely close control is required.

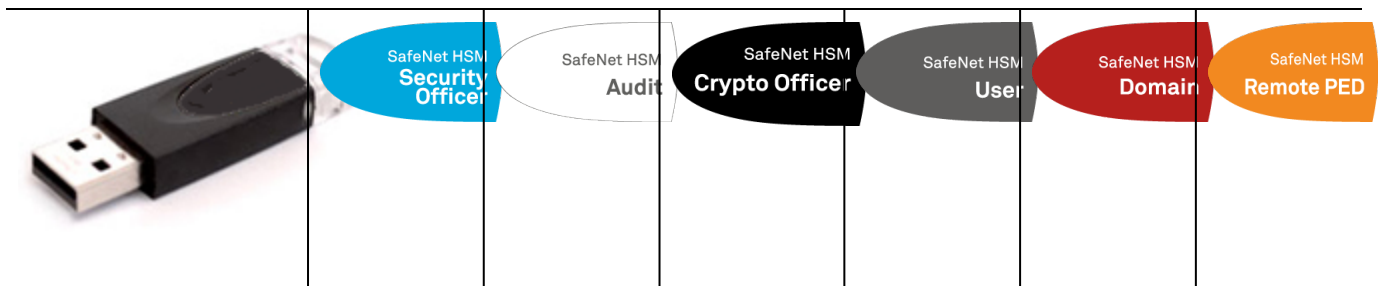
If you insert a service between your application(s) and the HSM, then the application no longer needs to know anything about HSM and its authentication. Instead, the service must handle all that, translating between the application and the HSM. The conditions described earlier now apply to the service.

Similarly, if you insert a provider (a translation layer like our CSP or KSP or JSP) between your application and the HSM, or between your service and the HSM, then the provider takes care of safeguarding and using the partition password (or partition challenge) secret as needed.

In either case, the need for PED Key presentation and PED button pressing is determined by the state of Partition Policies 22 and 23).

See also "[Commands that Require SafeNet PED Interaction](#)".

Shared or Group PED Keys



With the common administrative group option (answer "YES" to the SafeNet PED question :

Do you wish to reuse an existing keyset? during HSM initialization or Partition creation (one PED Key accesses multiple HSMs) – as opposed to the default unique secret (where each HSM has its own unique PED Key) – you can use numerous HSMs and not need to manage numerous keys.

For example, at an installation employing five SafeNet HSMs:

- the unique key option would create five different, mutually exclusive blue SO PED Keys, one to access each of the individual HSMs (a gain in exclusivity of HSM ownership, at the cost of additional PED Keys to manage and control)
- compared to**
- the common administrative group PED Key option where you might have a single SO PED Key that could access any of the five HSMs (a savings at the administrative level, at the cost of HSM ownership exclusivity (if one key is compromised, it compromises all five HSMs)).

How does it work?

During the process of initializing an HSM, or creating an HSM Partition (on SafeNet HSM with PED [Trusted Path] Authentication), SafeNet PED attempts to imprint a blue or a black or a red PED Key.

Similarly, the orange PED Key can be shared among several HSMs, although it is created in its own process, and not as part of HSM initialization or partition creation. The white Audit PED Key is also created and maintained in its own process, and not as part of HSM or partition initialization. Both the orange and white keys, like the others, can be made common among multiple HSMs if desired.

When the creation of an entity invokes the PED, the PED asks:

Do you wish to reuse an existing keyset?

Press “YES” on the SafeNet PED keypad if you are inserting a key that can access previous HSMs (meaning that another HSM was initialized with this PED Key). Choosing “YES” *preserves* the old access code on the PED Key and applies it also to the current HSM or token. Thereafter, the PED Key can access both (or multiple) HSMs or tokens that share the same access secret. The randomly-generated PIN on the PED-key is not overwritten.



In other words, saying “YES” to the PED prompt “Do you wish to reuse an existing keyset”, is the method to share a common authentication secret among multiple HSMs.

Alternatively, if you wish to have different PED Keys associated with each HSM in your possession, answer ‘NO’. A ‘NO’, is a choice to overwrite the PIN (if one is already present) and store a new, randomly-generated PIN on this PED Key – any existing authentication code on this PED Key is to be overwritten with a new code, good with only the current HSM or token. The same applies to black HSM Partition User PED Keys.



The red PED Keys **must** have the same domain secret for each HSM that will synchronize (backup and restore, or HA) with another. An HSM backup partition or token content can be restored only onto an HSM that was initialized with the same red key secret. You must always choose to “...reuse an existing keyset” when initializing any HSM after the first one in a cloning group, or any partition after the first one in a cloning group.



The orange RPK PED Key, for RPV (Remote PED Vector), carries a secret that matches the RPV on an HSM to which you will be remotely authenticating with SafeNet PED 2 remote version. If you wish more than one HSM to have the same RPK, then you would choose to “...reuse an existing keyset” when setting RPK with “hsm ped vector init”.



The white Audit PED Key carries the secret that authenticates the holder of the Audit role for the current HSM, and for any other HSMs where you have chosen to “Reuse” the PED Key when initializing the Audit role.

Reusing a PED Key forces all PED PINS to be the same

The Exception



The purple SRK PED Key differs from the others, in that it cannot be used with more than one HSM in common. You can reuse a purple PED Key with a different HSM by overwriting the key, but you cannot reuse the secret on that key with any HSM other than the one that originated the secret. The SRV (secure recovery vector) is not transferable. Each SRV is unique. An HSM can export an SRV split of its Master Tamper Key onto a purple PED Key (SRK) for use with only that HSM. If you imprint a valid purple PED Key with any other HSM, the key takes on a new SRV split that is valid with the new HSM, and is no longer useful with the original HSM.

Domain PED Keys



A domain PED key is an iKey 1000 secret.

(marked with)



and imprinted with a domain

A domain PED Key (the red one) carries the key-cloning vector (the domain identifier) that allows cloning to take place among HSMs and tokens. Cloning is a secure method of copying HSM (or Partition) or token objects, such that they can be replicated between HSMs and tokens, but:

- strongly encrypted (never in the clear), and
- only between HSMs and tokens that share a cloning domain.

Cloning is the method by which secure HSM and Partition backup is possible to a SafeNet Backup HSM, and by which restoring is possible from a Backup HSM or token to a SafeNet HSM or Partition. It is also used when HSM log records and files are verified by an HSM other than the one that originally created those records.

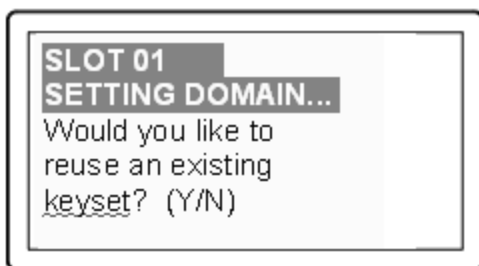
At initialization time, the key-cloning vector is created on the HSM and imprinted onto a red PED Key, or if a desired cloning domain already exists, then the existing key-cloning vector from a red PED Key is read from that PED Key and imprinted on the HSM (or Backup token) as the HSM (or token) is initialized. HSMs and tokens that share a key-cloning vector are said to be members of a cloning domain.

An HSM or token can be a member of only one domain. To make an HSM or token become a member of a second or different domain, you must initialize the HSM or token and imprint the new key-cloning vector -- the first one is destroyed and the HSM or token is now a member of only the second domain. This action also destroys any previous content on the HSM being initialized.

To cause a SafeNet HSM or Partition to be a duplicate or mirror image of another, the procedure is to backup the first HSM or Partition, and then restore from the Backup token onto the new HSM (or Partition).

The "New Domain" Question

When you initialize an HSM, and are prompted for a red PED Key, SafeNet PED first asks:



If you answer [No]:

- You are telling SafeNet PED that it should retrieve a new domain (Key Cloning Vector) from the HSM and prepare to overwrite that new domain secret onto a blank key that you are about to insert, or overwrite the existing random domain vector on a red PED Key that you are about to insert.
- This was your last chance (short of aborting the procedure) to make the current HSM part of an existing cloning

group. Further prompts in this sequence will give you the opportunity to remove keys that you have mistakenly offered (that have useful authentication secrets on them) and substitute another, but you get no more opportunity to change the "No" to a "Yes".

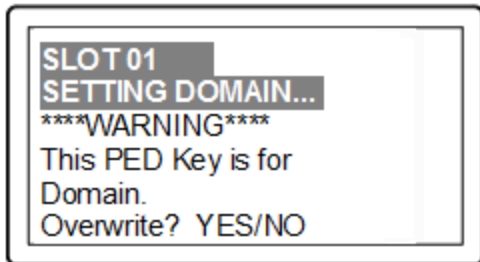
- If that red PED Key was already in use on an operational HSM (and Backup device), then that HSM (as well as the Backup device) carries the old domain and the newly overwritten red PED Key can no longer be used with it — therefore, unless you have a duplicate red PED Key with the old cloning domain (key-cloning vector), then that previous HSM cannot be backed up, and its Backup cannot be restored

If you answer [Yes]:

- SafeNet PED prepares to preserve the domain (key-cloning vector) value that it now expects to find on the red PED Key, and store it onto the HSM -- this causes the current HSM to share the domain with the previous HSM and/or Backup device
- With two or more HSMs (and at least one Backup HSM) sharing the same cloning domain, it is possible to clone the contents from one to another by means of backup and restore operations

Assuming that you responded [No], the PED asks additional preparatory questions, then asks you to insert a PED Key (which you should already have labeled with a red sticker). The PED scans the red PED Key for an existing key-cloning vector. If none is found, SafeNet PED imprints a new one, taken from the HSM, and that same new key-cloning vector is saved onto the HSM.

However, if an existing key-cloning vector (or other secret) *is* found, SafeNet PED needs to know whether to retain it. SafeNet PED asks:



If you answer Yes:

- SafeNet PED overwrites the existing random domain vector (or other secret) on the inserted red PED Key
- If that red PED Key was already in use on an operational HSM (and Backup device), then that HSM (as well as the Backup device) carries the old domain and the newly overwritten red PED Key can no longer be used with it — therefore, unless you have a duplicate red PED Key with the old cloning domain (key-cloning vector), then that previous HSM cannot be backed up, and its Backup cannot be restored

If you answer No:

- SafeNet PED goes back a step and asks you to "Insert a Domain PED Key", which is your opportunity to correct the mistake by removing the first PED Key and inserting either a fresh (never-imprinted PED Key, or inserting a PED Key that contains an outmoded secret (Domain, SO, User, RPV, SRV).
- Each time you insert a PED Key, during an operation that could write to the key, SafeNet PED tells you if it is blank or if it contains a pre-existing secret, and asks if you wish to overwrite. This continues until you insert a key and allow the PED to overwrite whatever is-or-isn't on that key, or until the operation times out.
- If two or more HSMs (and at least one Backup HSM) share the same cloning domain, it is possible to clone the contents from one to another by means of backup and restore operations

To What Does a Domain Apply?

Each HSM has a domain that covers any object that can exist in the SO space - this is created at HSM initialization time. Usually objects in the SO area of the HSM are specialized keys used to facilitate HSM operations (example, masking key).

Each partition in an HSM has a domain of its own - this is created when the partition is created/initialized. Partitions contain customer-owned keys used in client operations, as well as data objects.

Objects on a partition can be cloned to another partition (whether on the same HSM or on another HSM) only if both partitions share the same domain.

In the current SafeNet Network HSM 6.x sense, one domain is like another [there is nothing special about one firmware 6 domain versus another firmware 6 domain] and could be applied to any partition or HSM SO space. Only your security and management policies dictate how you share domains. You can segregate HSMs and partitions into clonable groups. Cloning can occur among any/all members of a group that share a domain. Cloning cannot occur between members of two different domain groups.

Any HSM SO space can have only one domain, assigned at initialization time.

Any partition can have only one domain, assigned at partition creation time. It is not possible for a partition or an SO space to be a member of more than one domain. It is possible for different partitions on the same HSM to be members of mutually exclusive domains.

There is no limit to the number of partitions or HSMs that can share a common domain.

What about Legacy HSMs and Partitions?

HSMs before the K6 (the HSM inside SafeNet Network HSM) and G5 (the HSM for PKI with SafeNet Network HSM, and the core of the SafeNet Backup HSM) - legacy HSMs - used an older, smaller domain secret, which is incompatible with current HSMs.

Cloning of objects between SafeNet HSMs requires a shared domain.

To provide a one-way migration path to move HSM objects from legacy HSMs to modern HSMs, a command `partition setLegacyDomain` allows an old-style domain to be linked to a new-style domain on a K6 or G5 HSM.

Summary

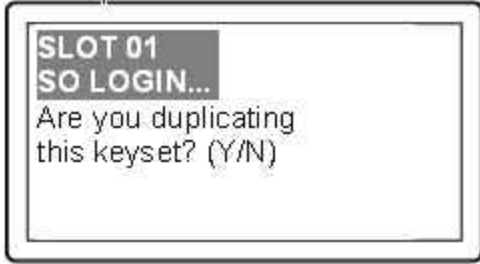
If you can account for all the HSMs to which you have presented your red Domain PED Key (meaning that you have maintained strict control of that red PED Key), then you know with certainty that nobody else could possibly have a copy of the sensitive keys that were created on your HSMs or partitions, or cloned to those HSMs or partitions.

Duplicating PED Keys

When you have imprinted any PED Key, having set its parameters,

- is it re-used?
- does it have an optional PED PIN?
- is the secret split into N parts?

you are then prompted:



If you answer YES:

- this invokes the duplication of the PED Key (any number), so that all duplicates can be interchangeable (backups)
- you can now use the original or any of the duplicates to access this HSM or Partition (blue or black keys, respectively), and distribute the others to other personnel or to secure storage
- you should decide how many backup PED Keys are required by your organizational security policies

If you answer NO:

- you are indicating that no duplicates/backups are necessary
- if you eventually require duplicate/backups for your SO PED Keys, you can do so when you initialize another HSM or when you perform an "hsm so-ped-key change" (saying "NO" to the "reusing" question, and then saying "YES" to the "duplicating" question at that time)
- if you eventually require duplicate/backups for your Partition User/Crypto Officer PED Keys, you can do so when you create another Partition (saying "NO" to the "reusing" question, and then saying "YES" to the "duplicating" question at that time)
- the same possibility is presented whenever you imprint any of the other keys (Domain, RPK, SRK)
- you can also create duplicates of any PED Key, except the purple (SRK), by means of SafeNet PED's Admin menu.

Considerations for Duplicate PED Keys

The duplicate PED Key option permits you to issue (or store) more than one PED Key (duplicates) for any of:

- HSM Admin
- Auditor
- Remote PED vector
- Secure Recovery vector
- Owner PED Key (legacy),
- Partition SO (PPSO),
- Crypto Officer or Crypto User per HSM Partition.

The most common use of this feature is to make backups of each PED Key, for secure storage against possible damage to, or loss of, the primary PED Key for an HSM or token.

Your in-house procedures and working arrangements might benefit from having two or more copies of some-or-all PED Keys for an HSM. For example, if your procedures require that each work-shift must either sign PED Keys over

to the next shift, or sign them into lockup storage, then you need only the single primary PED Key in "circulation", and you have very secure management of such keys.

However, your procedures could be somewhat less stringent. If it proves more convenient and workable to have each person carry his own PED Key(s) on his person at all times, then a copy of the relevant PED Key will be needed by each person who must ever have access to any given HSM Partition, and to each person with HSM Admin/SO privileges.

In summary, this is an **option**. If you need more copies of a particular PED Key, answer "YES" when you see the "Are you duplicating..." prompt. Any operation that causes SafeNet PED to offer the "Are you duplicating this PED Key? (YES/NO)" prompt is an opportunity to make as many more copies of that key as you wish. If you already have enough duplicates, just answer "NO" whenever you see the prompt.

Implications of Duplicate PED Keys

By implication, your security and operational procedures must ensure that no person takes advantage of that facility to make unauthorized or un-tracked copies of any key.

The SafeNet PED (and the associated HSM) do not know how many copies you have made, so you are given the option every time you initialize an HSM or create a role or secret, just in case you might want to create some more duplicates of the currently inserted key. You can also make copies at any time by using the on-board admin menu of the SafeNet PED 2.x. If your security model allows people to carry PED Keys around, this might be a good argument for imposing the use of PED PIN "something you know" secrets when initializing. If somebody loses an imprinted PED Key, the person who finds it has potential access to your HSM, in that role. However, if a misplaced (or stolen) imprinted PED Key also has a PED PIN associated with it, then it would be much more difficult for the finder to make use of the found/stolen PED Key.

What a duplicate PED Key is Not

Duplicate PED Keys are not the same as MofN-split PED Keys. Whatever secret is on a PED Key that you duplicate is the secret that is contained on the duplicate(s). If you selected "M value" and "N value" to be 1 (one) when creating the first PED Key, then there is no splitting of the secret, therefore any duplicate of that key is also a complete, self-contained copy of that secret, and either the original or the duplicate is fully sufficient to authenticate. If you choose to split a secret when creating it, by selecting "M value" and "N value" greater than 1 (one), then a duplicate of that secret must create duplicates of all the splits.

How Many PED Keys Do I Need?

You need enough to satisfy your operational and security-policy requirements. How that translates to an actual number of PED Keys depends on your situation. Here is some guidance.

Basic amount for operation

The basic amount is described in the topic/page ["About PED Keys" on page 186](#). If you elect not to make use of some of the roles and functions, then you have fewer to manage than the seven (legacy) or eight (with Per-Partition SO) that are described there.

The next question is: How many HSMs do you have? If you have just one SafeNet HSM, then there is no need to consider other HSMs when you determine your security/role/authentication policy. If you have more than one SafeNet HSM, you need to make decisions about how they are to be administered and who will do it. These issues are discussed and illustrated in the following sections.

Regarding HSM authentication, the factors to consider are:

- How many HSMs are to be administered by one identity (are you making a group PED Key for several HSMs, or a unique PED Key per HSM)?
- Does your security policy allow a single person to control access to HSM functions, or should that access be divided, requiring multiple authorized persons to oversee authentication to an HSM role (["About MofN" on page 1](#), or ["Many PED Keys for one HSM - MofN" on page 236](#) below)?
- Will these decisions apply to all HSMs within your sphere?
- Will these decisions apply to all, or to just some roles on each HSM?
- For each role or authentication secret related to an HSM, how many operational and on-site- and off-site-backup copies must be created and maintained and tracked?

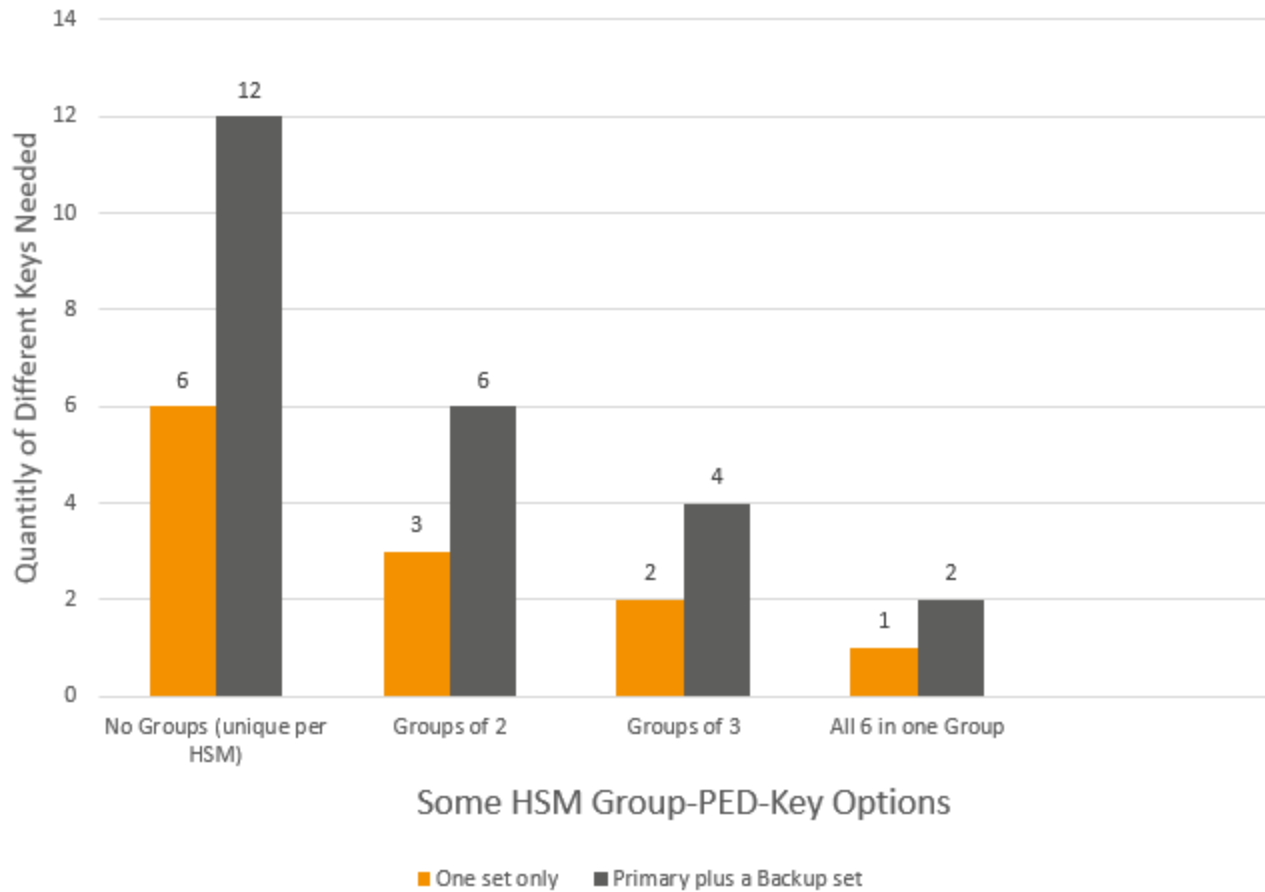
One PED Key for many HSMs - grouping or reuse

This section discusses the number of on-site full sets required, for different choices regarding PED Key reuse/grouping, and then considers the implications of having backup authentication keys. For this discussion, assume that you have more than one HSM to administer - for this example, we arbitrarily assume six, but the concepts could apply to any number.

The following example considers grouping of PED Keys, which is the outcome of your response to the SafeNet PED prompt "Do you wish to reuse a keyset? Y/N". You can imprint a new, unique secret onto both the HSM and the current PED Key, by responding "No" to the reuse question. This is the "no group" or the "group of 1" scenario.

Alternatively, you can accept a currently-valid PED Key and imprint that secret onto the HSM, such that the current PED Key can unlock its role or function on the current HSM as well as on one or more HSMs that already have the same secret (for that role or function). This is the "grouping" scenario, where one PED Key is good for multiple HSMs, which eases the administrative burden if your security policy so permits.

PED Keys Needed - 6 HSMs with Grouping



The above chart depicts some options for the "...reusing a keyset..." question from SafeNet PED when you are imprinting an HSM authentication secret. The example shown is an organization responsible for six (6) SafeNet HSMs.

The first column of each pair in the chart, in orange, depicts how many keys you would need for a given secret :

- On the left, "No Groups", if you set a unique secret for each of the six HSMs (responding "No" every time SafeNet PED asked if you were reusing a keyset; therefore 6 different keys, one for each HSM). Six keys are needed for that one secret.
- If you chose to imprint authentication secrets onto your six HSMs in three groups of two HSMs (responding "No" when asked if you were reusing a keyset, for the first HSM of each group, then responding "Yes" when imprinting the secret for the second HSM of each group). Three keys are needed for that secret.
- If you chose to imprint authentication secrets onto your six HSMs in two groups of three HSMs (responding "No" when asked if you were reusing a keyset, for the first HSM of each group, then responding "Yes" when imprinting the secret for the second HSM, and again for the third HSM of each group). Two keys are needed for that secret.
- If you chose to imprint authentication secrets onto your six HSMs in one group of six HSMs (responding "No" when asked if you were reusing a keyset, for the first HSM of the group, then responding "Yes" when imprinting the secret for the second HSM, and again for the third HSM, and again for the fourth HSM, and again for the fifth HSM, and again for the sixth HSM of the group). One key is needed for that secret.



Note: Yes, you could also do a group of five and a group of one, or a group of two and a group of four... any combination that works in your operational environment and satisfies your security requirements.



Note: So far, we are discussing just one authentication secret, out of seven or eight for any HSM. We would need to repeat this discussion for each of the other secrets, per HSM that you manage.

However, those quantities, enumerated above, imply that you have just one set of PED Keys for the HSMs that you need to access. This is unwise; if you lose or damage the only key of a given color (for a given role or feature), you lose access to that aspect of that HSM or that group of HSMs. With that in mind, the second column of each pair in the chart, in gray, represents how many physical PED Keys you would need in each situation for reliable access to your six HSMs, if you have one working (primary) set, as well as an additional, backup set of keys.

Many organizations would go further, and insist on having three complete sets, one primary or working set, one backup set to be stored in on-site secure lockup, and a third set to be stored in off-site secure lockup. This kind of requirement might be specified or implied in a Business Continuity/Disaster Recovery plan, or in a comprehensive security policy.

Again, with each HSM and with each authentication secret or secured function on that HSM, at the time the secret is created or reset, the PED gives you the opportunity:

- to "Reuse an existing keyset" - make the current HSM secret part of an existing group that is unlocked by an already-imprinted PED Key (or an already-imprinted MofN keyset), or
- to **not** "Reuse an existing keyset", and thereby use a fresh, unique secret generated by the current HSM that is not shared by any other HSM, meaning that this is the opportunity to start a new group, independent of any existing group of HSMs.

The same issues arise with respect to all PED-mediated secrets (HSM SO, partition SO, cloning domain, Crypto Officer and Crypto User, Remote PED, Auditor). The exception is the purple Secure Recovery PED Key, which is unique to its HSM and cannot be grouped.

The other special case is the red Cloning Domain PED Key, which **must** be shared where you wish to be able to clone contents among HSMs or among application partitions.

Secrets are independent for grouping

The different secrets are independent. You could group SO authentication keys but keep partition Crypto Officer authentication keys unique. You could maintain unique SO and Crypto Officer authentication keys for all your HSMs and their partitions, while at the same time having a single, grouped Cloning Domain for all... or for some.

You could decide that some combination of grouping and uniqueness was suitable for all your HSMs and their partitions and roles, but that you wanted a single Auditor identity for all HSMs in your organization.

You could implement any combination of unique and grouped authentication secrets that suited your organization's working methods and security policies.



CAUTION: Always have at least one complete set of backup PED Keys in addition to the working or operational set.

The above sections have discussed deploying HSM authentication secrets on a one-for-one (unique PED Key per HSM) or a one-for-many (grouped PED Key for multiple HSMs) basis, or any of several possible combinations.



Note: HSM grouping, or PED Key reuse, helps to reduce the number of PED Keys that must be tracked and managed in your organization. If you have multiple HSMs, and if there is no operational or security reason to maintain unique authentication for each, then grouping several HSMs to be accessed by one PED Key is a useful option.

The next sections discuss a different option, regarding PED Keys, that can have considerable effect on the number of PED Keys that you must manage.

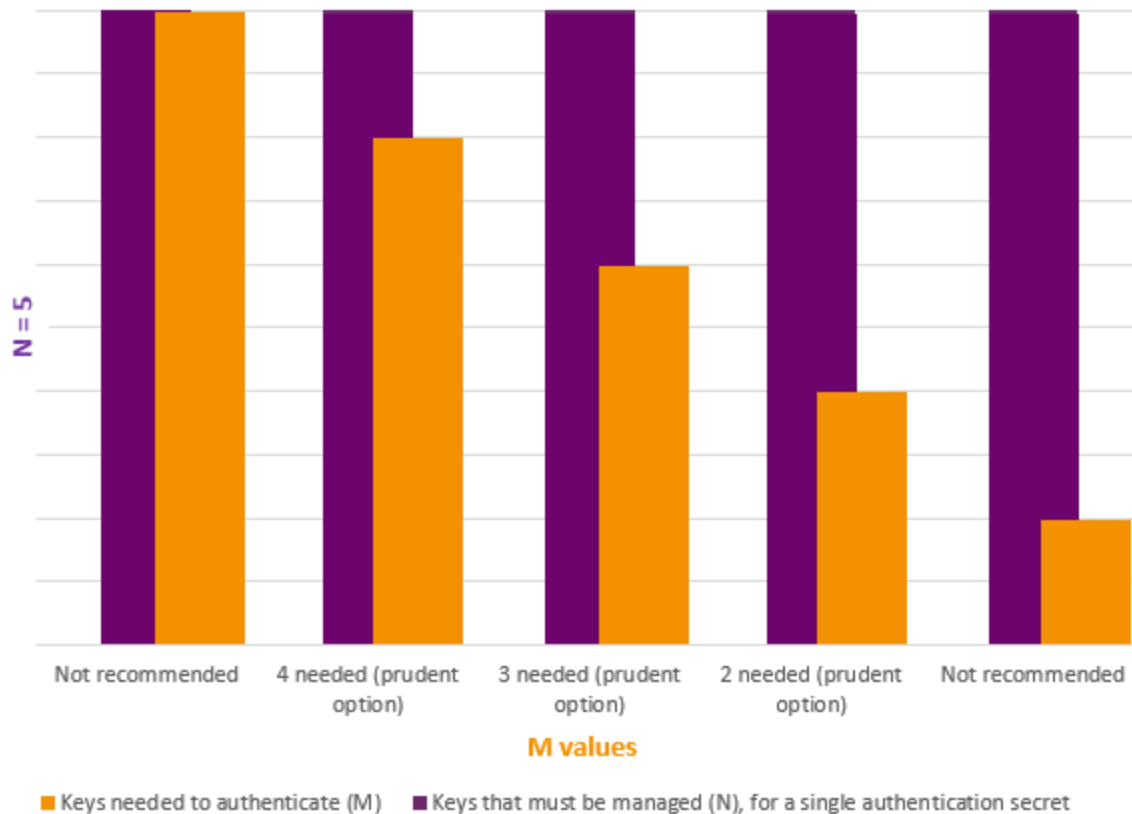
Many PED Keys for one HSM - MofN

Does your security policy allow you to trust your personnel? Perhaps you wish to spread the responsibility - and reduce the possibility of unilateral action - by splitting an authentication secret, invoking split-secret, multi-person, access control. Choose the MofN option so that (for example) no single blue PED Key is sufficient to unlock an HSM SO role.

If you invoke MofN, two or more of a given color of PED Keys (your choice, up to a maximum of 16 splits of each secret) would then be needed to access that role or that secure function on each HSM. Distribute each split to a different person, ensuring that no one person can unlock that aspect of the HSM (HSM SO, partition SO, cloning domain, Crypto Officer and Crypto User, Remote PED, Auditor, Secure Recovery).

If you have decided that you want (in this example) three different people to be present whenever a particular role authenticates to the HSM, you should also allow a few extra splits of that secret to accommodate accidents, illness, vacations, business travel, or other reasons that would take some key-holders away from the HSM site. Perhaps you settle on two additional splits as sufficient additional key-holders, beyond the initial three. You have thus specified MofN to be 3 of 5. So, if this example applied to the HSM SO, then each HSM's SO secret might be split into five components or partial secrets imprinted onto a set of five blue PED Keys, of which any three from that set can combine to reconstitute the SO secret, but never less than three.

PED Keys Needed for one authentication secret with M of N



The purple bars show $N=5$ for every choice of M , the orange bars in this example.

- $M=N$ is not recommended because it allows no scope for one of the holders to be unavailable, while still allowing you to access your HSM.
- $M=1$ is not recommended, because it is no more secure than if there were no splits of the secret - a single person can unlock the HSM role or function without oversight.
- Any choice where $N > M > 1$ is prudent and useful, as it ensures oversight but allows for at least one split-holder to be unavailable, while still permitting authorized access to the HSM roles and functions.

Whether you assigned SOs to HSMs on a one-for-one or a group basis (see "[One PED Key for many HSMs - grouping or reuse](#)" on page 233 above), you now multiply that number of SOs by N (the number of splits into which each SO secret is separated). There is no overlap - no split can be part of more than one secret. The number of PED Keys to manage has become significant, especially when you consider that each one (each split of each SO secret) should have at least one backup. You can apply the above example to any of the other authentication secrets instead of (or in addition to) the SO.

With $MofN$, you need very good procedures to physically identify and track the various keys.

Secrets are independent for MofN

Each HSM must have at least one application partition, in addition to the HSM administrative (SO) partition. Some SafeNet products can have multiple partitions. The number depends upon your operational requirement and the number that you purchased, per HSM, up to the product maximum per unit. Each partition requires as many as three role authentications (Partition SO, Crypto Officer, and (optionally) Crypto User) in addition to a cloning domain secret. For each of the three roles, plus the domain, you must initialize the role or secret and decide whether that role needs MofN access control, and if so, what the values of M and N should be for that role.

If you require verifiable HSM audit logs, you must initialize the HSM Auditor role (white PED Key) and decide whether that role needs MofN access control, and if so, what the values of M and N should be for that role.

If you intend to manage your PED-authenticated HSM remotely, you must initialize a Remote PED Vector (orange PED Key) and decide whether that role needs MofN access control, and if so, what the values of M and N should be for that role.

If you wish to set the HSM into Secure Transport Mode, or if you wish to require that any tamper event must be physically acknowledged and the HSM must wait until it is explicitly returned from tamper condition, then you must enable the Secure Recovery Vector (purple PED Key) and decide whether that role needs MofN access control, and if so, what the values of M and N should be for that role.

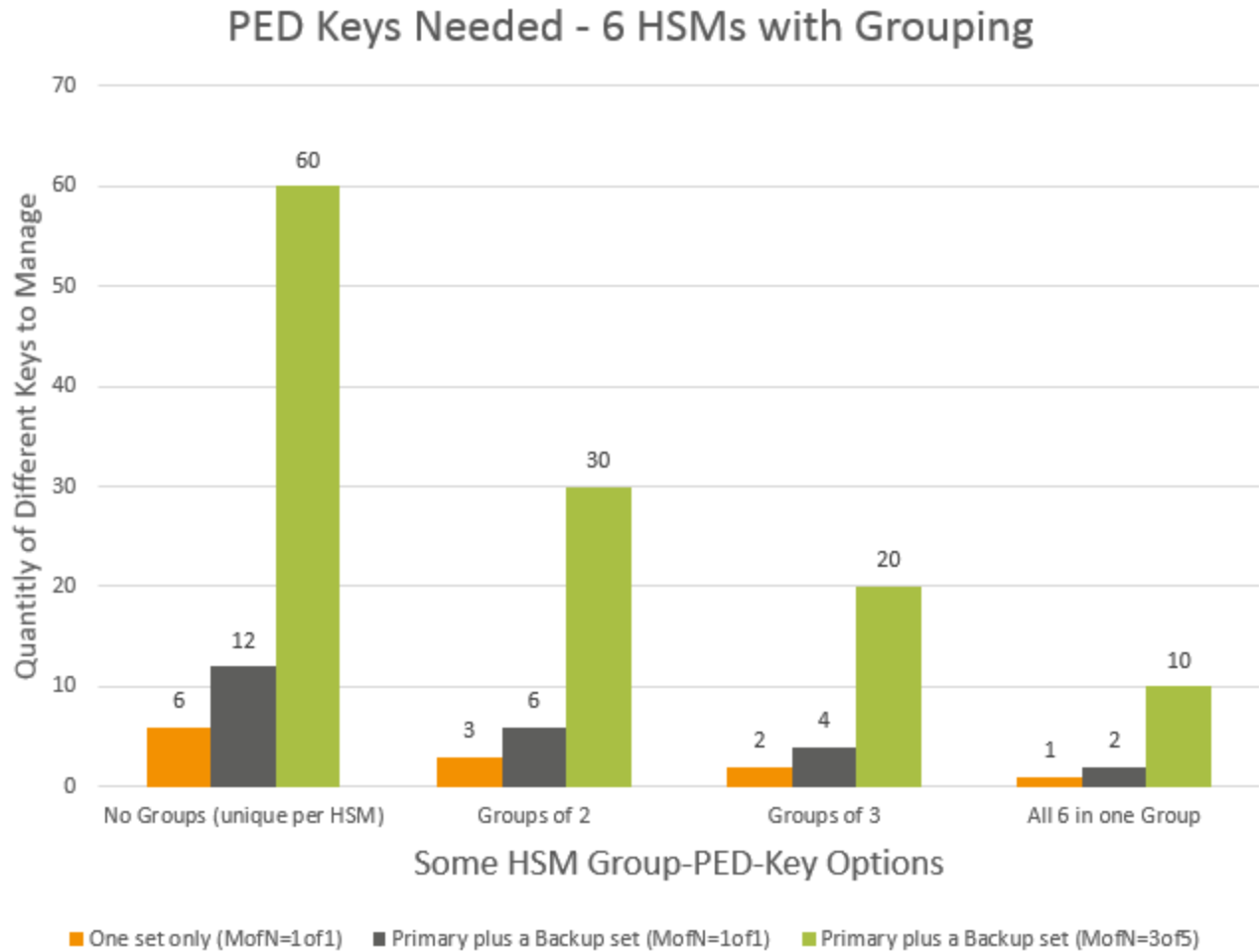
As shown, you can elect to split none of the HSM roles and secrets, some of them, or all of them. And if you do elect to impose MofN for some, or all roles and secrets, you can set different values of M and N, independently per HSM role or secret.

Combining MofN with Grouped/Unique Authentication

So, we have as many as eight different authentications on an HSM with a single application partition; nine if the application partition's cloning domain is different from the HSM SO's cloning domain. Only one of those - the purple SRK PED Key - is not subject to grouping if you have more than one HSM under your control.

Keeping in mind the need for backup sets, if you impose MofN for some or all secrets, that choice can drastically increase the number of PED Keys that must be imprinted, tracked, and managed, while the ability to group authentication secrets across some or all of your HSMs can help reduce the numbers of PED Keys in play.

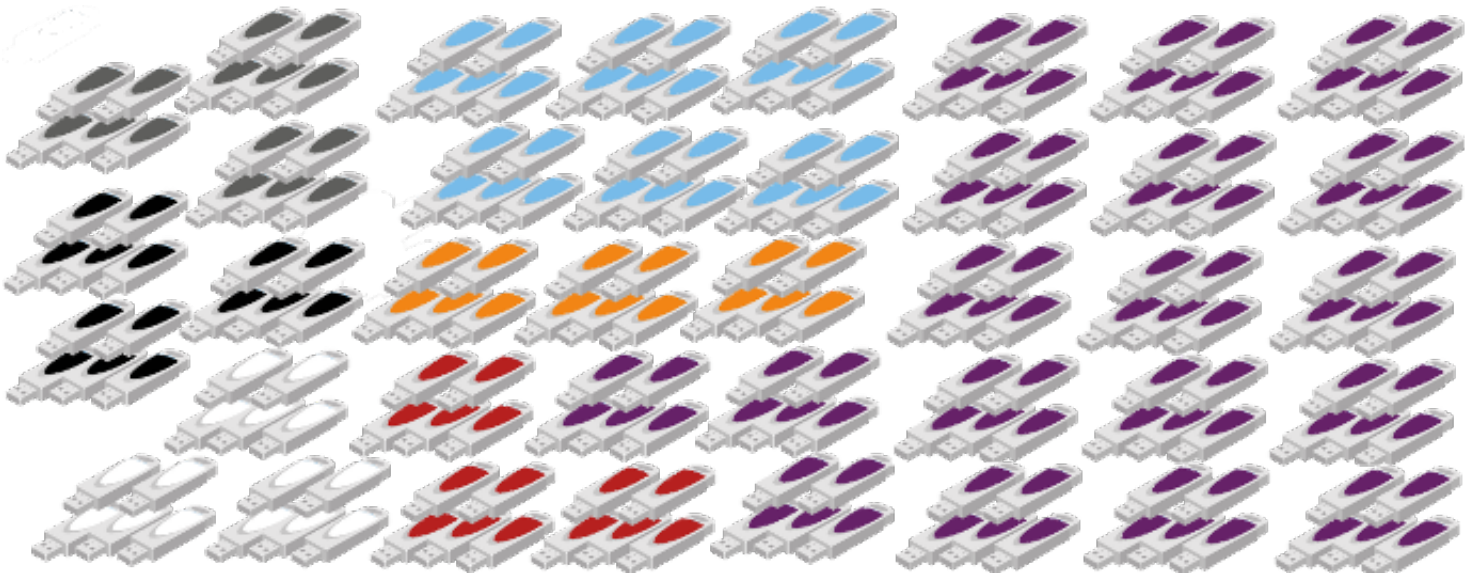
The chart below shows how the two options interact.



In the above chart, we modify the original PED Key grouping scenarios (earlier in this topic, $M=N=1$ so no MofN), by adding a requirement for MofN where the full set of keys for that secret (N) is five splits, and the number required to access that role or function of the HSM (M) is three. That covers just one secret (of the eight or nine) on six HSMs.

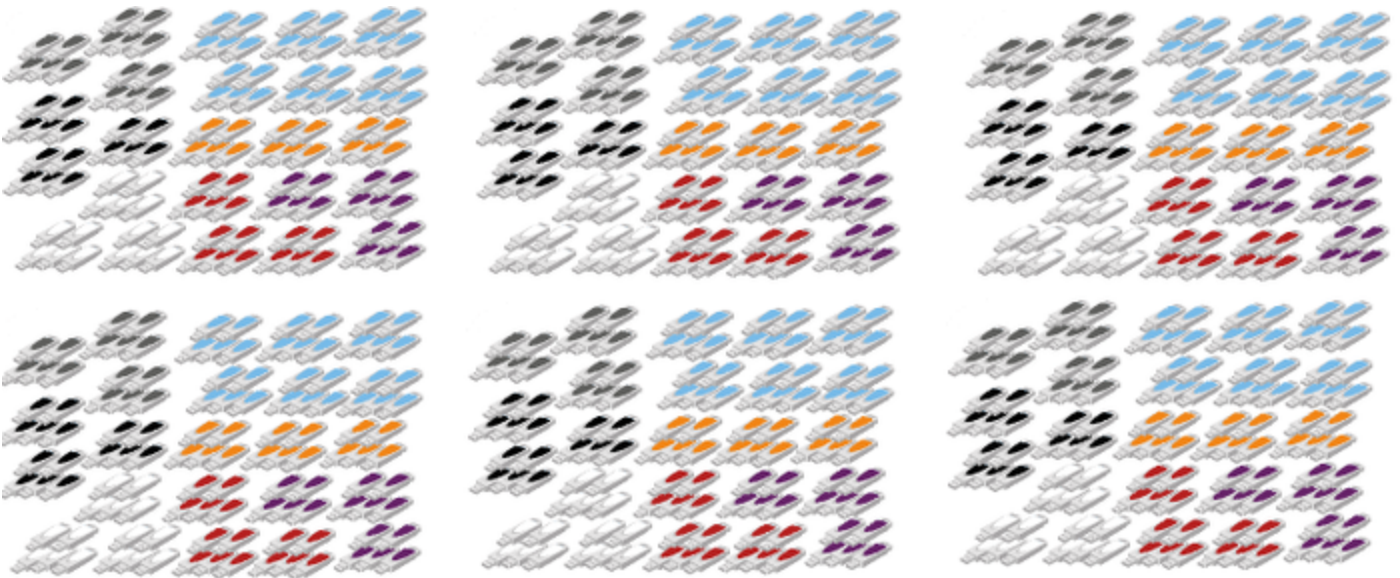
Here is a visual suggestion of how many physical PED Keys you would be dealing with in an example scenario where:

- one HSM is in play (or all secrets are grouped for several HSMs),
- you had invoked every possible role or access-controlled function on the HSM
- you had both an HSM SO and a different SO for one application partition
- you had chosen to apply MofN = 3of5 for every secret
- you were keeping an operational PED KEY set, as well as two backup sets in different locations.



Why are there so many purple keys, when we have grouped all the secrets for our six HSMs? Recall that you can group any of the other HSM access secrets, but the purple SRK PED Key cannot be grouped. The SRK is unique to each HSM. The above illustration shows the smallest number of PED Keys you could have if you insisted on MofN at 3of5 for every secret on the six HSMs that you manage in this scenario. (The value that you chose for "M" is not important in this context. It is the value of "N" that determines how many PED Keys you must manage.) This illustrated example is a good indication that you should carefully consider whether a given HSM access secret really needs to be protected by split-secret multi-person access control (MofN greater than 1of1).

Now, look what happens if you have the six HSMs from our earlier example, and instead of grouping each HSM/secret in a single group of six, you instead elected to have no HSM grouping for any secret.



The above large number of PED Keys represents the number of physical keys you would need to manage for just six SafeNet HSMs with a single application partition per HSM where you had made these choices:

- six HSMs are in play, with no grouping of any of the HSMs for any of their secrets, so each secret is unique for every HSM,
- every possible role or access-controlled function on each HSM has been invoked (*)

- the HSM SO and the SO for the application partition on each HSM are different secrets (so a blue key set for the HSM SO and a blue key set for the partition SO... times six)
- you had chosen to apply MofN = 3of5 for every secret
- you were keeping a complete operational PED KEY set for each HSM, as well as two backup sets in different locations, for each HSM.

Granted, the illustration seems extreme, but it is the inevitable outcome if you were to make the choices that are described.

(* It is very plausible that you might have initialized different domains for the HSM and for the application partition on each HSM, in which case you could have seen an additional bunch of three sets of red keys for each HSM in the above illustration.)

The take-away observations from the examples, as you plan the deployment of your own SafeNet HSMs in your organization, should be:

- if you need a role or an access-controlled HSM function, by all means, initialize it for that HSM, but if a role or function is not needed (for example, Audit or SRK), then leave it uninitialized
- grouping of HSM secrets (re-use of keysets from other HSMs that you control, when initializing roles and functions) is good, as long as there is no countervailing security concern
- MofN is an excellent security measure, but should be invoked only when necessary, to avoid proliferation of PED Keys and the associated management burden.

Example calculation

So, with a primary keyset (for a given role or function) plus a single backup set, if all HSMs have unique secrets ("Reuse..." was chosen as "No") then you need:

$$N * G * S = T$$

where:

N is the number of MofN splits, which in this example is 5

G is the number of groups (which is 6, in our example, if all 6 HSMs have a unique secret, no grouping, or which is 1 if all 6 HSMs use the same secret for that role)

S is the number of sets (which in this case is a primary set plus one backup set, or a total of 2)

T is the Total number of PED Keys needed for just the one role or function for your six HSMs

$5 * 6 * 2 = 60$ is the largest number, with MofN at 3of5 and no grouping (no reuse of this secret across HSMs)

If all six of the HSMs use the same keyset for that role (a single group), then the number of PED Keys required to manage is greatly reduced:

$5 * 1 * 2 = 10$ in total.

That was for one secret to access a role or function of the HSM.

Calculate PED Key requirements for some or all HSM authentication secrets

The calculations above were examples for just one of the eight or nine secrets that apply to your HSM and its partitions. If more than one authentication secret is split for multi-person access control, then the number of PED Keys that you must manage grows dramatically.

Note: We strongly urge methodical handling and labeling of your imprinted PED Keys, in order to track them and to avoid mixing members from different sets.

The PED Key secret that the HSM tests, for authentication to an HSM role or function, is attempted only when it is complete.



Without MofN, the secret is complete on one PED Key. Presenting a PED Key from a wrong set, when MofN is not involved, can result in lockout of a role, or zeroization of content, depending upon which secret is attempted. For SO PED Keys, you have three tries, for other roles the default is 10 tries, but HSM or partition policies can adjust that to a lower number.

With MofN, the secret is complete only when M unique splits from the needed set, are presented and successfully combined. Presenting a PED Key from a wrong MofN set, or presenting the same split twice because members of primary and backup sets were accidentally mingled, does not allow the splits to successfully combine. Therefore that error does not increment the bad-login-attempt counter for that secret. Instead, it results in looping prompts on the PED until it gets enough of the correct splits or until the operation times out.

Maintaining Your PED Keys

Other than tracking

- where PED Keys are,
- to which HSMs they apply,
- who is in possession of, or has access to, each PED Key,

maintaining your PED Keys means

- making new copies if any suffer damage, and
- ensuring that the secrets on the keys, and any associated passwords (PED PINs, client challenge secrets) are updated in a prompt and orderly fashion, in case of suspected compromise, and in compliance with any "password-change" security policy requirements at your organization.

That last item, adherence to a policy of frequent password changes, could be the most intensive. Consider some of the examples, earlier in this topic, and how much logistical and organizational effort would be involved in a mass password change for one secret. In one example, assuming that you had all six of your HSMs in one operation center, that would be ninety PED Keys located in two different locations at your operation site (the operational set and the on-site backup set), as well as at a distant secure off-site lockup. So that would be three sets of five keys for that secret (so 15 keys for one secret for one HSM), multiplied by the six HSMs in the no-grouping scenario (so 90 keys for one secret). Now consider that one iteration of the password-change process would cover one secret (like HSM SO, blue PED Key) and would leave an additional six, or seven secrets (red, black, gray, orange, purple, white, as well as another blue key secret if your application partition has its own SO) potentially needing to be updated.

These considerations are important when you develop your authentication and security schemes around SafeNet HSMs and PED Keys.

Conclusion

With all of the above in mind, it is not possible to suggest one "correct" number of PED Keys for your situation. It depends upon the choices that you make at several stages. In all cases, we repeat the recommendation to have at

least one backup in case a PED Key (any color) is lost or damaged.

HSM grouping (PED Key reuse) reduces the number of PED Keys that must be managed, but puts more than one HSM at risk if a PED Key is compromised. MofN increases the number of PED Keys that must be managed, but increases the security of all affected HSM roles or functions by ensuring that no single key-holder can act unilaterally.

Using MofN

MofN is designed to provide additional 'eyes' on the setup and deployment of an HSM in a customer environment. The feature implements a balance between this multi-person control and the requirement for these MofN key holders to be present for all operations. For a description of what MofN is, and how it works, refer to the Product Overview document (see ["About MofN" on page 1](#)).

Typical Practice

Three kinds

- The typical deployment of a SafeNet Network HSM appliance is for it to be installed in a secure area of a data-center, typically near the certificate servers that it is servicing. Customers demand that this appliance is secure, but alongside that requirement they need to ensure that their processes and procedures aren't hindered by the addition of this HSM - this is the age-old security-versus-usability discussion.
- The typical deployment of a SafeNet USB HSM is either attached to an application server, perhaps to serve as the root of a PKI, or attached to a SafeNet Network HSM appliance to serve in a similar capacity as part of a "PKI bundle".
- The typical deployment of a SafeNet PCIe HSM (K6) HSM is inside its application server - again, as the root of a PKI, or as the cryptographic engine to an application on that server.

It is frequently the case that the HSM and its server(s) are kept in a locked facility and either accessed remotely by secure channels or accessed directly and physically only under specific conditions.

To satisfy these design requirements we have a concept of Partition Activation (["About Activation and Auto-Activation" on page 167](#)). This allows administrators of the HSM to put it into such a state that the calling application is responsible for its own connections and sessions with the HSM, without requiring the presence of the operators for each and every login. This is important when an application or operating system might be rebooted for maintenance, or a power outage might occur (up to two hours duration), and it would be challenging to get the 3 or 5 management personnel together to present the MofN keys.

Another way to describe this might be:

- The black PED Key(s) is presented in order to set the partition into a state of "open for business".
- When that is true, clients can connect.
- Clients must still provide their own credentials (certificates were exchanged, to register the link) and present a challenge secret or password (previously distributed) to enable them to perform cryptographic operations on the partition.
- At any time, the holder of the partition User/Owner black PED Keys can close the partition to access (deactivate it) and clients can no longer access the partition, regardless of their registered status and their possession of the challenge secret.

Common MofN Usage

A common customer scenario would see the HSM configured and brought into production at a data-center. This activity would need, first, the quantity M holders of blue HSM SO PED Keys, so that the HSM administrator could log in and create partitions, adjust policies, and so on. Then, in the legacy partition model, quantity M holders of black User PED Keys would be needed in order to activate each partition, making it available for customer connection. At this time the key holders (who would typically be management personnel, rather than day-to-day operational personnel) would give their approval to access the HSM by presenting the M keys at first login, or first partition activation. This is the electronic equivalent of them 'signing off' that the HSM is properly installed where it should be, that the security officer, partition owner and cloning domain holder - as well as the PIN holders if separate - are the correct authorized personnel.

In the PPSO model of operation, you would add a second set of blue SO-split PED Keys, probably different from the HSM SO set.

MofN is optional until you decide to invoke it when a secret is first created, and it is optional per secret. But after MofN is invoked, it becomes mandatory for that role or function. That is (for example):

- You could choose not to invoke MofN for any HSM authentication secret - so only one blue SO key, and one black Crypto Officer key, one gray Crypto User key, one red cloning key, one orange Remote PED key, and one purple Secure Recovery Key, would be needed to access the respective HSM functions and roles. A single person, per role, would be able to perform each function without oversight.
- You could choose to invoke MofN for some secrets and not for others. For example, HSM-level access could be configured to require multiple blue PED Keys while, say, the partition-level access needs only one black PED Key. The HSM security officer would need M people to agree that she/he had the right to log into the HSM, each time, but any individual partition owner/User could activate her/his own partition with no oversight. The reverse could also be true, with the SO needing just a single blue key for HSM login and HSM administration, but the various partition owners needing multiple persons with black key splits to activate or deactivate their partitions, change passwords, etc.
- You could invoke MofN for every role, but set different M and N values per role. HSM administration might have a pool (N) of 5 blue keys and need 3 (M) of them for any HSM login event. Meanwhile the pool of black keys (N) for a given partition might be 3 or 6 or 10 or as many as 16, but the number of holders (M) needed to activate the partition might be just 2 (or any number up to N)... and so on, in as many combinations and permutations as make sense for your situation. Similar choices would apply for red, orange and purple key secrets and for the Audit role. As well, while you can choose to reuse a black PED Key (or an MofN set of black PED Key splits) to create and access multiple HSM Partitions (on a single HSM where permitted, or on different HSMs), you could also choose to imprint a different black PED Key secret (or separate MofN sets of black PED Key splits) for every partition, or any combination of those options.

MofN and PED PINs

In addition to the "something you have" authentication factor, each secret-share can also (optionally) have a "something you know" authentication factor. That is, for every split of every HSM secret, you have the option - or not - to declare a PED PIN ("[What is a PED PIN?](#)" on page 195) that must be entered at the keypad when that PED Key is presented.

As with MofN, the PED PIN secret is an option that is chosen via the PED. For each key that is imprinted, you are given the option to set a PED PIN secret (typed on the keypad) in addition to the secret contained inside that PED Key. As each PED Key is unique, it can be given:

- no PED PIN
- the same PED PIN as other members of a set

- a completely different PED PIN.

As you can imagine, combining permutations of MofN with permutations of PED PINs could make for a very complicated security scheme. You have these options; it is up to you to choose and combine them in ways that meet your security needs without over-complicating the lives of your personnel.

Revoking Means Re-initializing

Once MofN is set (either M=1 and N=1 for no multi-person access, or M and N each larger than 1 to invoke multi-person access control), that setting remains in place until the HSM or partition is zeroized and re-initialized.

So, if you decided that you wanted to stop using/requiring MofN, or that you wanted to have MofN, but with a different total number of split-keys (N) or a different minimum quantity of keys that must be presented (M) to re-construct the secret (blue, black, red, etc.), then you would need to zeroize (factory reset) and re-initialize the HSM. Or for just individual partitions, you would need to delete the partition and create a new one with the new authentication. To preserve the HSM and partition contents, you would perform backup before re-initializing the HSM, and then restore from backup afterward.

Red keys

Once an object is protected by an MofN cloning domain - red PED Key - that MofN protection is in force for the life of the object. Other role secrets can be changed (including their MofN settings) by initializing the HSM or [re-]creating a new application partition with new blue or black PED Key secrets.

But you cannot clone between differing cloning domains (red PED Key secret) - that is the whole point of domains, to ensure no loss of control of objects. This means that any cloning operation, including

- backing up,
- restoring from backup, or
- synchronizing within an HA group

requires a common cloning domain (red PED Key secret). You can't change the cloning domain for an existing object, and that is a designed security feature.

How to Add an MofN Requirement Where There Was No MofN Before

Historical Note:

On SafeNet Network HSM 4.x systems, if one HSM had MofN (using the legacy green PED Keys), and you wanted another HSM to use the same MofN splits, you had the option to clone MofN from the first HSM to the second.

Current Practice:

With SafeNet Network HSM 5 (containing the K6 HSM keycard) and newer, where MofN is a condition of each authentication secret independently, there is no provision to "clone MofN". Instead, if you wish to have two HSMs share the same MofN scheme, you must initialize one with the desired scheme, then initialize the second (and any additional) HSM and have it re-use the secret splits from the first HSM.

At secret-creation time for the HSM, when the PED is invoked, the PED asks if you wish to re-use an existing secret. If you say "yes" to that question, then the PED expects you to offer a PED Key (for example a blue PED Key, when you are initializing) that is already imprinted with a suitable secret. If you offer a blue key that contains a partial secret - a split from your other HSM - the PED accepts that key. The connected HSM recognizes that the secret is only a split, not a full SO secret, so the PED demands additional keys from that set, until it has received M of them, enough

to reconstitute the secret. It will not accept fewer than M different portions of the secret, and it will not accept members of another set.

Once the reconstituted secret has been imprinted on the new HSM, then that HSM can accept any M splits out of the full set of N, even though it has never seen some of those splits. Both HSMs now accept the same MofN authentication secret. You can do the same, individually for any of the other secrets on the new HSM (black partition User keys, red cloning Domain keys, orange RPV keys). The only exception is the purple PED Key (or Keys), since the MTK and SRK are unique to each HSM and cannot be cloned or shared.

Purple Keys

You *can* duplicate a purple PED Key while you are in the process of imprinting it (SRK enable, SRK resplit).

You *can* split the purple-key secret (which is already one split of a larger secret inside the HSM) so that the Secure Recovery Vector secret needs multiple purple key holders to invoke it.

You *can* re-split the internal MTK of your HSM, resulting in a new SRV portion imprinted on the external purple key (or keys, if M and N are greater than 1).

You *cannot* generate a new master secret on the HSM - the MTK is unique and permanent for each HSM and can be changed only by re-manufacturing. Factory reset and initialization have no effect on the MTK.

You *cannot* imprint a purple key secret from one HSM onto another (for the same reason as above), unlike all the other key colors where sharing/grouping are important options.

You *cannot* duplicate a purple PED Key via the PED's stand-alone (no HSM present) Admin menu. The "raw" duplication function, which works for all other PED Keys, refuses to duplicate purple keys. This is a security feature, so that no one can duplicate a purple key without access to the HSM that created it. This applies to splits of the SRK as it applies to a single SRK purple key.

Implementation Suggestions for backup MofN sets

Here is one suggestion for having the security benefit of MofN, including backups, but without the risk of accidentally mixing members of original split set and backup split sets.



Note: The risk is not that members of "original" and copy sets can't work together - they do - the risk is accidentally having copies of the same split-containing key together. The PED requires different splits when combining quantity M splits to recreate the authentication secret. If you offer it one split and then a copy of the same split (because they all look alike and you accidentally gathered them into incorrect groups), the PED will reject the identical offering because it correctly observes that you are offering the same split twice.

Say, for example, that you needed an MofN set to control access to an HSM partition. You want partition access to require the presence and cooperation of 3 black PED Key holders, and you want a couple of additional splits for two alternate officers to carry, in case not all of the original three are available. So you specify MofN to be 3 of 5. You know that mishaps can occur, so you want to have a full equivalent set of black PED Key splits as backup, in case one or more of the originals is lost or destroyed. These could be stored in a secure on-site lockup, ready when needed, but requiring higher authority to release them. So you would have an original set of black PED Key splits for that partition, and a full set of duplicates.

You also want to ensure that you have a fallback in case of disaster at your operation location. So, you want a second complete backup to be held in a secure off-site lockup.

All together, you have three identical sets of five different partial secrets, for a total of 15 black PED Keys. They must be carefully labeled and handled, because you need to avoid mixing the members of different sets.

Make one big MofN set, instead of three small sets

If your M and N numbers are small, like (say) 3 of 5, simply declare a large N (up to 16 splits is permitted, so in this case use 3 of 15) and simply gather them into groups of (say) 5, one group for regular operations, one group for standby, one group for off-site backup storage. In this way, all the splits are valid together in any combination- any three of the 15 can unlock the HSM. You do, of course, need to control distribution of, and access to, all those secret-split keys.

If your M number is larger, then this idea becomes less practical, since you have a maximum N of 16 to work with. It depends on how many sets of M you need. At the very least, you should have one backup of every HSM authentication secret, preferably in secure off-site storage.

MofN is not for everybody. For those who need it, it is crucial, and the added administrative task is a "cost of doing business". If you don't need MofN in your security regime, then we suggest that you not use it.

If your security policy demands that you use MofN multi-person access control and also demands that M be relatively large, consider carefully if your policy might need review. Any security regime should be no more complicated than it needs to be - no more complicated than yields a net-positive security benefit. The more complicated or onerous a security policy, the more your own personnel - even the most trust-worthy - are motivated to circumvent or simplify, in order to get on with their tasks.

Complexity When Managing PED Keys

These options, to create group PED Keys and duplicate PED Keys, can introduce complexity and another kind of risk to the management of PED Keys, especially when the options are combined. In many establishments, security policy demands that passwords be changed on a regular basis. Naturally, passwords/PINs on HSMs, Partitions and tokens and PED Keys can be changed as needed.

However, what might be a simple procedure for a single key (Change PIN) can quickly take on new dimensions when there might also be a backup PED Key in off-site safe storage, and there might be several working copies of the PED Key in the hands of Owners, alternate/backup Owners, and alternate/backup HSM Admins. Additionally, there might be several tokens, Partitions, or HSM Servers that are unlocked by any of the PED Keys (if you chose the group PED Key option when creating any of those).

The issue is that when authentication data is changed on a PED Key, it must be changed on the associated HSM or Backup Token at the same time; otherwise the two no longer match and the PED Key can no longer unlock the HSM (or Partition) or the token. The changePw procedure does take care of this for the HSM (or Partition) or token and for the blue (or black, as appropriate) PED Key that is in the SafeNet PED slot when the change command is issued. There is also provision (explained in following pages) for having other accessible PED Keys updated, during the procedure, to maintain synchronization with the HSM (or Partition) or Backup Token.

But, what about the set of backup PED Keys that you have sensibly stored off-site? If they are not brought in and updated during the same update procedure, they are no longer backups. Your security and maintenance procedures must address this situation.

To ease the task of updating multiple PED Keys, without a complicated dance involving all sets during the same update event, the PED provides a method of stand-alone, "raw" key duplication.

[<] to exit Local PED mode to the main PED menu

[4] to enter Admin mode

[1] to enter PED Key mode

[1] again to bypass key login, which is not applicable to the iKey 1000 model in current use and then

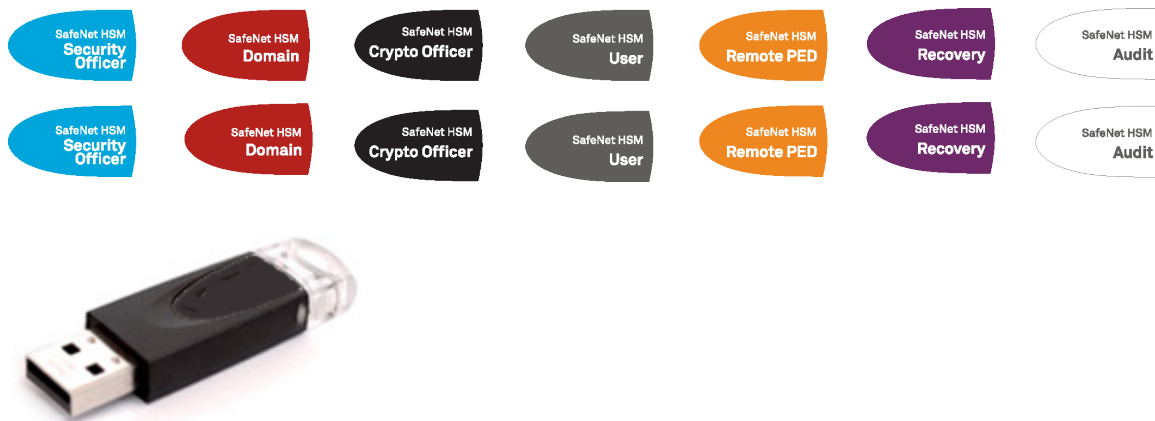
[7] to duplicate whichever key is presented next.

This is applicable to all imprinted PED Keys except the purple SRK (excluded for security reasons).

Because the above is a "raw" duplication, there is no opportunity to modify any PED PIN that is already associated with the presented source key. Duplicates by this method are exact.

Once you have updated any or all members of a working set of PED Keys, you can take one of those keys and a PED to any other location where duplicate sets were maintained (onsite backup, offsite backup, etc.) and update your backups without any need to involve the HSM. Always be aware of the location and state of any SafeNet PED key, and keep scrupulous records of all changes and hand-offs. Your security auditors will thank you.

General Advice on PED Key Handling



In addition to the cardinal admonitions about careful physical security and prompt, thorough backups of your HSM partitions and PED Keys, here are some practical tips to make the tasks as easy as possible.

Keep a Log

Keep careful records, both of the regular backup procedures, and of who has possession of any token and any PED Key at any time. Your records should show every hand-off or change of possession and your policy should enforce it. Proper security protocols demand that you be able to account for all primary devices (HSM Servers, tokens and PED Keys) at all times, without exception. Establish strict procedures governing when and how those devices may enter storage, be removed from storage, or change hands among users.

When performing backups and other maintenance functions (such as changing PINs on keys and HSMs), log the event, but also keep a worksheet of notes so that if the task is interrupted you can resume it without confusion or hesitancy as to which devices have been altered and which have not. To help in that regard, see the next section.

Apply Meaningful Labels

This suggestion has two aspects relating to everyday handling convenience and to the previous section, "Keep a Log":

1. Apply text-string labels to your HSM Servers and tokens.
2. Apply physical labels to the exterior of the physical devices.

In the first case, a unique, easily identifiable word or phrase serves as a final check in `lunash` or at the client when you are about to perform an action that could alter an HSM, a token or its contents. You might consider a label

consisting of a part (perhaps a word) that identifies the domain to which the token belongs, and another part (perhaps another word or a number) that identifies it as a particular member of that group.

The second case, physical labels, applies to HSM Servers and PED Keys.

When handling multiple HSMs and keys, it is easily possible to become confused as to which ones have been updated and which ones are yet to be updated. Worse (if you are using common administrative group PED Keys) would be restoring onto the wrong Partition or HSM Server, from a backup.

General physical handling is made easier if you have a way to identify a device visually. Easy identification facilitates log-keeping.

Do not cover or obstruct the connector end of a PED Key.

Do not permanently obscure the appliance serial number. You would want it visible in the unlikely event that you ever needed to contact SafeNet for assistance.

Keys

PED Keys have different roles. Colors help to easily distinguish the roles and you should use the labels included with the product (blue, red, black, orange, and purple) to mark PED Keys before you initialize them. The additional suggestions on this page are about applying *additional* labels (stickers, tags, other) of your own, to identify specific keys and key sets and where they fit in your operational scheme.

The PED Keys might further be in need of visual identifiers if you elect the MofN option, which adds several, visually-similar keys to the mix. It might be useful to identify the following:

- Which keys (blue, black, red) are associated with which HSMs or Partitions or Clusters).
- Which black keys are associated with which Partition and client. It normally makes sense to associate a key to a title or function, rather than to a specific person.
- Which key is which in an MofN group. This is particularly useful when the SO is initializing HSMs and keys (and could be accomplished by temporary labels in that situation).

You must decide whether visual identifiers of MofN status of keys would be useful once the keys and tokens are in operation (or in backup safe storage), or whether your security requirements would prohibit such tags or markings.

Updating PED Keys – Example

The following is just an illustrative example of changing PED Keys (or the authentication secrets on the PED Keys and the corresponding secrets on HSMs). For the purposes of the example, we will ignore additional complicating factors like PED PINs and MofN that might apply to your situation.

Say, for example, that you had shared PED Keys among three HSMs, and that you also made three other copies of that SO PED Key, so that you and two other persons could each work with one (or any) of the HSMs, and so that the fourth PED Key could be stored away securely.

Risk of Losing access

If you were to “Change PIN” for your own PED Key (and your HSM), then that PED Key would work for that HSM, but the PED Key would no longer work for any of the other HSMs and none of the other PED Key holders of your group could access your HSM. Your HSM would expect the new PIN, and the other people would be holding PED Keys with the original PIN.

Immediately, you see that any time you change passwords (PINs) it must be done for all HSMs (or Partitions) in such a group, and for all PED Key duplicates associated with that group of HSMs (or Partitions if you are changing black User PED Keys).

PIN-change Procedure for Multiple HSMs



CAUTION: You must retain at least one old-PIN PED Key until all HSMs have the new PIN, or you will find yourself unable to access old-PIN HSMs.

1. Choose an HSM and login as SO (with a blue PED Key).

2. Request a change of SO PED Key:

```
lunash:> hsm changePw
```

3. Respond to the PED prompts as follows:

```
Getting current SO PIN...
Reading SO PIN...
Insert a blue Key
```

This is where you insert a currently valid SO PED Key to confirm that you are the key holder.

```
<Press ENT>
```

The PED requests the key because an indeterminate amount of time might have elapsed since the last HSM login and confirmation is needed that the person asking for a change of secret is the person who logged in (and not an unauthorized person taking advantage of an unattended login session).

```
Reading SO PIN
Please wait..
Would you like to reuse an existing keyset? (Y/N)
```

Here you respond "NO" so that a new SO secret is generated.

```
M value (1-16)
>0
M value (1-16)
>0
Writing SO PIN...
Insert an SO Key
```

This is where you insert the first SO PED Key to be overwritten; it might be the same one that you just inserted to authenticate as SO

```
<Press ENT>
Writing SO PIN...
PED Key will be overwritten
```

The PED detects existing (old) data on the key and warns you that it will be overwritten if you proceed.

```
<Press ENT>
Writing SO PIN...
Enter new PED PIN
```

This is a new secret, so you have the opportunity to add a PED PIN to it, if you wish.

```
Writing PED PIN...
Confirm new PED PIN
```

Are you duplicating this keyset? (Y/N)

Answer "YES" because you want to overwrite the old secret on two of the remaining three PED Keys (in this example).

```
Writing SO PIN...
Insert SO key
```

This is where you insert the second SO PED Key

```
<Press ENT>
Writing SO PIN...
PED Key will be overwritten.
<Press ENT>
Writing SO PIN...
Enter new PED PIN
```

You can add a PED PIN to this duplicate key if you wish, or not. If you add a PED PIN it does not need to be the same as on the other key.

```
Writing PED PIN...
Confirm new PED PIN
Would you like to
make another'
duplicate set? (Y/N)
```

Respond "YES" and make the change on the third SO key, but leave the fourth key with the old secret for now.

```
Command Result : 0 (Success)
[luna22] lunash:>
```

At this point, you now have ONE HSM and three of your four SO keys imprinted with the new SO authentication secret. Ensure that you keep the keys separate and well identified. One PED key MUST retain the old secret until all HSMs are updated to the new secret.

4. Go to the second of your SafeNet appliances, login as admin.
5. Request a change of SO PED Key (this time you will not be changing key contents, you will be logging in with the old secret, then copying the new secret from one of the updated keys onto the second HSM):

```
lunash:> hsm changePw
```

6. Respond to the PED prompts as follows:

```
SO login...
```

This example step shows that if you had not already logged in prior to requesting "hsm changePw" then a login is forced.

```
Insert blue PED Key
```

Insert the old-secret PED Key, to login -- this HSM still has the old secret.

```
<Press ENT>
Getting current SO PIN...
Reading SO PIN...
Insert a blue PED key
```

The system does not track how long ago the login occurred, so before a key change is permitted, it requires you to prove that you are the valid keyholder, by producing the key again.

```
<Press ENT>
Reading SO PIN
```

```
Please wait...
Setting SO PIN
Would you like to
reuse an existing
keyset? (Y/N)
```

Here you respond "YES" so that the new SO secret will be read from the new-secret-containing key that you are about to insert.

```
Reading SO PIN...
Insert a blue PED Key
```

This is where you insert a new-secret SO PED Key so that its secret can be read and then imprinted on this second HSM.

```
<Press ENT>
Would you like to
make another'
duplicate set? (Y/N)
```

Respond "NO". This HSM now has the new secret.

```
Command Result : 0 (Success)
[luna22] lunash:>
```

At this point, you now have TWO HSMs and three of your four SO keys imprinted with the new SO authentication secret. Ensure that you keep the keys separate and well identified. One PED key MUST retain the old secret until all HSMs are updated to the new secret.

7. Remove the new-secret key from the PED and place it with the other new-secret keys.
8. Bring a PED and the remaining old-secret key to the third appliance and login as admin.
9. Request a change of SO PED Key (you will be logging in with the old secret, then copying the new secret from one of the updated keys onto the third HSM, then overwriting the final old-secret key with the new secret, once the old secret is no longer needed).



Note: You can explicitly login (with "hsm login") before issuing "hsm changePw", or you can wait until you issue the change command and be prompted to login.

```
lunash:> hsm changePw
```

10. Respond to the PED prompts as follows:

```
SO login...
Insert blue PED Key
```

This prompt appears if the HSM was not already in the login state. Insert the old-secret PED Key, to login -- this HSM still has the old secret.

```
<Press ENT>
Getting current SO PIN...
Reading SO PIN...
Insert a blue PED Key
```

Here, the PED wants the same secret that you used to login.

```
<Press ENT>
Reading SO PIN
Please wait...
Setting SO PIN
```

```
Would you like to
reuse an existing
keyset? (Y/N)
```

Here you respond "YES" so that the new SO secret will be read from the new-secret-containing key that you are about to insert.

```
Reading SO PIN...
Insert a blue PED Key
```

This is where you insert a new-secret SO PED Key so that its secret can be read and then imprinted on this third HSM.

```
<Press ENT>
Would you like to
make another'
duplicate set? (Y/N)
```

Respond "YES", and supply the last old-secret PED Key as the "blank".

```
Command Result : 0 (Success)
[luna22] lunash:>
```

At this point, you now have all three HSMs and all four SO keys imprinted with the new SO authentication secret.

If you prefer to be more cautious, you could have left the final PED Key with the old secret until you verified that all three HSMs are now unlockable by the new secret, and only then invoke the command one more time to imprint the last key with the new secret.

Alternatively, on a SafeNet PED 2.x, you can perform iKey PED Key copying or duplication at the PED without invoking commands at the HSM (however you still require a connection between PED and HSM to power the PED).



Note: You can perform the same operations with blue SO PED Keys, in similar circumstances, and observing the same precautions. Also, this sort of operation could be scaled up for larger groups of HSMs (if they share a group-User or group-SO PED Key) and for larger numbers of duplicate PED Keys.



Note: To avoid confusion, it's probably best if you mark each key to identify it, and keep a careful log of which key and which HSM has what operation done to it, at each step.

Updating PED Key for a Backup Token

There is no explicit provision for changing the authentication for a Backup Token. If you need to have new authentication for your Backup Tokens, then perform a new Backup operation.

Performing an HSM Backup or a Partition Backup will initialize the token and allow you either

- to imprint a new authentication secret (say "NO" to the "reuse ID" question, which causes a new random secret to be created and imprinted on both the PED Key and the token),

or else

- to share the authentication secret (say "YES" to the "reuse ID" question, which takes the token authentication from the PED Key that you insert, and not the other way around) that is already in use on other tokens, or on a SafeNet HSM.

Frequently Asked Questions

This section provides additional information by answering questions that are frequently asked by our customers.

How should SafeNet PED Keys(*) be stored? (*Model iKey 1000 for use with SafeNet PED2)

Physically, they are electronic devices, and should be stored in environments that are not subjected to extremes of temperature, humidity, dust, or vibration.

With that said, PED Keys that have their protective connector-caps in place are quite robust. PED Keys that have their caps on when not immediately in use have survived years of daily use being carried around in office-workers' pockets, here at SafeNet's labs.

Procedurally, they should be labeled and stored (filed) so that they are readily identifiable according to the HSM(s), the partitions, and the roles with which they have been associated.

So I shouldn't keep all the PED Keys for all my SafeNet HSMs in one box in a desk drawer?

No. The only place where that might be appropriate is in a test lab where HSMs are constantly re-configured for test purposes, and where they never contain important cryptographic material. PED Keys are just generic iKeys until you make them into specific kinds of PED Key by your administrative actions with SafeNet HSMs and SafeNet PED. Once a blank iKey has been turned into a Security Officer (blue), Domain (red), Partition Owner/Partition User (black), Audit (white), Secure Recovery (purple), or Remote PED (orange) key, you must ensure that it is labeled as such and that you handle it and store it in a way that it can never be mistaken for a different PED Key.

We have had at least one customer call us in a panic because they had "lost" the SO and Domain and Partition keys to an enterprise-critical HSM. They actually still had those keys, mixed with many others in a box. As you know, SafeNet HSM authentications do not permit a lot of "guessing". For example, you get only three tries to present the correct blue PED Key for HSM SO login, before the HSM contents are lost forever. A customer staffer, new to her job asked: "You make the HSMs and keys, why can't you just give us another one?" We had to explain that there is no 'back door', ever, to a SafeNet HSM. We (SafeNet) did not make her PED Keys. We made the iKeys, and her predecessor created them as the PED Keys for her organization's HSMs.

Fortunately, that customer's critical HSMs were in an HA configuration that had not yet synchronized, and the secondary HSM was still in logged-in state. After trying several red PED Keys it was possible to get a backup of the secondary HSM and restore onto a re-initialized primary HSM. After that, our responding support engineer spent many hours teaching the customer staffers the basic security and HSM administrative knowledge that had been lost due to staff turn-over at that company. That enterprise customer has since installed rigorous procedures and documentation for handling of HSMs and HSM authentication secrets.

I've lost my purple PED Key. Or, I forgot my PED PIN for my purple PED Key.

You are likely in for some cost and disruption, but this is not necessarily a fatal mistake.

At the present time (this note is written in November 2014) there is no way to recover from a tamper or from Secure Transport Mode if the external split of the Master Tamper Key (the SRK) is not available. If you haven't got a backup purple key, your HSM is locked the moment it experiences a tamper event, or if it was placed in Secure Transport Mode. The same applies if you do have the key, but have forgotten/lost the numeric PED PIN that you applied when the purple key was imprinted with the Secure Recovery Vector (the external split of the MTK). Either way, you must obtain an RMA and return the HSM to SafeNet for re-manufacture. All HSM contents are lost.

As with every PED Key that you imprint, we recommend that you make at least one backup copy of the purple PED Key, as well. If you can find that valid backup purple key, you can recover the HSM and make a new split, without problem. If the purple key that you lost was the only one... then see the preceding paragraph.

Note that simply not having the external MTK split available is not the end of your HSM and its contents. As long as it has not been tampered, or was not placed into Secure Transport Mode, then the HSM is still working and is perfectly accessible to other key-holders. However, you should immediately back-up all important HSM contents to other HSMs and have SafeNet re-manufacture the affected HSM. When that HSM is returned to you, it will be in one of two states:

- a) it will have both MTK splits internal (no SRK created), or
- b) it will have a new MTK and a new SRK (purple PED Key) if you requested that we ship the HSM to you in Secure Transport Mode.

In the first case, you have a "new" working HSM and can decide what you wish to do with respect to SRK - if it is not necessary to your security regime, simply never declare an external split and you will never need to worry about purple keys. Tamper events (if any) will be logged, but will recover automatically when the HSM restarts.

In the second case, you receive the HSM back from SafeNet, in STM (as requested) and you receive the associated purple key (SRK) by separate courier. You recover the HSM from Secure Transport Mode. At that point, you can elect to disable SRK (return the external split inside the HSM, simultaneously generating a new internal split pair, and invalidating your purple key). OR, you can elect to make a new external split. This imprints a new purple key (SRK) and invalidates the one that we shipped to you. You should make at least one backup copy of the new purple key when it is created, and take better care of your imprinted PED Keys in future.

Also, if your security regime does not require multi-factor authentication, then see the next question, about PED PINs.

Do we really need to include a PED PIN with each PED Key?

Not at all. Or, rather, you do if you already set a PED PIN when you initialized/imprinted that PED Key. But a PED PIN is an optional item when you first initialize an HSM or create a partition, etc. You have the choice, and you don't want to impose a PED PIN requirement on yourself without good reason.

A PED Key is single-factor physical authentication - "something you have". If that is sufficient to satisfy your organization's security requirements, then you do not need to impose PED PINs.

You can just press [Enter] on the PED keypad when the PED Keys are being imprinted (that is just press the [Enter] key with no digits), and you would never be troubled by a PED prompt about PED PINs again.

PED PINs are an option - until one is imposed; then it becomes mandatory. Only if your security regime requires two-factor authentication should you consider applying PED PINs to your various PED Keys. Where the physical PED Key is "something you have", the PED PIN is the second factor, the "something you know". A PED PIN is a convenient and effective second factor, but it does represent an additional item for you to remember and to track.

If you lose track - if you fail to remember a PED PIN, or if you have several and don't remember which is which - you can find yourself locked out of your HSM or your HSM partition as surely as if you lost the physical PED Key. More surely, in fact, since you probably have physical backups of your PED Keys (you do, don't you?). Remember, typing a wrong PED PIN on the PED's keypad is the same as offering the wrong physical PED Key to the HSM. It counts as a bad login attempt. PED PINs are good and essential when you need one, but they are not something to impose without a solid security-based requirement.

This chapter describes the actions you can take to maximize the performance of your HSMs. It contains the following sections:

- ["Performance Overview" below](#)
- ["HSM Information Monitor" on the next page](#)
- ["Performance and the PE1746Enabled Setting" on page 258](#)
- ["Frequently Asked Questions" on page 258](#)

Performance Overview

SafeNet Network HSM 5.x/6.x has a newer generation internal HSM; for discussion purposes, SafeNet refers to this HSM as K6. SafeNet refers to the HSM inside SafeNet Network HSM 4.x as K5. Both K5 and K6 rely on application-specific integrated circuits to accelerate cryptographic operations within the HSM. Each generation uses different ASICs. These ASICs use multiple “engines” – analogous to the processor inside a computer having multiple central processing units – to spread load and thereby increase performance.

With SafeNet Network HSM 4.x, a client application needs to create about 20 threads to achieve maximum RSA signing performance based on 1024-bit RSA operations. At this number of threads, within the K5, the ASIC achieves optimal distribution of cryptographic operations across its multiple engines. The ASIC within K6 has a different number of engines and a different algorithm for distributing load. To achieve maximum performance with SafeNet Network HSM 5.x or 6.x, a client application needs to create about 50 threads. With refinements made for SafeNet Network HSM 5.2, this number is now about 30 threads, and carries through for SafeNet 6.x.

Published performance figures for SafeNet Network HSM generally reflect repeated single operations against a single object that is imported or looked up one time before all the operations are performed. This is the most advantageous situation, under the best conditions to yield the highest attainable speed with the equipment. All manufacturers take the same approach.

"Real life" performance figures are often lower because of additional overhead, such as where an object must be fetched before each operation, or where the current task switches constantly from one operation type to another (example sign-and-verify in combination).

If you are using (say) the supplied multitoken tool in a lab setting, note that it defaults to a packet size of 1 kilobyte for symmetric encrypt/decrypt operations, a modest size that imposes a significant overhead. To obtain performance closer to "real life" for your situation, the test packet size should be modified to match the sizes that you expect to see in your intended application. For example, a packet size on the order of 256 bits for credit card numbers versus 64 kilobytes and larger for high-throughput encryption could show significantly different performance.

When HA is considered (two or more HSMs in a redundant group), further overhead is introduced in order to replicate/synchronize across all members of the group. Therefore, the type of operation - whether it requires a single initial replication before a large volume of operations against a static object, or whether it requires a new replication before each single operation - can have a very significant impact on performance.

HA Performance

For repetitive operations, like a high volume of signings using the same key, an HA group can expand SafeNet Network HSM performance in linear fashion as HA group members are added. HA groups of 16 members have undergone long-term, full-throttle testing, with excellent results.

Do keep in mind that simply adding more and more SafeNet Network HSM appliances to an HA group is not an infallible recipe for endless performance improvement. For best overall performance, all HA group members should be driven near their individual performance "sweet spot", which for SafeNet Network HSM 5.2 and later is around 30 simultaneous threads per HSM. If you assemble an HA group that is considerably larger than your server(s) can drive, then you might not achieve full performance from all.

The best approach is an HA group balanced in size for the capability of the application servers that will be driving the group, and the expected loads - with an additional unit to provide capacity for bursts of traffic and for redundancy.

HSM Information Monitor

An HSM administrator might find it helpful to know how busy the HSM is, currently, and at what percentage of its capacity it has been running.

The HSM Information Monitor is a use counter that provides an indication of momentary and cumulative resource usage on the HSM, in the form of a percentage number. The HSM firmware tracks the overall time elapsed since the last reset (Up-Time), and the overall time during which the processor was not performing useful work (Idle-Time).

On request, the HSM calculates "Busy-time" over an interval, by subtracting Idle-time for that interval from Up-time for the interval. Then, the load on the processor is calculated as the Busy-time divided by the Up-time, and expressed as a percentage.

You can use the available commands for a single, one-off query, which actually takes an initial reading and then another, five seconds later (the default setting), in order to calculate and show the one-time difference.

You can specify a sampling interval (five seconds is the shortest) and a number of repetitions for an extended view of processor activity/resource usage. The resulting records, showing the time of each measurement, the percentage value at that time, and the difference from the previous measurement, can be output to a file that you import into other tools to analyze and graph the trends.

By watching trends and correlating with what your application is doing, you can do the following:

- determine the kinds of loads you are placing on the HSM.
- seek efficiencies in how your applications are coded and configured.
- plan for expansion or upgrades of your existing HSM infrastructure.
- plan for upgrades of electrical capacity and HVAC capacity.

Notes about Monitor/Counter Behavior

When performing certain operations the HSM reaches its maximum performance capability before the counter reaches 100%. This occurs because the counter measures the load on the HSM's CPU and the CPU is able to saturate the asymmetric engines and still have capacity to perform other actions.

Also, symmetric cryptographic operations cause the counter to quickly rise to 90% even though there is significant remaining capacity. This behavior occurs because, as the HSM receives more concurrent symmetric commands, its CPU is able to handle them more efficiently (by performing them in bulk) – thus achieving more throughput from the same number of CPU cycles.

For `lunacm`, see "[hsm monitor](#)" on page 1 in the *LunaSH Reference Guide*.

Performance and the PE1746Enabled Setting

The K6-based HSMs include the SafeXcel 1746 security co-processor, which is used to offload packet processing and crypto computations from the host processor. Use of the SafeXcel 1746 security co-processor can affect performance, and is therefore optional. When enabled, the SafeXcel 1746 security co-processor improves application bulk performance, at the expense of small-packet performance. When disabled, small-packet performance is improved, at the expense of application bulk performance. Data packets less than 1Kb in size are considered small.

You can enable or disable the SafeXcel 1746 security co-processor via the **PE1746Enabled** statement in the **Chrystoki.conf** file (Linux and UNIX) or the **crystoki.ini** file (Windows). The SafeXcel 1746 security co-processor is disabled (0) by default.



Note: K6-based HSMs have a limit of 1000 contexts for SafeXcel 1746 operations, which is a consideration when many client threads are involved, and depends upon the number of concurrent threads.

To enable or disable the SafeXcel 1746 security co-processor

1. Login to your SafeNet HSM client workstation as an administrator.
2. Open the **Chrystoki.conf** (Linux and UNIX) or **crystoki.ini** (Windows) file, as relevant, for editing. The **PE1746Enabled** statement is located in the Misc section of the file, for example:

```
Misc = {
PE1746Enabled = 1;
reconnAtt = 50;
logLen = 262144;
haLog = /usr/safenet/lunaclient/bin/;
}
```

3. Set the value for **PE1746Enabled** as required. Set to 1 to enable. Set to 0 to disable.

Effect on HA

The **PE1746Enabled** setting can affect HA. See ["Performance" on page 110](#) for more information.

Resetting the Internal SafeNet Network HSM PE1746Enabled Setting Following an Upgrade

Because of the effect on some operations, it can happen that a large update to SafeNet Network HSM can fail verification if PE1746Enabled= 0 in the SafeNet Network HSM's internal configuration settings. A patch is available to force PE1746Enabled= 1 on the appliance.

Frequently Asked Questions

This section provides additional information by answering questions that are frequently asked by our customers.

Can I buy a SafeNet Network HSM 1700 and later upgrade it to SafeNet Network HSM 7000?

No. The SafeNet Network HSM 1700 appliance comes with a single power supply, and you can purchase a second PS, remove the blanking plate and install the new PS into the empty slot for the same power-supply redundancy as SafeNet Network HSM 7000, but you cannot increase the SafeNet Network HSM 1700 performance. If you need the performance, buy a SafeNet Network HSM 7000.

Note that you could buy and use the less-expensive SafeNet Network HSM 1700 for your testing, development, or integration, and then purchase SafeNet Network HSM 7000 appliances for your operational deployments where higher performance is desired. Other than performance, the two appliance models are functionally identical. An integration or application that works with one will work with the other, with no adjustment needed.

Note that the main difference (see next question) is performance using asymmetric operations. If your primary cryptographic activity is key-generation of any kind, or involves mostly symmetric operation, then SafeNet Network HSM 1700 should suit your needs at a smaller investment.

Can you highlight the relative performance figures?

Here is a quick summary of some excerpts from the performance testing:

- Key Generations – same performance for both
- Symmetric operations – same performance for both
- Asymmetric operations – check with your SafeNet Sales representative, but here are some samples.
 - RSA 1024: 7000 signings/second vs 1700 signings/second
 - RSA 2048: 1200 vs 350
 - RSA 4096: 160 vs 50
 - ECC P-256: 1000 vs 490

How can I achieve the kinds of performance numbers you quote?

We provide repeatable numbers that you could achieve if you tested in an environment similar to our test-lab environment. This method provides numbers that you can compare against numbers from any of our competitors. In general, that means automated scripting to perform a given operation with a "standard" keysize, or standard input parameters, and ensuring that no other operation or latency is allowed to intrude - an isolated high-speed network with no other activity.

Operation outside a controlled laboratory environment is messy and sometimes unpredictable, with many variables that could affect testing if we did not control the parameters and conditions as tightly as possible. Therefore, all manufacturers' test numbers tend to be the best you can get under ideal conditions.

So, for example, we use the RSA 1024-bit key as the standard for performance of asymmetric sign and verify operations, because that is what the industry uses as a common basis of comparison. That remains true even though the 1024 key size is now generally considered too small for modern operational use, and most applications would tend to use 2048 key sizes (at least that was the case when this was written in 2013).

As well, we bombard the HSM with at least 30 threads simultaneously performing that simple test operation (this is down from a previously required 50 threads due to refinements in SafeNet Network HSM 5.2). Any fewer might fail to exercise the HSM to its fullest. Significantly more threads would not increase the performance numbers, so we work in the "sweet spot" and we suggest that you do as well if performance is your greatest concern.

In operational, real-world situations, your clients are likely to have other responsibilities, or might make other requests of the HSM - for example, a fresh asymmetric key generation before each asymmetric signing operation would slow the sign/verify performance down to key-gen speeds... orders of magnitude slower than simple, repetitive signing that reuses a single key. Network latency and other factors also serve to degrade performance in non-laboratory operation.

Do you have any additional advice on how to interpret performance numbers? We're trying to match against a set of performance requirements that are stated as "signings per millisecond".

Remember that our numbers are not based upon sequential access, but are based upon an optimal number of concurrent threads accessing the HSM. That optimal number differs from one HSM to the next. Best performance for the legacy SafeNet CA4 is achieved from roughly 4 - 8 threads with a slight drop-off after that due to the overhead of context switching. On SafeNet Network HSM 5.0 and 5.1, the optimal number of threads has been in the range of 50, but refinements in SafeNet Network HSM 5.2 have brought the optimal number of concurrent threads down to 30.

Command latency (the time required for any one command to complete) is not a direct inverse of TPS, and can be dependent on several things including the number of threads, the network latency, and the interleaving of different command types to the HSM. (That is, ideal signing performance can be achieved only if the HSM is processing only signing requests.) As the number of threads increases (and thus the corresponding TPS) the latency increases as well. So you might be seeing what appears to be 6 responses per millisecond, but the round-trip latency of any one of those commands might be as high as 500 ms (for example).

This is partially why, when you observe numbers from our tools, the numbers are quite variable for the first several seconds, and then settle around stable values as the testing proceeds.

Performance requirements should state both the total throughput and the maximum latency that a command must execute, and the ideal number of threads should be adjusted accordingly.

We expect to generate millions of keys per year. What is the expected number of read/write operations that your HSM memory can perform before the solid-state memory begins to fail?

You are thinking of the HSM's flash memory, which would be used to store token objects. By their nature, those do not frequently change.

Your "key factory" application (generating keys that are pushed out to external devices like smart cards) should be generating your short-lived keys as session objects, rather than as token objects. Session objects do not use the flash memory at all - they are created and exist in the HSM's RAM only, which can perform virtually unlimited read/write operations.

In the "key factory" scenario, we need to generate approximately 30 ECC P224 keys/second. How many SafeNet Enterprise HSMs will we require?

SafeNet Network HSM 5.2, and later, can do about 35 ECC P224 keys per second. You would then need one appliance for performance reasons, if you could count on steady demand with no peaks. Consider having a backup as well.

In the situation where you might encounter bursts of key generation traffic, prudence suggests one operational SafeNet Network HSM, one in standby to accommodate the burst traffic, and a third unit for backup.

Public Key Infrastructure (PKI) and Removable HSMs

This chapter describes PKI in the context of SafeNet HSMs. It contains the following sections:

- ["PKI with SafeNet Network HSM" below](#)
- ["Using SafeNet USB HSM or Token-format HSM with SafeNet Network HSM Appliance" on page 263](#)
- ["Card Reader \(SafeNet DOCK 2\) and Token-style HSMs" on page 266](#)
- ["Frequently Asked Questions" on page 269](#)

PKI with SafeNet Network HSM

The PKI feature with SafeNet Network HSM is summarized as follows:

- Legacy SafeNet PCM token HSMs can be configured as PKI devices via SafeNet Dock 2 (an external, USB-connected SafeNet card reader).
- SafeNet USB HSMs can be connected to the SafeNet Network HSM USB port and configured as a PKI device.
- Each PKI device can support only one partition.
- One SafeNet Network HSM can support up to six PCM token HSMs (via three SafeNet Dock 2 readers), or three SafeNet USB HSMs, or a mix of both (limited to three USB connections to the appliance).

What to Do

If you are an end-user of SafeNet HSM products, then it is assumed that you are using your SafeNet HSM in conjunction with a third-party application that is HSM-aware. Simply follow the instructions and procedures associated with that application, once you have installed the SafeNet HSM and configured it (described in the Installation Guide and Configuration Guide).

If you are a developer or integrator of applications, then refer to the Software Development Kit Guide, along with the "Extensions to PKCS # 11" (SafeNet's augmentation of the PKCS # 11 standard API), and in particular to the token pki commands in the Reference section of this Help.

Special commands are provided under the token pki menu to perform HSM management operations on the removable HSMs (SafeNet tokens or SafeNet USB HSMs). Briefly, to make use of SafeNet tokens and SafeNet USB HSMs with SafeNet Network HSM, you need to use:

- similar to initializing the onboard SafeNet Network HSM, but prepares the removable token to be used in this context
- make the named token available to the SafeNet Network HSM appliance as an additional PKCS#11 slot (like an additional, removable HSM Partition)
- make the deployed token/slot available/accessible to Clients

- generate or clone to populate the token with the necessary keys, certs, etc.

The is used to make the inserted, deployed token unavailable, such as when preparing to remove it. The remaining commands, under token pki are for general management of the tokens, and are similar to equivalent HSM and Partition commands.

Lunash "token" command set provides 16 commands to administer the external PKI HSM (SafeNet USB HSM). You need just two of those "token" commands, plus one "client" command to make the PKI HSM ready to use, as follows:

1. Pre-deploy the external HSM, to prepare it. Type:

```
[mylunaSA] lunash:>token pki predeploy -l G5Pki -serial 475289
```

```
[mylunaSA] lunash:>token pki predeploy -l G5Pki -serial 475289
```

```
Please type "proceed" to continue, anything else to abort: proceed
```

```
*****
*
* About to factory Reset the HSM
*
*****
*****
*
* About to initialize the HSM
* Please pay attention to the PED
*
*****
```

```
Do you want to use FIPS-approved algorithms and key strengths only (yes or no)? Yes
```

```
*****
*
* About to change the HSM FIPS policy
* Please pay attention to the PED
*
*****
*****
*
* About to create a partition on the HSM
* Please pay attention to the PED
*
*****
*****
*
* About to set the partition policies
* Please pay attention to the PED
*
*****
*****
*
* About to create a partition challenge
* and activate the partition.
* Please pay attention to the PED
* Please write down the PED secret!
*
*****
```

```
Success predeploying the token!!
```

```
Command Result : 0 (Success)
```

```
[mylunaSA] lunash:>
```

- Now deploy the pre-deployed HSM to make that HSM available to the SafeNet Network HSM as another application partition or PKCS#11 slot. Type :

```
[mylunaSA] lunash:>token pki deploy -label G5Pki -serial 475289
```

```
*****
*
* About to activate the token for testing. *
* Please pay attention to the PED *
*
*****
Please enter the current user challenge:
Success deploying token StellaG5Pki with serial num 475289 !
```

```
Command Result : 0 (Success)
```

```
[StellaSA2] lunash:>token pki listDeployed
```

```
Label                      Serial Num
-----
StellaG5Pki                475289
```

```
Command Result : 0 (Success)
```

```
[StellaSA2] lunash:>client assignPartition -partition StellaG5Pki -c StellaLap
```

```
'client assignPartition' successful.
```

```
Command Result : 0 (Success)
```

HA

The SafeNet Network HSM's HA (high availability) feature, when implemented for PCM tokens or SafeNet USB HSMs must be used only across multiple SafeNet Network HSM appliances. NEVER allow multiple SafeNet PCM tokens or SafeNet USB HSMs to be placed in an HA configuration on a single SafeNet Network HSM appliance. This is similar to the requirement to not include two partitions of the same HSM in a single HA group.

Using SafeNet USB HSM or Token-format HSM with SafeNet Network HSM Appliance

Traditionally, Public Key Infrastructure (PKI) with SafeNet HSMs has been implemented using removable token-style (PCMCIA format) HSMs securely connected to a local workstation via a card reader. The portable HSM contained the PKI root certificate, and was inserted, read, updated, etc., as needed, then removed and returned to safe storage. This was a high-security, low-volume/low-speed environment and requirement.

This differed from the transaction-security world where HSMs needed to be network-available in order to perform and accelerate high volumes of secure transactions.

When those two applications began to converge, the SafeNet Network HSM, with its model of large, fast, network-connected HSM providing multiple virtual-HSM (Partition) workspaces, was adapted to support the addition of token-format PKI HSMs (such as SafeNet PCM or SafeNet CA4).

External HSMs (Token-style and G5 style)

You can connect a SafeNet DOCK2 card reader for limited use with SafeNet Backup tokens (legacy G4 PCMCIA removable token-format HSMs). The removable-token backup HSM was used to backup legacy SafeNet Network 4.x HSMs and can be connected to SafeNet Network HSM 5.x or 6.x to restore the legacy key material as part of a one-way migration.

You can connect the more modern SafeNet USB HSM as an externally connected PKI slot, for use in the PKI Bundle option. Some customers use this arrangement to hold a root CA. The following caveats apply:

- The **token backup** commands can see and manage only the backup device, and not PKI devices.
- The **token pki** commands can see and manage only the PKI devices, and not backup devices.
- The PKI device must use PED authentication only, to be deployed.
- The **token pki update** commands update the capability and firmware for PKI devices.
- The process to move keys off G4 token HSMs (SafeNet CA4) is to migrate the keys to a K6 HSM (either the K6 inside SafeNet Network HSM, or the standalone K6 (SafeNet PCIe HSM inside a host computer)) and then to SafeNet USB HSM. Cloning between G4 and G5 devices is not supported.



CAUTION: Migration is not supported to firmware 6.22.0. Migrate first to an HSM at a firmware version older than 6.22.0, and then update the HSM firmware to version 6.22.0 or newer.



CAUTION: Beginning with SafeNet HSM 6, we do not support PKI bundle using removable PCMCIA token HSMs (SafeNet CA4) and the SafeNet DOCK 2 reader. The SafeNet DOCK 2 reader is supported only for migration. If you need the PKI bundle function from removable tokens, do not upgrade.



Note: PPSO is not supported for the PKI-bundle configuration using SafeNet USB HSM. There is no provision to apply PPSO capability via SafeNet Network HSM to the externally connected SafeNet USB HSM. If the SafeNet USB HSM was removed to a host computer and updated to firmware 6.22.0 and had the PPSO capability applied (destructive operation), then returned to the SafeNet Network HSM to resume PKI-bundle operation, the interface has no provision to create a PPSO partition in the external HSM. Rather, a legacy-style partition would be created for PKI-bundle operation.

Constraints

To use an external PKI HSM directly with SafeNet Network HSM 5 requires a SafeNet USB HSM, or a SafeNet DOCK2 reader with SafeNet CA4 token-style HSM at firmware 4.8.7 or later.

Whether you are using the onboard HSM or not, in order to use a SafeNet Network HSM for PKI bundle operations (using Luna/HSM CA4 or Luna/HSM PCM tokens in the appliance's card-reader) you **must** at least **initialize** the

onboard (K6) HSM in order to use the connected HSMs. Any further preparation of the onboard HSM depends on how (or if) you intend to make use of it, but having the main HSM initialized before you attempt operations with PKI HSMs connected to it is a minimum requirement.

PKI and HA

You can combine the PKI bundle configuration (a SafeNet USB HSM, or a SafeNet DOCK2 with inserted SafeNet CA4, connected to your SafeNet Network HSM appliance) with the HA grouping functionality. That is, PKI can be part of HA redundancy and load balancing. However, by design, we do not support the assigning of two or more devices from the same SafeNet Network HSM to one HA group. That is:

- while SafeNet Network HSM supports multiple HSM partitions, you cannot combine two or more partitions from one SafeNet Network HSM into an HA group, and
- while you can attach a SafeNet USB HSM or a SafeNet CA4 token HSM to a SafeNet Network HSM, you cannot combine two (or more) HSMs or partitions, associated with a single SafeNet Network HSM, into a single HA group.

In either case, that sort of arrangement would allow the SafeNet Network HSM to become a potential single-point-of-failure, which defeats HA's redundancy.

Instead, if you have multiple SafeNet USB HSMs or SafeNet CA4 token HSMs that you wish to use in PKI bundling with SafeNet Network HSM, then you should connect each SafeNet USB HSM or SafeNet CA4 HSM to a separate SafeNet Network HSM. You should not attempt to include more than one SafeNet Network HSM partition, or a partition and an externally connected HSM, in a single HA group. The HA logic recognizes HA member slots from different NTLA/NTLS links, only. This is by design.

Slot Enumeration

The client-side utility command "vtl listslot" shows all detected slots, including HSM partitions on the primary HSM, partitions on connected external HSMs, and HA virtual slots. Here is an example:

```
bash-3.2# ./vtl listslot
Number of slots: 11
The following slots were found:
```

Slot #	Description	Label	Serial #	Status
slot #1	LunaNet Slot	-	-	Not present
slot #2	LunaNet Slot	sa76_p1	150518006	Present
slot #3	LunaNet Slot	sa77_p1	150475010	Present
slot #4	LunaNet Slot	G5179	700179008	Present
slot #5	LunaNet Slot	pki1	700180008	Present
slot #6	LunaNet Slot	CA4223	300223001	Present
slot #7	LunaNet Slot	CA4129	300129001	Present
slot #8	HA Virtual Card Slot	-	-	Not present
slot #9	HA Virtual Card Slot	-	-	Not present
slot #10	HA Virtual Card Slot	ha3	343610292	Present
slot #11	HA Virtual Card Slot	G5_HA	1700179008	Present



Note: The deploy/undeploy of a PKI device increments/decrements the SafeNet Network HSM client slot enumeration list (slots appear or disappear from the list, and the slot numbers adjust for the change). When the PKI slot is temporarily not available (e.g., due to NTLS stop, unplugging of LAN/USB cable, power off, etc.), the slot list does not shift.



Note: If you attempt to perform actions (such as deployment) that require PED operations, against a token/HSM, while other applications are accessing either the onboard HSM or another token in your appliance, then the PED-requiring operations might be noticeably slow. In general, try to reserve such maintenance operations for times when clients are not accessing the HSM or other token. The possible slowness is merely inconvenient and does no harm.

See also "[Card Reader \(SafeNet DOCK 2\) and Token-style HSMs](#)" below.

Contact SafeNet Technical Support -- e-mail: support@safenet-inc.com or phone 800-545-6608 (+1 410-931-7520 International) for the relevant Key Migration document, which includes explicit instructions to migrate your cryptographic objects between different types of SafeNet HSM (generally from legacy models to current models of HSM).

Card Reader (SafeNet DOCK 2) and Token-style HSMs

The card reader sold for use with SafeNet products (PKI) is the SafeNet DOCK 2.



Uses with SafeNet Network HSM 6 are:

- for migration from earlier backups or PKI tokens
- for current (limited) use of legacy PKI tokens (SafeNet CA4) with SafeNet Network HSM.

External HSMs (Token-style and G5 style)

You can connect a SafeNet DOCK2 card reader for limited use with SafeNet Backup tokens (legacy G4 PCMCIA removable token-format HSMs). The removable-token backup HSM was used to backup legacy SafeNet Network 4.x HSMs and can be connected to SafeNet Network HSM 5.x or 6.x to restore the legacy key material as part of a one-way migration.

You can connect the more modern SafeNet USB HSM as an externally connected PKI slot, for use in the PKI Bundle option. Some customers use this arrangement to hold a root CA. The following caveats apply:

- The **token backup** commands can see and manage only the backup device, and not PKI devices.
- The **token pki** commands can see and manage only the PKI devices, and not backup devices.
- The PKI device must use PED authentication only, to be deployed.
- The **token pki update** commands update the capability and firmware for PKI devices.
- The process to move keys off G4 token HSMs (SafeNet CA4) is to migrate the keys to a K6 HSM (either the K6 inside SafeNet Network HSM, or the standalone K6 (SafeNet PCIe HSM inside a host computer)) and then to

SafeNet USB HSM. Cloning between G4 and G5 devices is not supported.



CAUTION: Migration is not supported to firmware 6.22.0. Migrate first to an HSM at a firmware version older than 6.22.0, and then update the HSM firmware to version 6.22.0 or newer.



CAUTION: Beginning with SafeNet HSM 6, we do not support PKI bundle using removable PCMCIA token HSMs (SafeNet CA4) and the SafeNet DOCK 2 reader. The SafeNet DOCK 2 reader is supported only for migration. If you need the PKI bundle function from removable tokens, do not upgrade.

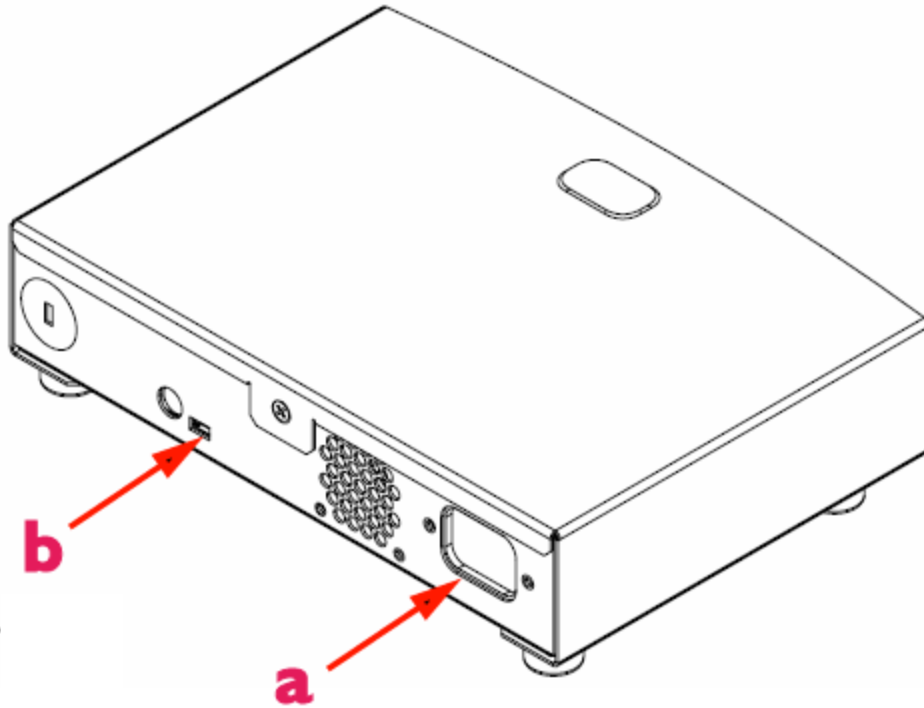


Note: PPSO is not supported for the PKI-bundle configuration using SafeNet USB HSM. There is no provision to apply PPSO capability via SafeNet Network HSM to the externally connected SafeNet USB HSM. If the SafeNet USB HSM was removed to a host computer and updated to firmware 6.22.0 and had the PPSO capability applied (destructive operation), then returned to the SafeNet Network HSM to resume PKI-bundle operation, the interface has no provision to create a PPSO partition in the external HSM. Rather, a legacy-style partition would be created for PKI-bundle operation.

Do not install SafeNet client software on the same system as legacy SafeNet CA³, SafeNet CA4, SafeNet PCM, or SafeNet PCI software. The software is intended for modern/current SafeNet HSMs, SafeNet Network HSM, SafeNet PCIe HSM, SafeNet USB HSM, SafeNet (Remote) Backup HSM.

Connect the SafeNet DOCK2 card reader:

- a) to the AC main power, and
- b) via supplied USB cable to the USB port of your SafeNet Network HSM 5.x.



If power is disconnected for any reason, you might need to restart your application.

The SafeNet PKI Bundle feature supports PED-authenticated PKI HSMs only (SafeNet CA4 for legacy, and SafeNet USB HSM for modern). Use of password-authenticated PKI tokens is not supported. There is no "pass-through" of PED data and commands from SafeNet Network HSM, so your SafeNet DOCK2 (or SafeNet USB HSM) must have its own SafeNet PED connected directly.



Your SafeNet Network HSM needs its own SafeNet PED.

SafeNet Network HSM can be served by a locally-connected PED, if the administrator is located near the appliance, or SafeNet Network HSM can be served by Remote PED, but SafeNet DOCK2 and any inserted token HSMs require a PED to be connected directly and locally to the reader - use of Remote PED to serve an external HSM (such as SafeNet USB HSM, SafeNet Backup HSM, or SafeNet CA4) connected to SafeNet Network HSM is not supported.

See also [PKI - Using an external HSM with SafeNet Network HSM Appliance](#).

Contact SafeNet Technical Support -- e-mail: support@safenet-inc.com or phone 800-545-6608 (+1 410-931-7520 International) for the relevant Key Migration document, which includes explicit instructions to migrate your cryptographic objects between different types of SafeNet HSM (generally from legacy models to current models of HSM).

Frequently Asked Questions

This section provides additional information by answering questions that are frequently asked by our customers.

We operate a Managed PKI and must satisfy our auditors that the root and intermediate keys and certs are protected according to an accepted standard, including when cloned/backed-up.

We have documented procedures for cloning or backup/restore, and for migration from legacy HSMs to current HSMs, but the procedures are only to ensure that the operations complete successfully. Security of private keys is enforced by the HSM(s) and does not rely on procedure.

The encryption key is either 3-key TDES or AES 256, depending on the HSM firmware version, which itself is afforded the same high level of protection as a CA signing private key. The encryption key is derived using the data from the Red PED Key (48 bytes of HSM-generated random data) along with source and target HSM random nonces that are exchanged using RSA 2048 bit encryption. Both the source and target HSMs must be legitimate SafeNet HSMs and their RSA certificates (used to exchange encrypted nonces) are signed by the SafeNet manufacturing PKI when the devices are manufactured.

This chapter describes how to use the remote PED to authenticate to an PED-authenticated HSM at a remote location. It includes the following sections:

- ["About Remote PED" below](#)
- ["Remote PED Architecture" on page 306](#)
- ["Remote PED and pedclient and pedserver " on page 279](#)
- ["Configuring Remote PED" on page 280](#)
- ["Using the Remote PED Feature" on page 289](#)
- ["Using Client-initiated Remote PED Connection" on page 294](#)
- ["Using Server-initiated \(Peer-to-Peer\) Remote PED Connection" on page 298 \(Peer-to-Peer\)](#)
- ["pedServer Configuration File" on page 302](#)
- ["Troubleshooting Remote PED" on page 303](#)

About Remote PED

The Remote PED (SafeNet PED with Remote Capability) allows you to administer HSMs that are housed away from their owners/administrators, at physically remote sites or inside heavily-secured premises, where obtaining local physical access to the HSM is difficult or time-consuming. Remote PED provides administrative convenience similar to remotely accessing a Password-authenticated HSM, but with the added security and role separation of PED authentication.

What do you need?

The feature requires:

- a Remote PED Server instance
 - on a workstation
 - that connects over a network link
 - to a Remote PED Client
 - in the computer or appliance that hosts the HSM
- a SafeNet PED 2.4.0-3 or greater, **with** the Remote PED feature installed, (which has the capability to operate in Local PED or Remote PED mode, as needed).



Note: Not every PED 2.4.0 (or greater) includes the Remote PED feature. That PED capability must be ordered specifically and factory installed.

- an orange, Remote PED, PED Key that provides the authentication for the Remote PED connection between

- the workstation computer (with PED connected and PEDServer running) and
- the remotely located SafeNet HSM with the PEDclient running on the HSM's host.

Term	Meaning
Remote PED	A SafeNet PED, with Remote capability, connected, powered on, and set to Remote mode.
RPV	Remote PED Vector - a randomly generated, encrypted value used to authenticate between a Remote PED (via PedServer) and a distant SafeNet HSM (PEDclient).
RPK	Remote PED Key - an orange PED Key, the portable repository of an RPV value, for use in the Remote PED process.
PedServer	The PED server program that resides on a workstation and mediates between a locally-connected Remote PED and a distant PEDclient (running at a distant SafeNet HSM).
PEDClient	The PED Client program. For a SafeNet Network HSM appliance, PEDclient is embedded. For SafeNet PCIe HSM, SafeNet USB HSM, or SafeNet Backup HSM, PEDclient must be installed on the HSM's host computer. The PED client anchors the HSM end of the Remote PED service and initiates the contact with a PedServer instance, on behalf of its HSM.

Why do I want it?

You want to locate your operational HSM hosts at remote locations or multiple locations around the city, country, world, and be able to administer them fully, from a different location, without need for site visits and without needing to carry PED Keys through unsecured areas.

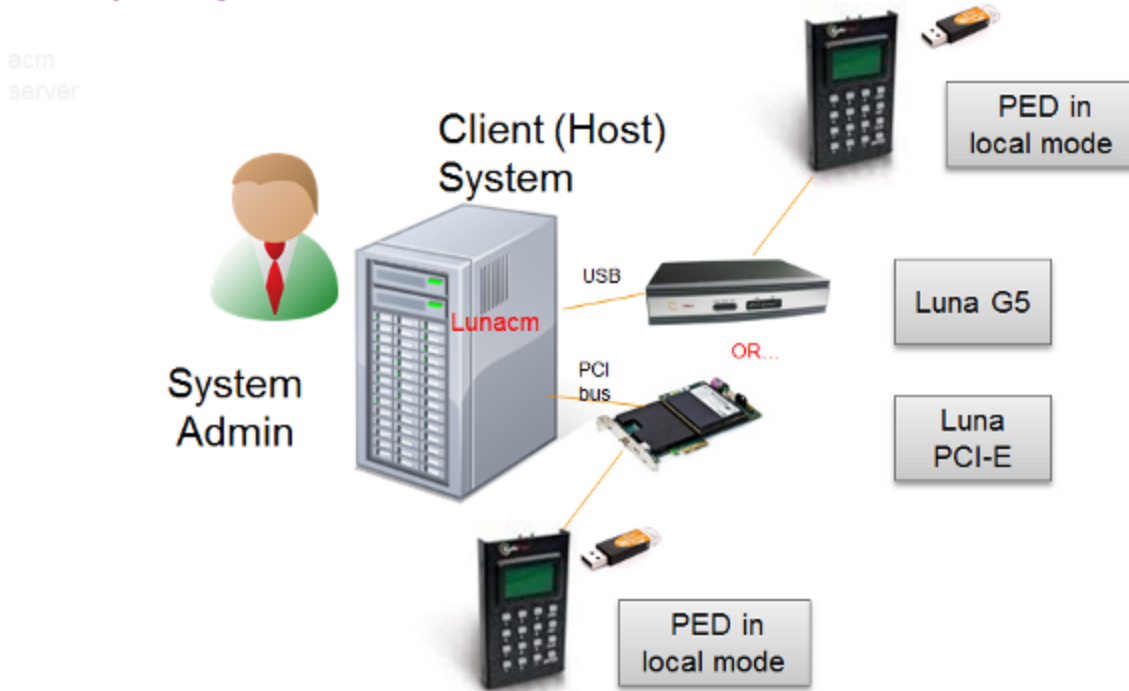
How does it work?

The HSM must initially be configured with a local PED, in order to set its authentication and create a relationship between the HSM and an orange PED Key (RPV, or Remote PED Vector). That RPV, carried via the orange PED Key, is the means by which a PED at a remote (PedServer) location can be recognized and trusted over a distance, by an HSM that shares the same RPV. During the imprinting process, the HSM can take on the RPV of an existing orange PED Key (RPK, or Remote PED Key), or the HSM can generate a new RPV and imprint it on an orange PED Key. This allows you to choose whether

- the current HSM is one of possibly several HSMs that can be remotely administered with the PED data path secured by a single orange PED Key, or
- the current HSM will be the only one secured with that particular Remote PED authentication (it could remain alone, or could be the start of a new authentication group for Remote PED activity - your choice)

The following diagram shows the preliminary imprinting step, where the HSM and (at least one) orange PED Key are made to share an RPV. Again, this must take place via a locally connected PED. The administrator could be co-located with the HSM, or could be elsewhere issuing the commands, but either the administrator or an assistant must be present at the HSM to present the orange PED Key for the RPV imprinting. Once that is completed, further PED operations can be untethered from direct local PED connection and moved anywhere along with that RPV-bearing orange PED Key.

Preparing for Remote PED



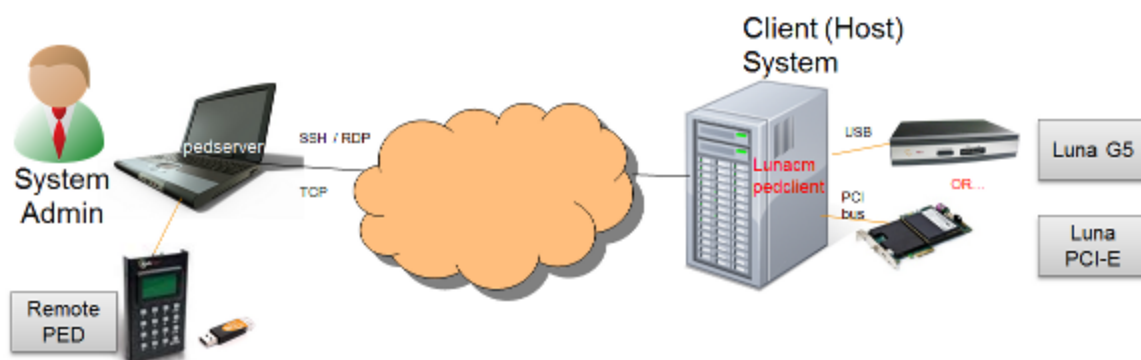
The HSM is then shipped and installed at its remote location.

At your administrative location, a workstation is configured with special (PedServer) software, and a SafeNet PED 2 Remote (remote-capable PED) is connected via USB to that workstation.

Using SSH, you open an administrative session (connect and log in as "admin") on the remote HSM. You tell the HSM to expect a remote PED, rather than local PED. You issue commands as needed.

When an HSM command requires authentication to the HSM, the HSM looks for a remote PED server with the same Remote PED Vector. If it can authenticate properly with that remote PED server, the HSM accepts authentication data via that connection.

Using Remote PED



One-to-One Remote PED Connections

A SafeNet HSM running the PEDclient can establish a Remote PED connection with any workstation that meets the following criteria:

- is running PEDserver.exe
- has a suitable Remote PED connected
- has the correct PED Keys (including the orange key) for that HSM.

The SafeNet HSM can make only a single connection for Remote PED operation at one time. The current session must timeout or be deliberately stopped before another workstation can be called into a Remote PED connection with that SafeNet HSM.

Similarly, a given workstation can enter into a Remote PED connection with any SafeNet HSM with PEDClient, or any SafeNet HSM, that initiates such a connection (provided the proper PED, PED Keys, software, etc. are all in place), but it can make only one such connection at a time. This contrasts with SSH connections, where that same workstation could have multiple SSH windows open to multiple admin sessions on a single or multiple SafeNet HSMs.

There is no requirement for the workstation providing the Remote PED connection to be the same one that provides SSH administrative access to the HSM, nor is there any requirement that they be different workstations. You can overlap those functions or keep them separate, for your convenience.

Legacy vs Peer-to-peer Remote PED Connections

The above descriptions apply to Remote PED operations in general. The PEDClient runs on the computer that hosts the SafeNet HSM, while the PEDServer runs on the computer that hosts the SafeNet PED. Historically, this meant that PEDClient was launched and tried to open a connection to a specified instance of PEDServer. However, in some cases the network and firewall rules that surround the HSM host forbid that host initiating the connection from inside the firewall. For those situations, PEDServer now supports the ability to initiate the Remote PED connection from the PEDServer side, over a link secured by means of the HSM host's (usually SafeNet Network Appliance) server certificate. In aid of that peer-to-peer connection, Remote PED also makes use of a configuration file separate from the LunaClient crystoki.ini file.

Modes

Peer-to-peer connection is supported by **PedServer.exe -mode connect** and **-mode disconnect** commands. The rules governing the connections are as follows:

- The default mode when PEDServer starts is legacy mode (where PEDServer waits for a connection to be initiated from an instance of PEDClient).
- When you type the **-mode connect** command for PEDServer (requesting the start of peer-to-peer mode), that command requires the registered HSM appliance name, which is stored in the PedServer configuration file, where each appliance name is associated with an IP address. That information, along with the appliance certificate is used to create the connection to an instance of PEDClient on the HSM host computer (usually SafeNet Network HSM).
- The **-mode connect** command detects if legacy mode is running.
 - If legacy mode *is not* running, the **-mode connect** command initiates a connection to the indicated HSM host (normally SafeNet Network HSM appliance).
 - If legacy mode *is* running, then **PEDServer -mode connect** checks if a legacy-mode connection currently

exists. If such a connection currently exists, then the **-mode connect** presents an error message informing you to terminate that connection before retrying the requested peer-to-peer connection. That is, a **-mode connect** request for peer connection does not override an existing legacy connection; it just tells you about it and lets you make the decision.

- If legacy mode is running and **PEDServer -mode connect** does NOT discover an existing legacy connection in effect, then the legacy mode is shut down and the peer-to-peer connection is initiated.
- The **-mode disconnect** command terminates an existing peer connection and returns the PEDServer to legacy mode. If the PEDServer is already in legacy mode, the **-mode disconnect** command is ignored.

Connections and security - Client-initiated vs Server-initiated

Direction	Connection and Security
Client-to-Server (original) connection	TCP/IP only; encryption by orange PED Key secret (RPV)
Server-initiated (a.k.a. Peer-to-peer)	TLS with certificate exchange to create the secure tunnel and then also encrypted by the orange PED Key (RPV)

Note:

Client-to-server - the server is on the Remote PED host, and the connection is initiated by the Network HSM (or other HSM host).



Server-initiated - the server is still on the Remote PED host, but it now initiates the connection to a Network HSM (or other HSM host) where that HSM host might be behind a firewall that forbids outbound connection requests).

Constraints

The following constraints apply to Remote PED connections:

- A maximum of twenty connections is supported on the PedClient.
- A maximum of 80 Network HSM appliances can be registered in PedServer.
- If the connection is terminated abnormally (for example, a router switch died), there is no auto-connection. The PedServer automatically restarts and runs in legacy mode.
- When running in server-initiated connection mode, the PedServer does not engage the listening service, for security reasons and to simplify usability.
- Once the PedServer connection to the PedClient is established, the connection remains up until the
 - the **-mode disconnect** command is executed from the PedServer, or
 - PedClient terminates the connection.

Certificate exchange and port

The exchange of certificates for server-initiated Remote PED is similar to the exchange and registration of certificates for NTLS, and enables a TLS 1.2 link. It is implemented separately because a Remote PED host computer might also be a client for SafeNet Network HSM appliances, or it might not, and the secure peer-to-peer Remote PED connection must be able to work in either circumstance.

On a SafeNet Network HSM appliance at version 6.2.1 or newer, to support peer-mode (server-initiated) Remote PED, port number 9697 is opened for public TCP access. This is not considered a security hazard, since only incoming connections with a certificate are accepted.

Security

In either mode, the major security for your HSM contents is the physical security that you maintain around the PED Keys.

- The Remote PED link can occur only if the person operating the PED can present an orange PED Key that contains an RPV that exactly matches the RPV on the HSM.
- The other PED Keys that are served over the Remote PED link must be the correct keys for each role/function on that HSM (SO, Crypto Officer or Crypto User, Cloning Domain, Auditor, SRK, respectively).

Configuration file

From release 6.2.1, onward, configuration file `pedServer.ini` supports certificate management and other aspects of peer-to-peer Remote PED. The `pedServer.ini` configuration file is used to change the timeouts values and to store a list of appliance certificates, to support SSL TLS v1.2 connections between the PedServer and a SafeNet Network HSM appliance.

The following entries are available :

```
[RemotePed]
PongTimeout = 5
PingInterval = 1
LogFileTrace = 0
LogFileError = 1
LogFileWarning = 1
LogFileInfo = 1 ;
MaxLogFileSize = 4194304
LogFileName = .\remotePedServerLog.log
BGProcessShutdownTimeoutSeconds = 25
BGProcessStartupTimeoutSeconds = 10
InternalShutdownTimeoutSeconds = 10
SocketWriteTimeoutSeconds = 50
SocketReadRspTimeoutSeconds = 180
SocketReadTimeoutSeconds = 100
ExternalServerIF = 1
ServerPortValue = 1503
ExternalAdminIF = 0
AdminPort = 1502
IdleConnectionTimeoutSeconds = 1800
RpkSerialNumberQueryTimeout = 15

[Appliances]
SSLConfigFile = \usr\safenet\lunaclient\bin\openssl.cnf
ServerCAFile = \root\CAFile.pem
```

```

ServerIP00 = 192.20.11.86
ServerPort00 = 9696
ServerName00 = mySafeNetAppliance
CommonCertName00 = test1
ServerName01 = myotherSafeNetAppliance
ServerIP01 = 192.20.9.46
ServerPort01 = 9697
CommonCertName01 = test2

```

An entry is added in the chrystoki configuration file to point to the location of the PedServer configuration file.

```

[Ped Server]
PedConfigFile = \usr\safenet\lunaclient\data\ped\config

```

Priority and Lockout

When a Remote PED connection is in force, the local PED interface to the HSM is disabled. If a local PED operation is in progress, it is not possible to start a Remote PED connection until the current local-PED-mediated HSM operation completes. But it must be an active operation sequence - merely having a local PED physically connected to the HSM does not lock out the initiation of a Remote PED connection.

For example, if you had started an HSM command that began using a connected local PED and PED Key for authentication, and you started an SSH session in which you issued the **ped connect** (LunaCM) command or **hsm ped connect** (LunaSH) command, one of the following two things would happen:

- the remote "PED connect" command would begin executing, but would pause while the local-PED operation (started in the other command session) was in progress, and resume when the local-PED operation terminated

OR

- the remote "PED connect" command would begin executing, but would pause while the local-PED operation was in progress, and eventually time-out if the local-PED operation did not terminate sufficiently quickly.

If a Remote PED connection is currently in force, then the local PED is ignored, and all PED requests are routed to the Remote PED.

If a Remote PED connection is currently in force, then subsequent attempts to start a different connection are refused until the current connection times out or is deliberately stopped.

Remote PED Timeout

In local PED mode, one SafeNet PED is connected directly to the HSM. Timeouts are governed by the configuration of the appliance or host computer and the HSM and are not generally modifiable.

In Remote PED mode, the PED Server on each remote Workstation has a timeout setting (which can be modified), and the HSM has a Remote PED timeout setting that can be seen and modified in the configuration file. If nothing has been set, then the default value for the Remote PED connection timeout (1800 seconds) is in effect.

Priority

The Remote PED server instances on workstations, and the Remote PED client inside the SafeNet Network HSM appliance or on an HSM host computer, are not aware of each other's timeout values. For a given Remote PED connection, the shorter timeout value rules.

Thus, if a Remote PED server on one of your workstation computers were to timeout during a Remote PED sequence, it would log the event and send a message to the appliance or the HSM host that the connection had been open too long. The Remote PED Client on the SafeNet Network HSM appliance or on an HSM host computer, receiving that message, would gracefully close the link and the host-side timeout would not be reached.

Generally, the **state** that causes the HSM to look for PED authentication via the Remote path, rather than from a locally-connected PED, persists unless you change it. The session between the Remote PED and the PedServer on that host also remains intact. It is the **link** between PedClient (at the HSM end) and PedServer (at the Remote PED end) that goes down for lack of use, and that link can be restarted with a single command when needed.

If it has been some time (more than half an hour) since you performed any authentication operations via Remote PED, the link has probably lapsed. Find out with `lunacm:>ped show`. If it says "not assigned", then the connection has been lost. Simply issue the **ped connect** command again, when needed.

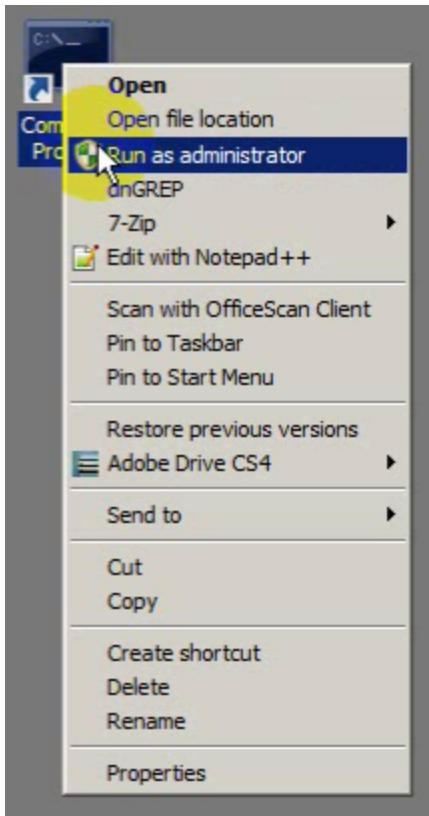
Ports

We suggest port 1503 for the legacy Remote PED connection, but you can use any port that does not conflict with another operation.

Windows 7

PedServer.exe (on the computer to which your Remote PED is attached) is run from the command line. If you accepted default locations when installing LunaClient (or Remote PED option from within LunaClient installer), then the software is installed on your C: drive under "Program Files". This has implications regarding the permissions you have on the system.

To use PedServer from within a protected location on a Windows 7 computer, right-click the Command Prompt icon, and from the resulting menu select "Run as Administrator".



Note: If you lack system permissions to operate as Administrator on the computer that is to host the PED Server, contact your IT department to address the situation.

Alternatively, install LunaClient (or a subset of it, such as Remote PED) in a non-default location in your computer that is not subject to permission restrictions. If you do so, you avoid this permission problem, but you must translate all file-location references in this documentation to reflect the chosen install location.

If you open a command-prompt window as an ordinary user in Windows 7, and run PedServer.exe, the program detects if it lacks access and permissions, and returns an error like the following:

```
C:\Program Files\SafeNet\LunaClient>pedserver mode start
Ped Server Version 1.0.5 (10005)

Failed to load configuration file. Using default settings.

Ped Server launched in startup mode.
Starting background process
InternalRead: 10 seconds timeout
Failed to recv query response command: RC_OPERATION_TIMED_OUT c0000303
Background process startup timed out after 10 seconds.
Startup failed. : 0xc0000303 RC_OPERATION_TIMED_OUT

C:\Program Files\SafeNet\LunaClient>
```

If you encounter the error above, use Windows Task Manager to select the PedServer process, right-click, and select "End process", before cleanly retrying **PedServer.exe** via an Administrator Command Prompt.

Other Windows versions have not exhibited this requirement.

Limitations

Regardless of whether you use Client-initiated mode or server-initiated, the connection is one-on-one. While a Remote PED connection is active between one HSM and one remote PED workstation (running PedServer.exe), neither entity is able to make a similar connection with a different partner. The connection must time out, or be deliberately stopped before the HSM can connect with another PedServer workstation and enter a new remote PED authentication arrangement.

Who can connect a Remote PED with your HSM

When an RPV is created, it is a randomly-generated value that exists nowhere else. You control which (and how many) HSMs will contain that RPV, and which (and how many) orange RPK PED Keys will contain copies of it.

A Remote PED with an inserted RPK (orange Remote PED Key) can be used only with distant SafeNet HSMs that share that exact RPV. If you launch a Remote PedServer with a connected Remote PED and provide any other orange PED Key, it is not accepted by any distant SafeNet HSM that does not have the matching RPV. In this manner, you can segregate the ability of personnel to remotely control specific HSMs, by controlling which orange PED Keys they are issued. Two people in the same office could have access and control of entirely different sets of remotely located HSMs, with no overlap, as long as you trusted them not to exchange orange PED Keys. You can further control who has what access by invoking MofN when you first create an RPV.

Compatibility

Remote PED for SafeNet HSM 5.2 and newer is not compatible with earlier HSM versions.

Security of Remote PED

All communication between the Remote PED and the HSM in its host is transmitted within an AES-256 encrypted channel using session keys based on secrets (the Remote PED Vector (RPV) on the orange Remote PED Key (RPK)) that are shared out-of-band via the Remote PED role. This is considered a very secure query/response mechanism.

Remote PED and pedclient and pedserver

When it is not convenient to be physically near the host computer that contains a SafeNet HSM, you can remotely and securely connect a SafeNet PED and present PED Keys, as follows:

1. On the computer used to host the HSM or SO Partition, allow remote desktop access or ssh, and have the `pedclient.exe` program available.
2. On the remote administrative workstation used to host the remote PED, (which for this purpose must run the Windows operating system) use remote-desktop client or use ssh, have a SafeNet PED2 (with Remote capability) connected, and have the `pedserver` tool installed and running.
3. Using remote desktop or ssh, make the Remote PED connection between the HSM host and the remote administrative workstation:
 - a. Start the `pedserver` listening on the remote PED host.

- b. Start `pedclient` on the HSM host indicating the slot number of the HSM for which Remote PED services are to be provided.

The combination of `pedserver` on one computer and `pedclient` on the other provides the trusted path for secure transfer of authentication data.

4. Run commands on the HSM via the remote desktop or `ssh`.

Use **static IP addressing** for PED Client / PED Server. PED Client can fail to find a server if a dynamic address is indicated. An example error might look like this:

```
lunash:>hsm ped connect -ip 192.20.11.67 -port 1503
Luna PED operation required to connect to Remote PED - use orange PED Key(s).
Ped Client Version 1.0.5 (10005)
Ped Client launched in startup mode.
readIPFromConfigFile() : config file did not contain an IP address.
Startup failed. : 0xc0000404 RC_FILE_ERROR
Command Result : 65535 (Luna Shell execution)
lunash:>
```

Security of Remote PED

The authentication conversation is between the HSM and the PED. Authentication data retrieved from the PED Keys never exists unencrypted outside of the PED or the HSM.

PEDClient and PEDServer merely provide the communication pathway between the PED and the HSM. Along that path, the authentication data remains encrypted.

Multiple HSMs and Remote PED

Remote PED (via `pedclient.exe`) can provide PED services to only one HSM slot at a time. To provide PED interaction (remotely) to another slot, you must close `pedclient.exe` for that first slot/HSM and then open `pedclient.exe` for the next slot/HSM.

Once a slot has been set up with its authentication data cached (autoActivation), and `pedclient` has closed (perhaps because you need to open `pedclient` for another slot), you must not issue any command to that original slot that would require PED interaction. If you issue a command that invokes a PED operation, when no PED is connected to the HSM (such as when `pedclient` and the Remote PED are busy with another slot, or when `pedclient.exe` is simply not running), the affected HSM pauses until the requested operation times out. This means that any client application that was using that HSM stops for the duration of the timeout.

Configuring Remote PED

The PED is an accessory device that allows compatible SafeNet HSMs to securely store their authentication data on PED Keys (specially configured USB tokens), to retrieve that data when needed, and to modify the content of PED Keys for security and operational purposes. All of the SafeNet PED and PED Key actions can be accomplished with the SafeNet PED directly connected to the SafeNet HSM, and powered by that HSM. Sometimes that direct connection is inconvenient, due to location of the HSM and of the personnel who are charged with controlling and managing the HSM. In such circumstances, it can be useful to employ a SafeNet PED with Remote capability.

Remote PED is supported (and requires installation/configuration) in two parts:

- **PEDClient**, which runs on the HSM host and allows the HSM to seek PED Key data from a remotely located SafeNet PED. PEDClient is part of the SafeNet HSM Client software installation for every type of SafeNet HSM except SafeNet Network HSM (because PEDClient is already present, by default, within the SafeNet Network

HSM appliance).

- **PEDServer**, which runs on Remote PED host. PEDServer is installed if the "Remote PED" option is selected during SafeNet Client software installation, and includes the PedServer.exe executable, along with the SafeNet PED device drivers. If the target computer is intended to be a PEDServer, but is not going to be a Client to your SafeNet HSM, then you can use SafeNet HSM Client installer to install only the Remote PED option and exclude any unneeded client-side options.

You will need:

- An HSM host, configured as described elsewhere in this document, with PEDClient available, and with its own working network connection.
- A remote PED host computer with a supported operating system (see the Customer Release Notes for supported platforms) to run PEDServer.
- Sufficient privileges on the remote PED host, depending on platform and location (local network, WAN, VPN...)
- Current SafeNet HSM Client installer (LunaClient.msi)
- SafeNet PED (Remote capable) V.2.4.0-3 or newer (see the bottom of the PED's Select Mode menu for the version)
- The power block and cord that accompanied your Remote PED, and the USB-A to USB-Mini-b cable
- PED Keys.
- A network connection.

Configuring the PEDClient and PEDServer

This configuration takes place in two locations:

- on the HSM host.
- on the Remote PED host.

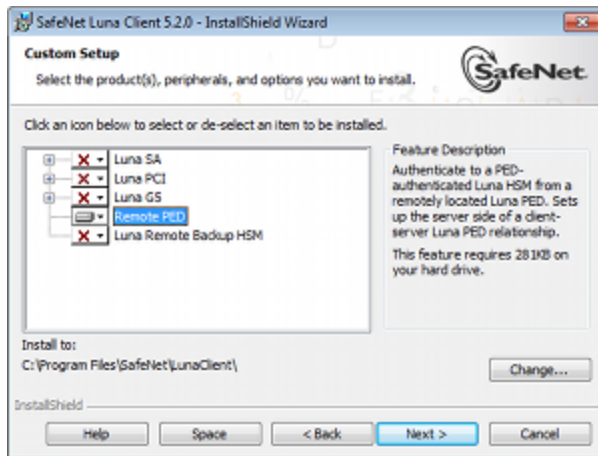
To configure the HSM host computer

1. Install/configure your HSM host as described previously.
2. Change to the directory where SafeNet HSM Client is installed and launch lunacm.
Type: `c:\Program Files\SafeNet\LunaClient> lunacm`
3. With a SafeNet PED connected locally, initialize a Remote PED Vector for the HSM and for an orange PED Key.
Type: `lunacm:> ped vector init` and respond to the SafeNet PED prompts.
By means of your responses to the PED prompts, you can choose to have the HSM generate a new RPV to be held by both the HSM and a new orange PED Key, or you can choose to re-use an RPV already on an existing orange PED Key, and imprint that on the HSM.
As always, we suggest that you make at least one extra copy of the Remote PED Key.
4. Bring an orange PED Key, containing the RPV for this HSM, from the HSM to the location of the Remote PED server.

To configure the Remote PED host computer

1. SafeNet PED should not yet be connected to the PEDServer computer.

- Install the SafeNet HSM Client software, selecting "Remote PED" option - for the purposes of Remote PED. Any additional SafeNet HSM Client installation choices are optional for this host system.

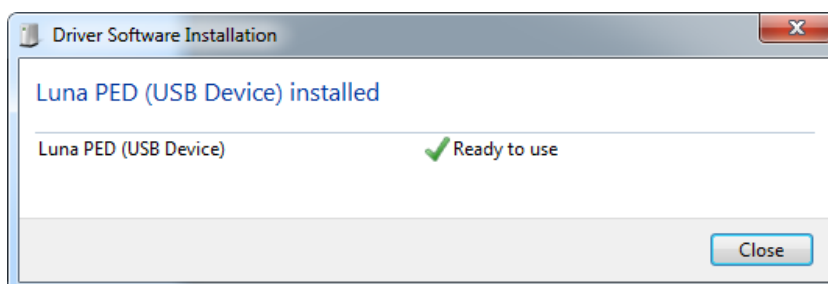


- Click **Install** when prompted to install the driver.



- Reboot the computer to ensure that the LunaPED driver is accepted by the operating system. This is not required for Windows Server Series.
- Connect the Remote Capable SafeNet PED to AC power, using the supplied power block, and to the PEDServer computer, using the supplied USB-A to USB-mini-b cable.

Windows acknowledges the new device.



SafeNet PED performs its start-up sequence, and settles into Local Mode, by default.

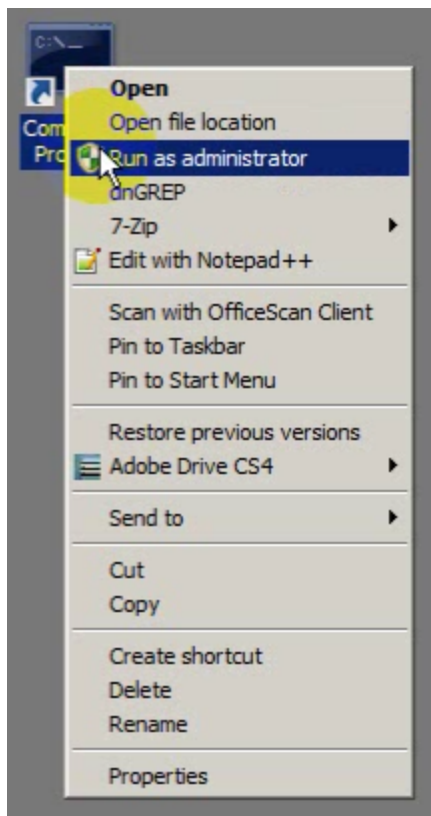
- Press the [<] key on the PED to access the "Select Mode" menu.
- Press [7] to select "Remote PED" mode.

8. Ensure that your firewall does not block communication between PEDClient and PEDServer. If switching off the firewall for Home and Public Network is not an option, see the Troubleshooting section below.
9. Open a Command Prompt window.

If PedServer.exe attempts to access the pedServer.ini file in C:\Program Files\... that is treated as an action in a restricted area in some versions of Windows. In that case, you should open the Command Prompt as Administrator, rather than as your normal user. To do so, right-click the Command Prompt icon and, from the pop-up menu, select **Run as administrator**.



Note: Windows Server 2008 launches Command Prompt as Administrator, by default, so no special steps are necessary.



Note: By default, PedServer.exe attempts to access pedServer.ini if such a file exists in the expected location. If it does not exist, then default values are used by PedServer.exe until you perform a "-mode config -set" operation to create a pedServer.ini.

10. Go to the installed SafeNet HSM Client directory.
Type `cd "\Program Files\SafeNet\LunaClient"`
11. Launch the PEDServer.
Type `pedserver -mode start`
12. Verify that the service has started.

Type `pedserver -mode show` and look for mention of the default port "1503" (or other, if you specified a different listening port). In addition, "Ped2 Connection Status:" should say "Connected". This indicates that the SafeNet PED that you connected (above) was found by PEDServer.



Note: If a port other than the default 1503 was specified in `pedserver -mode start`, for example `pedserver -mode start -port 1523`, then `pedserver -mode show` command should pass in the same port, for example `pedserver -mode show -port 1523`.

If a non-default value for the listening port was configured (meaning that it was present in `pedServer.ini`), then `pedserver -mode show` finds the port from that file.

- Note the IP address of the PEDServer host. We generally recommend using static IP, but if you are operating over a VPN, you will likely need to ascertain the current address each time you [re-]connect to the VPN server and are assigned an address.

```
C:\windows\system32>ipconfig
```

```
Windows IP Configuration
```

```
Ethernet adapter Bluetooth Network Connection 2:
```

```
Media State . . . . . : Media disconnected
Connection-specific DNS Suffix . :
```

```
Wireless LAN adapter Wireless Network Connection 4:
```

```
Media State . . . . . : Media disconnected
Connection-specific DNS Suffix . :
```

```
Wireless LAN adapter Wireless Network Connection 3:
```

```
Connection-specific DNS Suffix . :
Link-local IPv6 Address . . . . . : fe80::cd74:173c:692a:22b0%26
IPv4 Address. . . . . : 192.168.0.16
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . : 192.168.0.1
```

```
Ethernet adapter Local Area Connection 3:
```

```
Media State . . . . . : Media disconnected
Connection-specific DNS Suffix . :
```

```
Ethernet adapter Local Area Connection 2:
```

```
Connection-specific DNS Suffix . :
Link-local IPv6 Address . . . . . : fe80::5456:b034:a1ff:96fe%14
IPv4 Address. . . . . : 182.16.153.114 <<--- this one, in our example
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . :
```

```
Tunnel adapter isatap.{9EE24CB0-63D2-4D40-902B-3DC3193701FA}:
```

```
Media State . . . . . : Media disconnected
```

```

Connection-specific DNS Suffix . :

Tunnel adapter Local Area Connection* 17:

    Connection-specific DNS Suffix . :
    IPv6 Address. . . . . : 2001:0:9d38:90d7:3cca:2f17:3f57:ffef
    Link-local IPv6 Address . . . . . : fe80::3cca:2f17:3f57:ffef%11
    Default Gateway . . . . . : ::

Tunnel adapter isatap.{9D552290-62C3-479B-A312-FAEA518B1655}:

    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix . :

Tunnel adapter isatap.{184652AE-5DF0-470C-84BE-B4D09760D3C9}:

    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix . :

C:\windows\system32>

```



Note: Your organization's VPN might be configured with a relatively short lease time, so that you might need to re-establish the SafeNet Remote PED connection at intervals of hours or days, providing the newly assigned IP address of your PEDServer computer each time.



Note: We generally advise not specifying the IP address when starting the PED server, unless you have a specific reason to set an address there. Just say "pedserver -mode start". In a volatile network or VPN situation, this means that, when the host IP changes on the PED server, only pedclient needs restarting with the new pedserver IP address. There is no need to also stop-and-restart pedserver.exe with a new IP.

Once started, pedserver.exe remains on, and listening until you explicitly tell it to stop, or until the host computer stops.

To configure the HSM to use the remote PED



Note: For the purposes of the PEDClient (the HSM that seeks a Remote PED connection) you can specify the PEDServer's IP address and listening port each time you connect. Or you can use the `lunacm:> ped set` command to configure either, or both of those parameters, which are then picked up by the `lunacm:> ped connect` command when you wish to establish the connection.

If the listening port of the PEDServer is not specified, then the default value "1503" is assumed. The IP address must be specified somewhere; there is no original default. If an IP address or a port is specified in the `lunacm:> ped connect` command, it overrides any value that was set by `lunacm:> ped set`, but only for the current connection.

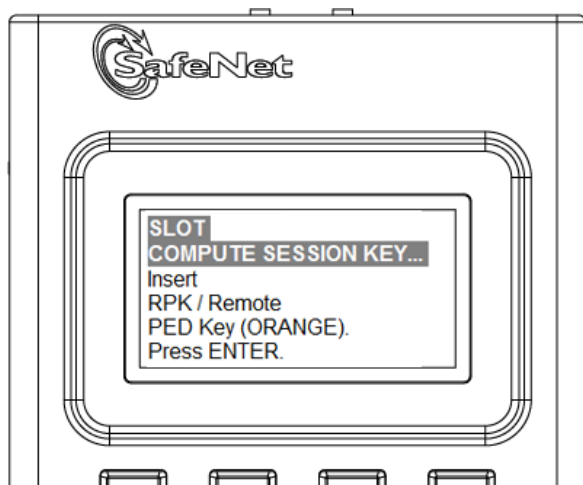
1. Launch the PEDClient on your HSM server, identifying the PEDServer instance (configured above) to which the HSM is to connect for its authentication requirements.

Type `lunacm:> ped connect -ip <pedserver ip> -port <pedserver listening port>`
 (substituting your actual PEDServer IP and port)

for example: `lunacm:> ped connect -ip 182.16.153.114 -port 1503`

SafeNet PED operation required to to connect to Remote PED - use orange PED Key(s).

At this point, the remote SafeNet PED should come to life, briefly saying "Token found..." followed by this prompt:



2. Insert the orange PED Key that you brought from the HSM to the remote PED, and press [Enter] on the PED keypad.

When the orange PED Key is accepted, control returns to the HSM command-line with a success message:
 "Command Result : 0 (Success)"

Once you have reached this point, you can continue to issue HSM or Partition commands, and whenever authentication is needed, the Remote PED will prompt for the required PED Key and associated key-presses.

Relinquishing Remote PED

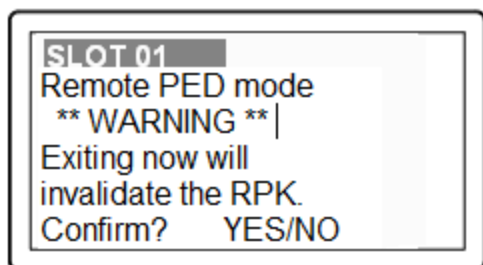
The PEDServer utility continues to run until explicitly stopped.

On the HSM end, PEDClient (launched by the "connect" command) continues to run until you explicitly stop with the "disconnect" command, or the link is broken. At any time, you can run the command in "show" mode to see what state it is in.

If you physically disconnect the Remote PED from its host, the link between PEDClient and PEDServer is dropped.

If the network connection is disrupted, or if your VPN closes, the link between PEDClient and PEDServer is dropped.

If you attempt to change menus on the Remote PED, the PED warns you:



If you persist, the link between PEDClient and PEDServer is dropped.

If the "IdleConnectionTimeoutSeconds" is reached, the link between PEDClient and PEDServer is dropped. The default is 1800 seconds, or 30 minutes. You can modify the default value with the "-idletimeout" option.

Any time the link is dropped, as long as the network connection is intact (or is resumed), you can restart PEDClient and PEDServer to reestablish the Remote PED link. In a stable network situation, the link should remain available until timeout.

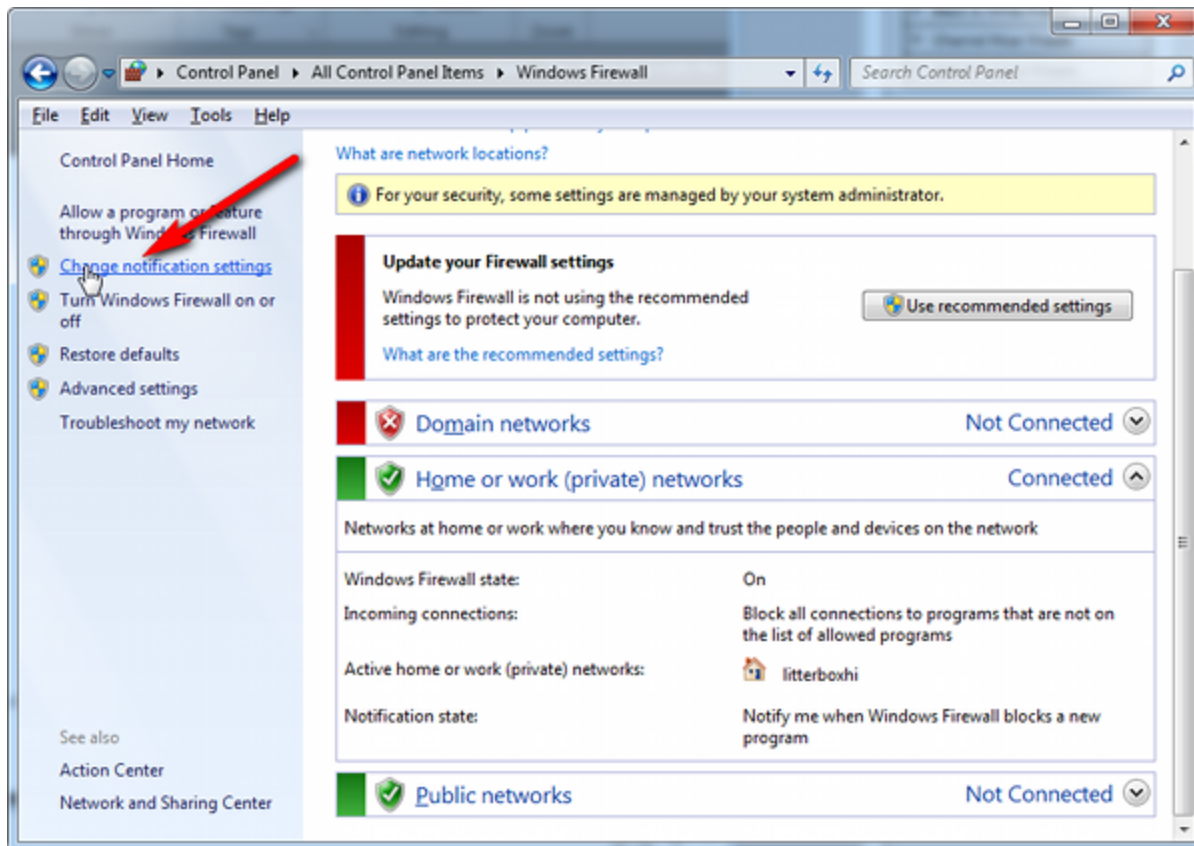
Troubleshooting

Here are some suggestions for addressing some possible issues when configuring SafeNet Remote PED.

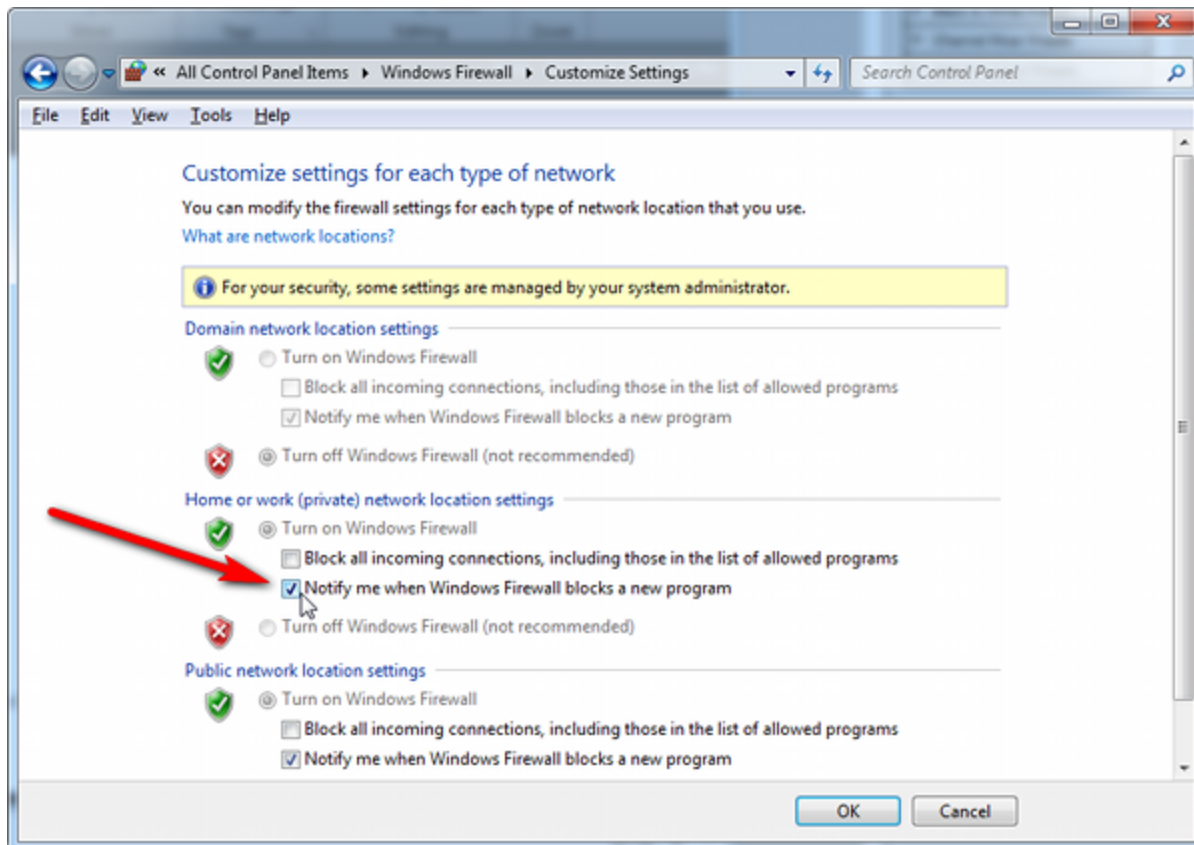
Firewall blocking

If you experience problems while attempting to configure a SafeNet Remote PED session over VPN, you might need to adjust Windows Firewall settings.

1. From the Windows Start Menu, select "Control Panel".
2. From the "Control Panel", select "Windows Firewall".
3. From the "Windows Firewall" dialog, select "Change notification settings".



4. In the dialog "Customize settings for each type of network", go to the appropriate section and activate "Notify me when Windows Firewall blocks a new program".



Without this setting, it might not matter that you have Administrator-level privileges on the PEDServer host computer, because Windows would silently block the connection from PEDClient to PEDServer, and not give you an opportunity to exercise your power to approve the connection.

With notification turned on, a dialog box pops up whenever Windows Firewall blocks a program, allowing you to override the block, which permits the SafeNet Remote PED connection to successfully listen for PEDClient connections.

Port Access

Another possible issue is that some networks might be configured to block access to certain ports. If such policy on your network includes ports 1503 (the default PEDServer listening port) and 1502 the administrative port, then you might need to choose a port other than the default, when starting PEDServer, and similarly, when you launch the connection from the HSM end and provide the IP and port where it should look for the PEDServer. Otherwise, perhaps your network administrator can assist.

"Jump" Server [option]

An option that some customers use is a port-forwarding "jump" server, co-located with the SafeNet HSM appliances, on the datacenter side of the firewall. The datacenter is usually a very stable/static network environment. By contrast, a client host on a desktop in a corporate office is more likely to be separated from the internet by an assortment of switches, firewalls, routers, etc., subject to change for any number of reasons. Implementing a jump server can be a low-cost and useful addition:

- to get around port-blocking problems, or to be able to react quickly to shifts in the corporate port and routing environment,

- as a way to implement a PKI authentication layer for Remote PED, and optionally for other SSH access, by (for example) setting up smart-card access control to the jump server.

For our own test of the solution, we used a standard Ubuntu Server distribution, with OpenSSH installed. No other changes were made to the system from the standard installation.

1. Connect a SafeNet PED, set to Remote Mode, to a Windows host with SafeNet HSM Client installed, and PEDServer running (see above for details).
2. From the Windows host, in an Administrator Command Prompt, run this command:

```
plink -ssh -N -T -R 1600:localhost:1503 <user>@<IP of Linux Server>
```

3. From the SafeNet Network HSM, run the command:

```
hsm ped connect -ip <IP of Linux Server> -port 1600
```

The connection is made to the Windows host running PEDServer, via the Linux Server, through the SSH session that was initiated out-bound from the Windows host.

A variant of this arrangement has port 22 also routed through the jump server, which allows you to bring administrative access to the SafeNet appliance under the PKI access-control scheme.

Using the Remote PED Feature

To use Remote PED for the first time, you will need:

- a SafeNet PED 2.4.0-3 (or later) with Remote PED feature installed (new Remote PED units are shipped with this



sticker on the front)

- a power adapter for the Remote PED (when the PED is not connected to a SafeNet Network HSM, via the PED port, it requires the separate power adapter to supply its power - the USB connection is insufficient for that purpose)
- a complete set of PED Keys, including an orange Remote PED key (either new/empty or already containing a Remote PED vector)
- local access to the SafeNet HSM (for the first session only)
- HSM that supports the Remote PED feature (includes the Remote PED Client)
- a workstation/PC with the PEDserver.exe (Remote PED Server application) running, and with the appropriate PED driver already installed

You will need physical access to your SafeNet Network HSM when first setting up Remote PED, because the Remote PED vector must be created by the HSM and imprinted on a blank PED Key, or it must be acquired from a previously imprinted orange PED Key and stored in the HSM. Thereafter, the orange PED Key is used with the Remote PED from a remote location, and the connection is secured by having the matching Remote PED vector at both the HSM and the Remote PED server (your remote workstation with Remote PED attached).



Note: If you encounter timeout problems (possible if you are using MofN with many keys, or if you are reading instructions as you go, or are otherwise not speedy while following prompts), you can adjust timeout values to allow for a more relaxed pace. For PedServer.exe, you can do:

```
pedserver -mode config set -socketreadrsptimeout <seconds>
```

but you would also need to increase the timeout in the crystoki.ini client software configuration file. Moreover, the PEDServer -socketreadrsptimeout must always be larger than the timeout in the configuration file.



Note: In general, do not change settings (especially in the crystoki.ini file) unless you have good reason to do so, or are instructed to do so, by Gemalto Customer Support.

Use **static IP addressing** for PED Client / PED Server. PED Client can fail to find a server if a dynamic address is indicated. An example error might look like this:

```
lunash:>hsm ped connect -ip 192.20.11.67 -port 1503
Luna PED operation required to connect to Remote PED - use orange PED Key(s).
Ped Client Version 1.0.5 (10005)
Ped Client launched in startup mode.
readIPFromConfigFile() : config file did not contain an IP address.
Startup failed. : 0xc0000404 RC_FILE_ERROR
Command Result : 65535 (Luna Shell execution)
lunash:>
```



Note: If the HSM host (a SafeNet Network HSM appliance or a host computer with SafeNet PCIe HSM or SafeNet USB HSM) has more than one SafeNet HSM connected, then you might need to specify the "-serial" option, to identify the desired HSM by its serial number.

If "-serial" is not specified in commands

```
hsm ped vector init
hsm ped vector erase
hsm ped connect
hsm ped disconnect
```

then the action defaults to the first HSM that is found.

Prepare a Remote PED Vector

This section creates or copies a Remote PED Vector (RPV) such that the same RPV exists on the HSM and on an orange PED Key that can accompany the Remote PED, to permit a connection between that Remote PED and that HSM.

Setup Instructions

The steps to set up Remote PED are:

1. [Initialize the HSM](#) [if you have not already done so]- the creation of the orange Remote PED key requires HSM login; HSM login requires an initialized HSM, all of which must be done with a local PED connection the first time.
2. Have the SafeNet PED connected to the PED port of the HSM, and set to Local PED mode.
3. Login as SO:

```
[myluna] lunash:>hsm login
```

```
Luna PED operation required to login as HSM Administrator - use blue PED key(s).
'hsm login' successful.
Command Result : 0 (Success)
[myluna] lunash:>
```

4. Have a blank PED Key, with orange label, ready. Create and imprint the RPV (Remote PED Vector):

```
[myluna] lunash:>hsm ped vector init
```

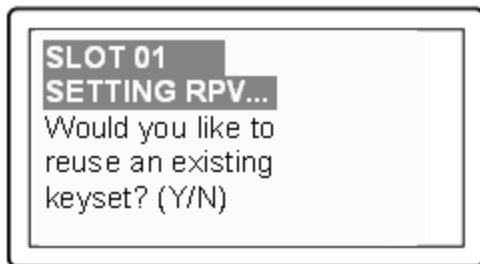
WARNING !! This command will initialize remote PED vector (RPV).

If you are sure that you wish to proceed, then enter 'proceed', otherwise this command will abort. > proceed

Proceeding... SafeNet PED operation required to initialize remote PED key vector - use orange PED key(s).

(At this time, go to the SafeNet PED and respond to the prompts by providing either a "fresh" orange PED key (which prompts creation and imprinting of a new/unique RPV) or an already-imprinted orange PED Key (which prompts the PED to ask you to reuse the existing PED Key data), along with additional blanks if you intend to make duplicates.)

The PED says:

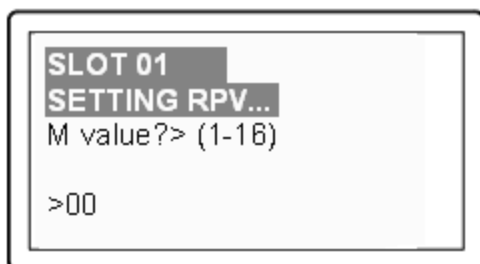


If this is the first RPV that you are creating, then answer [NO].

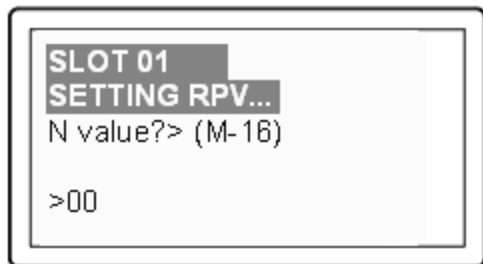
If you have an existing RPV on an orange PED Key, then answer [YES] if you want to preserve it and add it to this current HSM, or [No] if you have made a mistake and wish to find a different blank (or outdated) key to imprint.

For this example, we will assume no existing RPV.

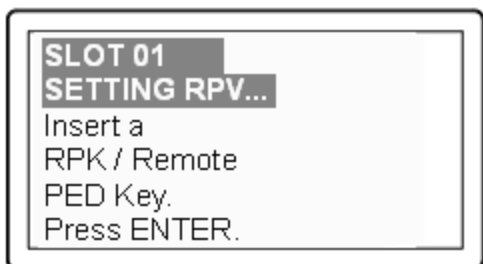
The PED says:



If you wish to split the RPV secret over several RPKs, for MofN split-knowledge, multi-person access control of the Remote PED function, then input a value for M that is greater than "1". This is the number of persons - each holding an orange key containing a split of the RPV secret - who must come together and present their portions whenever the RPK is required. If you prefer not to invoke MofN, then press [1], followed by [Enter].

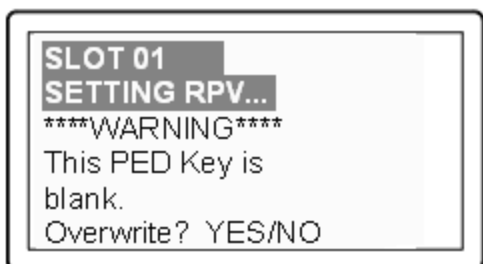


If you have invoked MofN with an M value greater than "1", then you must enter a value for N that is equal to, or greater than, M. N is the total pool of orange keys over which your RPV will be split, from which sub-groups of quantity M will be required for authentication. The simplest scheme is to declare a value for M that gives you the desired security oversight of the Remote PED function, and then specify N slightly larger so that you can always have at least quantity M key-holders available, even when some are absent for vacation, travel, illness or other reasons. Example: M=3, N=5, where any 3 of the total 5 splits can combine to reconstitute the secret.

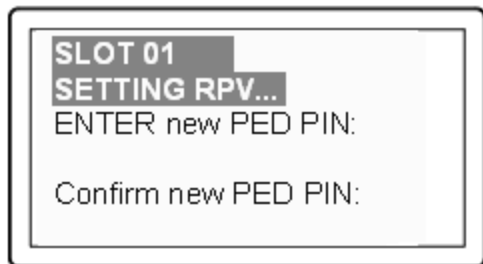


Do as prompted, inserting an unused PED Key into the PED's key slot (top-right of the PED), and press [ENTER].

For a fresh, new, never-before imprinted PED Key, the PED says:



Answer [YES] so that the HSM can create an RPV and transfer it to the PED, where it is imprinted onto the blank PED Key that you have inserted. If you invoked MofN, then the PED will prompt you to continue inserting orange PED Keys for imprinting with portions of the secret until you have imprinted quantity N of them.

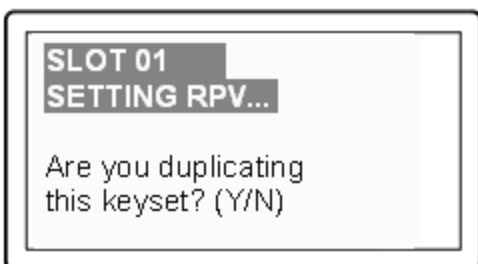


If you need two-part security to protect the Remote PED function, and wish to add a "something you know" component to the "something you have" (physical PED Key), type a series of digits on the keypad, then type them again to confirm. Now, whenever you are prompted to present the orange RPK, you must also input the code - called a PED PIN - that you have just added. The secret that unlocks the HSM to perform Remote PED operation is now a combination of a data secret contained in the physical key, and a typed-in numeric code that you must remember.

Press [Enter] with no digits, if you do not wish an additional "something you know" secret attached to this PED Key. In future, SafeNet PED will nevertheless prompt you for a PED PIN whenever you present the RPK, but you will always just press [Enter] (with no digits) at that prompt - no PED PIN required.

This completes the imprinting of the key (or keys if you opted for MofN).

While the imprinted orange PED Key is still in the PED's slot, SafeNet PED then wants to know if you intend to make some copies of the currently-inserted PED Key (that now carries the RPV for the HSM) or group of PED Keys:



Answer [YES] if you wish to make copies, and follow the instructions to insert keys and press ENTER. Respond to the prompts about overwriting, and PED PIN, etc.

When you have made all the copies that you wish, respond [NO] to the final prompt.

Control is returned to the lunash command line.

```
Ped Client Version 1.0.0 (10000)
Ped Client launched in shutdown mode.
Ped Client is not currently running.
Shutdown passed.
Command Result : 0 (Success)
[myluna] lunash:>
```

(If you see references to "shutdown mode", that's the shell [lunash] exchanging messages with the Remote PED

Client application (also found on your SafeNet appliance), which is called, runs in the background, and shuts down, possibly multiple times, depending upon the task that you have initiated via [lunash:>] commands.)

5. Go to either Client-(HSM)-initiated Remote PED ("[Using Client-initiated Remote PED Connection](#)" below) or Server-initiated Remote PED ("[Using Server-initiated \(Peer-to-Peer\) Remote PED Connection](#)" on page 298).

Using Client-initiated Remote PED Connection

At this point, you have a Remote PED Vector (RPV) shared between at least one orange PED Key and at least one HSM. The RPV is the means by which a Remote PED server authenticates to an HSM, allowing the HSM to accept other PED Key values (blue, red, black, gray, white, purple) from a SafeNet PED that is not directly, physically connected.

The following section details how to launch that connection from the HSM host (which is considered the client in the Remote PED world). If you have firewall or other constraints that prevent the initiation of a connection from your HSM host out to the PedServer in the external network, then see "[Using Server-initiated \(Peer-to-Peer\) Remote PED Connection](#)" on page 298, instead. The general process for client-initiated Remote PED is the same for Remote PED service toward

- a host computer containing an internally installed SafeNet PCIe HSM
 - a host computer containing an externally connected SafeNet USB HSM
 - a SafeNet Network HSM appliance.
1. Bring a SafeNet PED 2 with Remote PED capability, the PED Keys (blue and black and red), and at least one imprinted orange PED Key to the location of your compatible Windows workstation computer (anywhere in the world with a suitable network connection). You should already have the most recent PED driver software and the PedServer.exe software installed on that computer.



Note: The software and driver are provided on the SafeNet Client installer, but are optional during the installation process. If you intend to use Remote PED (and therefore need the PED driver and the PedServer executable program), ensure that Remote PED is among the options selected during installation. Alternatively, you can launch the installer at a later time and modify the existing SafeNet HSM Client installation to include Remote PED at that time.

When you connect your SafeNet PED2 Remote to electrical mains power (AC power outlet) and to your computer's USB port, the operating system detects the new hardware and should locate the appropriate driver. If that does not happen, then the system presents a dialog for you to help it find the location where the LunaPED driver has been placed.

2. Connect the Remote PED to its power source via the power adapter.
3. Connect the Remote PED to the workstation computer via the USB cable.
4. When the PED powers on and completes its self-test, it is in Local PED mode by default. Press the [<] key to reach the "Select Mode" menu. Press [7] to enter Remote PED mode.

- Open a Command Prompt window on the computer (for Windows 7, this must be an Administrator Command Prompt), locate and run PedServer.exe (we suggest that you try it out beforehand, to become familiar with the modes and options - if you experience any problem with PED operation timeout being too short, use "PedServer -mode config -set <value in seconds>" to increment the "sreadrsptimeout" value).

Set PedServer.exe to its "listening" mode.

```
c: > PedServer -m start
Ped Server Version 1.0.5 (10005)
Ped Server launched in startup mode.
Starting background process
Background process started
Ped Server Process created, exiting this process.
c:\PED\ >
```



Note: When running **pedserver -mode start** on an IPv6 network, you must include the **-ip <IPv6_address>** option.



Note: If you encounter a message "Failed to load configuration file...", this is not an error. It just means that you have not changed the default configuration, so no file has been created. The server default values are used.

- Open an ssh session to the SafeNet Network HSM appliance and login as admin.

- Start the PED Client (the Remote PED enabling process on the appliance):

```
lush:> hsm ped connect -i 183.21.12.161 -port 1503
Luna PED operation required to connect to Remote PED - use orange PED key(s).
Ped Client Version 1.0.0 (10000)
Ped Client launched in startup mode.
Starting background process
Background process started
Ped Client Process created, exiting this process.
Command Result : 0 (Success)
[luna27] lush:>
```



Note: The serial number option on command **hsm ped connect** is needed if you are using Remote PED with an HSM other than the on-board SafeNet Network HSM (such as a connected SafeNet USB HSM for PKI). If a serial number is not specified, the internal HSM is assumed by default.

- To verify that the Remote PED connection is functional, try some HSM commands that require PED action and PED Key authentication - the simplest is **hsm login**. First logout, because you were already logged in to the HSM...

```
[luna27] lush:>hsm logout
'hsm logout' successful.
Command Result : 0 (Success)
[luna27] lush:>hsm login
Luna PED operation required to login as HSM Administrator - use Security Officer (blue)
PED key.
'hsm login' successful.
Command Result : 0 (Success)
```

```
[luna27] lush:>
```

9. At this point, you have successfully set up a Remote PED link between a workstation computer (with PED attached to its USB port) and a distant SafeNet Network HSM/appliance. You have demonstrated that success by performing an HSM operation that demanded SO/HSM Admin PED Key authentication, without being physically near to the SafeNet Network HSM/appliance, and without having a SafeNet Network HSM PED directly attached to the SafeNet Network HSM/appliance.

You can now perform any HSM administration chores (including Cluster creation/administration) as though you were physically adjacent to the HSM, with equal confidence in the security of the system [HSM products that include Remote PED are now routinely submitted to approving agencies (like NIST/FIPS) for validation].

10. To disconnect:

```
[luna27] lush:>hsm ped disconnect
```

```
WARNING !! This command will disconnect remote PED.
```

```
If you are sure that you wish to proceed, then enter 'proceed', otherwise this command will abort.
```

```
> proceed
Proceeding...
```

```
Ped Client Version 1.0.0 (10000)
```

```
Ped Client launched in shutdown mode.
```

```
Shutdown passed.
```

```
Command Result : 0 (Success)
```

```
[luna27] lush:>
```

Note: If a Remote PED session is in effect and you press the [<] key on the PED (to go to the PED's "Select mode" menu), that action amounts to exiting the Remote PED mode.

Therefore, the PED displays a message:

```
** WARNING **
Exiting now will
invalidate the RPK.
Confirm ?      YES/NO
```



If you press [YES], the RPK-validated Remote PED session is dropped and must be re-established from the HSM (with "hsm ped connect <network-target>" before you can resume activity with the Remote PED.

In other words, if you want to use that PED for any other purpose than the current connection with one remote HSM, you have to drop the current session to make such other use of the PED, and then have the appropriate RPK available when you are ready to re-establish the prior Remote PED connection.)



Note: The above note talks about a "session" that exists only between the Remote PED and the computer (actually the PedServer software running on that computer) to which the Remote PED is connected. That is separate from the session that was established between the distant appliance/HSM and the PedServer on your computer. The session between computer and HSM is time-sensitive - it is in existence while needed and is either dropped intentionally or times out after brief inactivity. The session between the Remote PED and its attached computer persists until you disconnect the PED or change modes, or until you stop the PedServer.exe process on the computer.

******* The default timeout for a Remote PED link between PedClient at the HSM and PedServer at the Remote PED, is 1800 seconds, or 30 minutes. If no Remote PED activity is requested for the entire timeout duration, the link ends, and must be re-established. While that link is down, and the HSM remains set to expect Remote PED operation, any requested PED operations simply fail. We recommend performing a disconnect before performing a connect, to ensure that the old link is cleanly severed and that a new link is cleanly established. *******



Note: PED KEY MIGRATION from older classic-PED Datakeys (the PED Keys that look like toy plastic keys) is NOT SUPPORTED over Remote PED, because the old classic PED 1.x has no way to connect to the PED Server. Migration of PED Keys from DataKeys to iKeys must be done locally.)

HSM Selection

As a user of the HSM (or an application partition on that HSM) wanting to perform an HSM operation that requires a PED operation, do the following:

1. From LunaSH, run command **hsm ped select -h <hostname>**. The <hostname> is the PED Server hostname.



Note: The two LunaSH commands **hsm ped deselect -host <hostname>** and **hsm ped select -host <hostname> -serial <serial number>** both support peer-connection mode.

- PED Client sends a message to the PED Server with the HSM serial number to notify that the PED Server is now selected for PED operations.
 - PED Server receives the message and updates the processing status from waiting to process commands (read and write commands from and to the PED).
2. A user of the HSM (or an application partition of the HSM) executes an operation that requires authentication via PED. The behavior is the same as for non-peer mode if the connection was initiated from the HSM side.



Note: There is no timeout for the connection between PED Server and PED Client when using the server-initiated (peer-to-peer) mode of connection.

If you need to deselect the PED Server, run **hsm ped deselect -host <hostname>**.

1. PED Client sends a message to the PED Server that it is no longer selected.
2. PED Server acknowledges the message and resets the PED to clear the current session ID and the generated Diffie-Hellman key.

3. PED Server sets the PED to stand-by. Any additional read and write command from PED Client is ignored and is logged for security and debugging purposes.

If the user executes the disconnect command in PED Server, or if the connection is terminated abnormally, the PED Client receives the message and removes that PED Server from the connection table.

If you encounter problems with Remote PED, "[Troubleshooting Remote PED](#)" on page 303.

Using Server-initiated (Peer-to-Peer) Remote PED Connection

By default, when Remote PED is needed, a SafeNet HSM uses a local instance of PEDClient to initiate a connection with a distant instance of PEDServer. In cases where a SafeNet Network HSM resides behind a firewall with rules prohibiting the HSM host from initiating external connections, it is possible to have the PEDServer perform the initial call toward the HSM host in peer-connection mode.

The default mode (initiated by PEDClient) and the peer-connection mode (initiated by PEDServer) are mutually exclusive.



Note: The server-initiated (or peer-to-peer) Remote PED connection is carried over a TLS channel and secured by exchanged certificates from the participants.

Server-initiated connection mode is configured by two commands on the PedServer end:

- **PedServer.exe –appliance register**
- **PedServer.exe –appliance delete**

For use with SafeNet Network HSM, the path to the pedServer.ini file and to the PEDserverCAFile.pem must be in the Remote PED server host's crystoki.ini file, and HSM appliance's server certificate must be added to the Remote PED server host's PEDserverCAFile.pem file. The server.pem is secure-copied from the SafeNet Network HSM appliance, and **PedServer.exe -appliance register** command adds it to the PEDserverCAFile.pem file in the cert folder.

Server-initiated connection mode is enacted by two commands:

- **PedServer.exe –mode connect**
- **PedServer.exe –mode disconnect**

The **PEDServer.exe -mode disconnect** command is used to terminate any existing peer connection with the intended HSM host, before a new connection can be launched.

The PedClient on the SafeNet Network HSM appliance runs in background and listens on port 9697 for incoming Remote PED peer connection requests. You can specify different ports if needed, at both the PedClient and PedServer ends.

Three Lunash commands support peer-connection mode:

- **hsm ped server register -certificate <filename>**
- **hsm ped select –host <hostname> -serial <serial number>**
- **hsm ped deselect –host <hostname>**

Constraints

The following constraints apply:

- This feature is not currently supported for use with IPv6 networks.
- A maximum of twenty connections is supported on the PedClient.
- If the connection is terminated abnormally (for example, router switch died), there will be no auto-connection.
- When running in peer connection mode, the PedServer will have the listening service (the default mode) down for security reasons and to simplify the usability. That is, if you have set the PedServer for server-initiated connection, then the PedServer stops listening for a PedClient to attempt a connection.
- Once the PedServer connection to the PedClient is established, the connection remains up until
 - disconnect command is executed from the PedServer
 - PedClient terminates the connection

Configuration Prerequisite

The following prerequisites are necessary to establish a functioning server-initiated Remote PED link.

- The PedClient (SafeNet Network HSM in this case) and the PedServer must be network accessible to each other.
- There must be no blocking firewall rules or other impediments to performing a certificate exchange and establishing a secure connection. Contact your network administrator, if this is an issue.

PedServer has the commands to create a host certificate if necessary and to register a retrieved server certificate obtained from the HSM appliance. Upload/download of the certs is done with scp/pscp (provided).

Physical setup for server-initiated Remote PED connection

Refer to the *Installation Guide* for detailed hardware and software instructions, with diagrams.

To set up the Remote PED server

1. Bring a SafeNet PED 2 with Remote PED capability, the PED Keys (blue and black and red), and at least one imprinted orange PED Key to the location of your compatible Windows workstation computer (anywhere in the world with a suitable network connection). You should already have the most recent PED driver software and the PedServer.exe software installed on that computer.



Note: The software and driver are provided on the SafeNet Client installer, but are optional during the installation process. If you intend to use Remote PED (and therefore need the PED driver and the PedServer executable program), ensure that Remote PED is among the options selected during installation. Alternatively, you can launch the installer at a later time and modify the existing SafeNet HSM Client installation to include Remote PED at that time.

When you connect your SafeNet PED2 Remote to electrical mains power (AC power outlet) and to your computer's USB port, the operating system detects the new hardware and should locate the appropriate driver. If that does not happen, then the system presents a dialog for you to help it find the location where the LunaPED driver has been placed.

2. Connect the Remote PED to its power source via the power adapter.
3. Connect the Remote PED to the workstation computer via the USB cable.
4. When the PED powers on and completes its self-test, it is in Local PED mode by default.
Press the [<] key to reach the "Select Mode" menu.
Press [7] to enter Remote PED mode.
5. Open a Command Prompt window on the computer (for Windows 7, this must be an Administrator Command Prompt), locate and run PedServer.exe (we suggest that you try it out beforehand, to become familiar with the modes and options - if you experience any problem with PED operation timeout being too short, use "PedServer -mode config -set <value in seconds>" to increment the "sreadrsptimeout" value).
Set PedServer.exe to its "listening" mode.

```
c: > PedServer -m start
Ped Server Version 1.0.5 (10005)
Ped Server launched in startup mode.
Starting background process
Background process started
Ped Server Process created, exiting this process.
c:\PED\ >
```

NOTE: if you encounter a message "Failed to load configuration file...", this is not an error. It just means that you have not changed the default configuration, so no file has been created. The server default values are used.

Setting up the server-initiated Remote PED connection

To set up the connection between Remote PED server and HSM appliance

Below is a step by step connection setup between PedServer and PedClient:

1. If the PedServer host does not have a certificate, create one with command:

```
pedServer -regen -commonname <common name>
```

(Create the new cert if you do not already have a host certificate, otherwise keep and reuse an existing certificate if it is already in use for other purposes.)
2. Secure copy (SCP or PSCP) the host certificate to the admin account on the SafeNet Network HSM appliance.

```
pscp <pedserver-host-certificate>.pem admin@<hsm-appliance-hostname-or-ip>:
```
3. Secure copy (SCP or PSCP) the server.pem from SafeNet Network HSM appliance to the PedServer host.

```
pscp admin@<hsm-appliance-hostname-or-ip>:server.pem .
```

(The dot at the end ensures that the incoming file lands in the current folder.)
4. Register the server.pem with the **pedServer appliance register** command.

```
pedServer -appliance register -name <unique name> -certificate <Network HSM certificate file> -ip <Network HSM address> [-port <port number>]
```
5. In an SSH session to the admin account on the SafeNet Network HSM appliance, register the PedServer host certificate with the appliance, using the **hsm ped server register** command.

```
lunash:>hsm ped server register -certificate <pedserver-cert-name>.pem
```

6. Connect to the PedClient with command:

```
pedServer -mode connect -name <SafeNet HSM server name>
```

What happens

- a. PedClient receives the TLS connection from the PedServer by listening at port 9697 (unless a different port was specified).
- b. PedClient validates the PedServer client certificate.
- c. PedClient sends the client information identity to the PedServer.
- d. PedServer receives the client information identity and sends its own identity to the PedClient.
- e. PedClient receives the server information identity and adds to the connection table.
- f. PedClient sends a message back to the PedServer that the SSL connection is initialized and ready to go.

At this point, the secure network connection is in place between the PedServer and PedClient, which might be one of several PedServers available and connected to that PedClient, but the current PedServer is not selected to perform PED actions for the HSM associated with that PedClient. The PedClient might have another of its connected/available PedServers selected, or it might have none selected.

To open the now-established Remote PED connection for use

As a user of the HSM (or an application partition on that HSM) wanting to perform an HSM operation that requires a PED operation do the following:

1. From Lunash, run command:

```
hsm ped select -h <hostname> from Lush.
```

The hostname is the PedServer hostname (or IP address if that was used in the certificate).

What is happening

- a. PedClient sends a message to the PedServer with the HSM serial number to notify that the PedServer is now selected for PED operations.
 - b. PedServer receives the message and updates the processing status from waiting to process commands (read and write commands from and to the PED).
2. A user of the HSM (or an application partition of the HSM) executes an operation that requires authentication via PED.

What is happening

- a. The behavior is the same as for non-peer mode if the connection was initiated from the HSM side.

Deselecting a PedServer, from the appliance

If you need to deselect the PedServer, do the following:

1. From Lunashell run the hsm ped deselect command

```
hsm ped deselect [-host <hostname>]
```

What is happening

- a. PedClient sends a message to the PedServer that it is no longer selected.
- b. PedServer acknowledges the message and resets the PED to clear the current session ID and the generated Diffie-Hellman key.

- c. PedServer sets the PED to stand-by. Any additional read and write command from PedClient is ignored and is logged for security and debugging purposes.

To disconnect, from the PedServer host side

1. Use the **PedServer -mode disconnect** command

```
pedServer -mode disconnect
```

The connection is terminated.

pedServer Configuration File

Peer-to-peer Remote PED introduces a pedServer.conf or pedServer.ini file with SafeNet HSM Client 6.2.1 and newer.

```
RemotePed = {
PongTimeout = 5;
PingInterval = 1;
LogFileTrace = 0;
LogFileError = 1;
LogFileWarning = 1;
LogFileInfo = 1;
MaxLogFileSize = 4194304;
LogFileName = ./remotePedServerLog.log;
BGProcessShutdownTimeoutSeconds = 25;
BGProcessStartupTimeoutSeconds = 10;
InternalShutdownTimeoutSeconds = 10;
SocketWriteTimeoutSeconds = 50;
SocketReadRspTimeoutSeconds = 180;
SocketReadTimeoutSeconds = 100;
ExternalServerIF = 1;
ServerPortValue = 1503;
ExternalAdminIF = 0;
AdminPort = 1502;
IdleConnectionTimeoutSeconds = 1800;
RpkSerialNumberQueryTimeout = 15;
}
Appliances = {
SSLConfigFile = /usr/safenet/lunaclient/bin/openssl.cnf;
ServerCAFile = /root/CAFile.pem;
ServerIP00 = 192.20.11.86;
ServerPort00 = 9696;
ServerName00 = eddiebox;
CommonCertName00 = test1;
ServerName01 = devbox;
ServerIP01 = 192.20.9.46;
ServerPort01 = 9697;
CommonCertName01 = test2;
}
```

The Appliances section manages registered appliances.

A new entry in the main Crystoki.ini / chrystoki.conf file points to the location of the pedServer.ini or pedServer.conf file.

```
[Ped Server]
PedConfigFile = /usr/safenet/lunaclient/data/ped/config
```

Troubleshooting Remote PED

This section describes how to recognize and fix some problems that you could encounter.

Ped connect can fail if IP is not accessible

On a system with two network connections, if pedserver attempts to use an IP address that is not accessible externally, then command lunacm:>ped connect can fail.

Here is an example:

This host computer is accessible through 192.20.10.175 and has an additional IP address 192.168.72.1 (that is not accessible).

Ethernet adapter VMware Network Adapter VMnet8:

```
Connection-specific DNS Suffix . :
IP Address. . . . . : 192.168.20.1
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . :
```

Ethernet adapter Local Area Connection:

```
Connection-specific DNS Suffix . :
IP Address. . . . . : 192.20.10.175
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . : 192.20.10.10
```

Ethernet adapter Wireless Network Connection:

```
Media State . . . . . : Media disconnected
```

Ethernet adapter VMware Network Adapter VMnet1:

```
Connection-specific DNS Suffix . :
IP Address. . . . . : 192.168.72.1
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . :
```

Command lunacm:> pedserver -m show returns

```
Ped Server Version 1.0.5 (10005)
Ped Server launched in status mode.
```

Server Information:

Hostname: noi1-502192
IP: 192.168.72.1
Firmware Version: 0.0.0-0
PedII Protocol Version: 0.0.0-0
Software Version: 1.0.5 (10005)

Ped2 Connection Status: Disconnected
Ped2 RPK Count 0
Ped2 RPK Serial Numbers (none)

Client Information: Not Available

Operating Information:
Server Port: 1503
External Server Interface: Yes
Admin Port: 1502
External Admin Interface: No

Server Up Time: 5 (secs)
Server Idle Time: 5 (secs) (100%)
Idle Timeout Value: 1800 (secs)

Current Connection Time: 0 (secs)
Current Connection Idle Time: 0 (secs)
Current Connection Total Idle Time: 0 (secs) (100%)
Total Connection Time: 0 (secs)
Total Connection Idle Time: 0 (secs) (100%)

To resolve this problem

1. Ensure that Pedserver is listening on the IP address that is accessible from outside.
2. If that condition (step 1) is not the case then disable the network connection on which Pedserver is listening.
3. Restart Pedserver and confirm that Pedserver is listening on the IP address that is accessible from outside.

VPN

If pedserver is running on a laptop that might change location, it can happen that the active network address changes, even though the laptop is not shutdown. An example might be if you unplugged from working at home, over the corporate VPN, commuted to the office, and reconnected the laptop there. The IP address that your laptop is assigned when joining directly to the office network would be different from the address that had been assigned by the VPN server.

In that situation, pedserver is still configured with the address you had while using the VPN. That pedserver session cannot work with the new address that is now assigned to your laptop. Running pedserver -mode stop does not completely clear all settings, so running pedserver -mode start again fails with a message like "Startup failed. : 0x0000303 RC_OPERATION_TIMED_OUT".

To resolve this problem

To resolve this problem,

1. Close the current Command Prompt window.

2. Open a new Command Prompt.
3. Verify the current IP address with command **ipconfig**
4. Run **pedserver -mode start -ip <new-ip-address> -port <port-number>** and it should now succeed.

Timeout

The default timeout for a Remote PED link between PedClient at the HSM and PedServer at the Remote PED, is 1800 seconds, or 30 minutes. If no Remote PED activity is requested for the entire timeout duration, the link ends, and must be re-established. While that link is down, and the HSM remains set to expect Remote PED operation, any requested PED operations simply fail. We recommend performing a disconnect before performing a connect, to ensure that the old link is cleanly severed and that a new link is cleanly established.

Pedserver fails to start with "LOGGER_init failed"

The pedserver.exe process must be run using administrator privileges. If you forget and accidentally launch pedserver.exe in an ordinary, non-privileged-user command-prompt window, the pedserver fails to work, but the process is launched and continues in the background. If you then attempt to launch pedserver.exe from an Administrator command prompt, it fails with message "LOGGER_init failed". The logger is a necessary service for pedserver, and has failed to initialize for the new attempt because the earlier, non-functional instance of pedserver has locked the logger.

To resolve this problem

If pedserver fails with the message "LOGGER_init failed", proceed as follows:

1. Check that the pedserver process is not already running.
2. If it is, stop the process (to free the logger service).
3. Start the pedserver process again, as Administrator.

DEK establishment is based on the Diffie-Hellman key establishment algorithm and an RPV (Remote PED Vector), shared between the HSM and the PED via the orange Remote PED iKey (RPK). Once a common Diffie-Hellman value is established between the parties via the Diffie-Hellman handshake, the RPV is mixed into the value to create a 256-bit AES DEK on each side. If the PED and the HSM do not hold the same RPV, the resulting DEKs will be different between them, and the parties won't be able to talk. This property is used in providing mutual authentication of the PED and the HSM.

Mutual authentication is achieved by exchanging random nonces, encrypted using the derived data encryption key. The authentication scheme operates as follows:

HSM	–	Remote PED
Send 8 bytes random nonce, R1, encrypted using the derived encryption key.	$\{R1 \parallel \text{padding}\}_{Ke} \rightarrow$	
	$\leftarrow \{R2 \parallel R1\}_{Ke}$	Decrypt R1. Generate an 8 byte random nonce, R2. Concatenate R2 R1 and encrypt the result using the derived encryption key.
Decrypt R2 R1. Verify that received R1 value is the same as the originally generated value. Re-encrypt R2 and return it to Remote PED.	$\{\text{padding} \parallel R2\}_{Ke} \rightarrow$	Verify that received R2 value is the same as the originally generated value.

Following successful authentication, the random nonce values are used to initialize the feedback buffers needed to support AES-OFB mode encryption of the two communications streams (one for each direction).

Sensitive data in transition between a PED and an HSM is end-to-end encrypted: plaintext security-relevant data is never exposed beyond the HSM and the PED boundaries at any time. The sensitive data is also hashed, using a SHA-256 digest, to protect its integrity during transmission.

Removing/Destroying Content for Safe Disposal

During the lifetime of a Thales Luna HSM, you might have cause to take the HSM out of service, and wish to perform actions that assure no trace of your sensitive material remains. Those events might include:

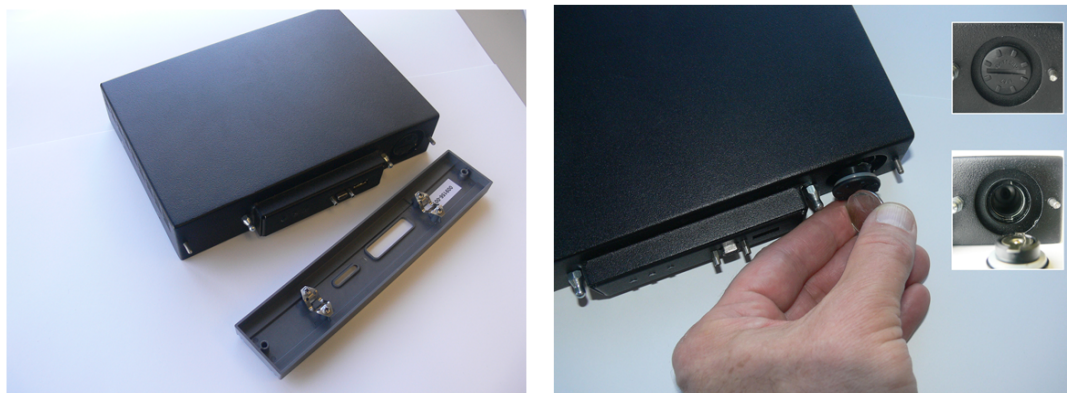
- placing the unit into storage, perhaps as a spare
- shipping to another location or business unit in your organization
- shipping the unit back to SafeNet/Gemalto for repair/re-manufacture
- removing the HSM permanently from operational use, for disposal at end-of-life

This chapter describes the available options in the following sections:

- ["Tamper/Declassify/Decommission the Thales Luna USB HSM" below](#)
- ["Resetting to Factory Condition" on page 1](#)
- ["End of service and disposal" on the next page](#)
- ["Comparison of Destruction/Denial Actions" on the next page](#)
- ["RMA and Shipping Back to SafeNet" on page 312](#)
- ["What Does Zeroized Mean?" on page 313](#)

Tamper/Declassify/Decommission the Thales Luna USB HSM

To ship back to Gemalto for repair, perform a zeroizing action, and tamper by disconnecting main power, and removing the battery.



To Declassify/Decommission the Thales Luna USB HSM for disposal, contact Thales technical support for instructions to physically drill/destroy critical parts.

The USB HSM differs from the PCIe HSM, and does not have a decommission button or exposed circuit-board header.

End of service and disposal

SafeNet (Thales Luna) HSMs and appliances are deployed into a wide variety of markets and environments. Arranging for the eventual disposal of a SafeNet HSM or HSM appliance that is no longer needed can be a simple accounting task and a call to your local computer recycling service, or it can be a complex and rigorous set of procedures intended to protect very sensitive information.

Needs Can Differ

Some users of SafeNet HSMs employ cryptographic keys and material that have a very short "shelf life". A relatively short time after the HSM is taken out of service, any objects that it contains are no longer relevant. The HSM could be disposed of, with no concern about any material that might remain in it.

The majority of our customers are concerned with their keys and objects that are stored on the HSM. It is important to them that those items never be exposed. The fact is that they are never exposed, but see below for explanations and actions that address the concerns of auditors who might be more accustomed to other ways of safeguarding HSM contents.

SafeNet HSM Protects Your Keys and Objects

The design philosophy of our SafeNet HSMs ensures that contents are safe from attackers. Unlike other HSM products on the market, SafeNet HSMs never store sensitive objects, like cryptographic keys, unencrypted. Therefore, SafeNet HSMs have no real need - other than perception or "optics" - to perform active erasure of HSM contents, in case of an attack or tamper event.

Instead, the basic state of a SafeNet (Thales Luna) HSM is that any stored keys and objects are strongly encrypted. They are decrypted only for current use, and only into volatile memory within the HSM.

If power is removed from the HSM, or if the current session closes, the temporarily-decrypted objects instantly evaporate. The encrypted originals remain, but they are unusable by anyone who does not have the correct HSM keys to decrypt them.

Comparison of Destruction/Denial Actions

Various operations on the Thales Luna HSM are intended to make HSM contents unavailable to potential intruders. The effect of those actions are summarized and contrasted in the following table, along with notes on how to recognize and how to recover from each scenario.

Event	MTK is destroyed HSM is unavailable, but use/access can be recovered after reboot (See Note 1)	Reset appliance admin password	How to discover (See Note 3)	How to recover
<ul style="list-style-type: none"> - three bad SO login attempts or - lunash:> hsm zeroize (for Thales Luna Network HSM), lunacm:> hsm zeroize (for Thales Luna PCIe HSM and Thales Luna USB HSM) or - lunash:> hsm factoryReset (for Thales Luna Network HSM), lunacm:> hsm factoryreset (for Thales Luna PCIe HSM and Thales Luna USB HSM) or - any change to a destructive policy or - firmware rollback (See Note 4) 	NO	NO	hsm.log entry or "Zeroized: Yes" in HSM Information (from hsm show command)	Restore HSM objects from Backup
<p>hardware tamper</p> <ul style="list-style-type: none"> - undervoltage or overvoltage during operation or - under-temperature or over-temperature during operation or - chassis interference (such as cover, fans, etc.) <p>OR</p> <p>software (command-initiated) tamper</p>	YES	NO	<p>Best practice - have external MTK split on SRK (purple PED Key), which forces administrative intervention to recover from tamper. Otherwise, parse hsm.log for text like "tamper", "TVK was corrupted", or "Generating new TVK", indicating that a tamper event was logged.</p> <p>Example: RTC: external tamper</p>	Reboot [See Note 1]

Event	MTK is destroyed HSM is unavailable, but use/access can be recovered after reboot (See Note 1)	Reset appliance admin password	How to discover (See Note 3)	How to recover
<ul style="list-style-type: none"> - lunash:> hsm srk transportMode enter or - lunacm:> srk transport 			latched/MTK: security function was zeroized on previous tamper event and has not been restored yet ... also, keywords in hsm.log like: "HSM internal error", "device error" Thales Luna Network HSM appliance front panel flashes error 30.	

Note 1: MTK is an independent layer of encryption on HSM contents, to manage tamper and Secure Transport Mode. A destroyed MTK is recovered on next reboot if no external split (SRK) of the recovery vector exists (that is, if both portions of the recovery key are available inside the HSM).

A destroyed MTK can be recovered if one of the recovery components has been moved outside the HSM onto a secure recovery key (SRK), and that SRK can be presented via Thales Luna PED at the next HSM reboot.

If MTK cannot be recovered, only restoring from backup onto a new or re-manufactured HSM can retrieve your keys and HSM data.

Note 2: If an external SRV exists, and you wish to stop using it (having no need for Secure Transport Mode, and no need for enforced administrative response in case of a tamper event), you must bring the external SRV back into the HSM with **srk disable** command. You may not simply destroy or overwrite the purple PED Key without first ensuring that the HSM is in possession of all the MTK recovery components.

Note 3: To check the health of a remote HSM, script a frequent login to the HSM host and execution of a subset of HSM commands. If a command fails, check the logs for an indication of the cause.

Note 4: These actions all create a situation where **hsm init** is required, or strongly recommended before the HSM is used again.

In addition, another event/action that has a destructive component is HSM initialization, which can be of either the "soft" or "hard" variety.

- HSM init is soft if you have not performed an **hsm factoryReset** before **hsm init**.
- HSM init is hard if it is performed following **hsm factoryReset**.

Either way, HSM and partition objects are gone, so only a restore from backup can bring them back. Effects of soft versus hard initializations are summarized below :

Condition/Effect	Soft init	Hard init
SO authentication required?	Yes	No
Can set new HSM label	Yes	Yes
Creates new SO identity	No	Yes
Creates new Domain	No	Yes
Destroys partitions	Yes	No (none exist to destroy) *
Destroys SO objects	Yes	No (none exist to destroy) *

* hsm factoryReset was performed, and destroyed partitions and objects, before the hard init... otherwise, it could not be a hard init.

RMA and Shipping Back to SafeNet

Although rare, it could happen that you need to ship a SafeNet HSM back to SafeNet.

You would deal through your SafeNet representative to obtain the Return Material Authorization (RMA) and instructions for packing and shipping.

However, you might wish to take the maximum precaution with any contents in your HSM before it leaves your possession. Or your security policy (or security auditors) might require it.

For HSM Network appliance

- Press the "decommission" button on the appliance back panel; this forcibly clears all HSM contents.
- If the appliance uses PED (Trusted Path) authentication, set the HSM into Secure Transport Mode (`hsm srk enable` (if not already enabled) followed by `hsm srk transportMode enter`), and simply do not send us the purple key. We have no way to access the HSM, and no choice but to remanufacture it.

For PCIe HSM

- Use a screwdriver or other conductive tool to bridge the two pins of the decommission header on the PCIe circuit board.
- If the HSM uses PED (Trusted Path) authentication, set the HSM into Secure Transport Mode (`hsm srk enable` (if not already enabled) followed by `hsm srk transportMode enter`), and simply do not send us the purple key. We have no way to access the HSM, and no choice but to remanufacture it.

For USB HSM

- The USB HSM does not have a "decommission" option; you can perform **hsm init** or make more than three bad login attempts on the SO, and perform **hsm factoryreset**

- If the HSM uses PED (Trusted Path) authentication, set the HSM into Secure Transport Mode (`hsm srk enable` (if not already enabled) followed by `hsm srk transportMode enter`), and simply do not send us the purple key. We have no way to access the HSM, and no choice but to remanufacture it.



Note: For Password-authenticated HSMs, the Secure Transport option is not available, as it is not possible to extract a portion of the SRK off the HSM.

What Does Zeroized Mean?

In the context of HSMs in general, the term to "zeroize" means to erase all plaintext keys. Some HSMs keep all keys in **plaintext** within the HSM boundary. Thales Luna HSMs do not.

In the context of Thales Luna HSMs, keys at rest [keys or objects that are stored in the HSM] are encrypted. Keys are decrypted into a volatile working memory space inside the HSM only while they are being used. Items in volatile memory disappear when power is removed. The action that we loosely call "zeroizing", or clearing, erases volatile memory as well as destroying the key that encrypts stored objects.

Therefore,

- if you perform `hsm factoryReset`, or
- if you make too many bad login attempts on the SO account, or

not only are any temporarily decrypted keys destroyed, but all customer keys on the HSM are immediately rendered inaccessible and unrecoverable.

User and Password Administration

This section describes tasks related to identities in the HSM or HSM partitions, including changing and resetting passwords, events or actions that cause HSM contents to be lost, and so on. It contains the following sections:

- ["About Changing HSM and Partition Passwords" below](#)
- ["Failed Logins" on page 316](#)
- ["Resetting Passwords" on page 318](#)
- ["Default Challenge Password" on page 321](#)

About Changing HSM and Partition Passwords

From time to time, you might have reason to change the various passwords on the appliance and HSM. This might be because a password has possibly been compromised, lost, or forgotten, or it might be because you have security procedures that mandate password-change intervals.

The two options are:

Action	Description	When used
Resetting PW	A higher authority sets a user's credentials back to a known default value (without requiring the knowledge or cooperation of the affected user),	<ul style="list-style-type: none"> • current holder has lost or forgotten his/her credential (forgot a password, misplaced a PED Key) • current credential is known or suspected to have become compromised • current holder has departed organization
contrasts with...		
Changing PW	The legitimate holder of the credential is able to log in with current credentials before directing the HSM, under the current logged-in user's own authority, to change that user's credential to a new value.	<ul style="list-style-type: none"> • credential holder suspects possible compromise of credential • credential holder is complying with organization security provisions (such as mandatory password-change interval)

HSM Passwords

Resetting HSM Password

There is no provision to reset the HSM Admin password (for Password Authentication) or PED Key (for Trusted Path), except to re-initialize the HSM, which zeroizes the contents of the HSM and of all Partitions on that HSM.

Resetting the password/authentication of a role or user requires a higher authority to invoke the reset. On the HSM, there is no authority higher than the SO / HSM Admin.

Changing HSM Password

To change the HSM password (for Password Authentication) or the secret on the blue PED Key (for Trusted Path), you must log in as HSM Admin using the current password (or blue PED Key). This is prompted by the `hsm changePw` command, so you do not need to log in separately.

```
lunash:> hsm changePw
Luna PED operation required to login as HSM Administrator - use Security Officer (blue) PED
key.
Command result : (0) success
lunash:>
```

Partition Passwords

A deliberate change to a Partition password is different from a [password reset](#). In both cases, the Partition or HSM contents remain intact.

Resetting Partition Password

- you must be logged in as HSM Admin, but
- you do not need to know the existing Partition password (for Password Authenticated systems) nor do you need to have the existing Partition Owner (black) PED Key (for Trusted Path Authenticated systems).

```
lunash:> partition resetPw -newpw mynewpw -partition mypartition1
```

Changing Partition Password

- you do not have to be logged in as HSM Admin or SO, but
- you do need to know the current Partition password. For Trusted Path HSMs, you must provide the current black PED Key.

```
lunash:> partition changePw -newpw mynewpw -oldpw myoldpw -partition mypartition1
```

You can choose not to include the passwords with the command, which:

- causes the system to prompt for old and new passwords (obscuring them with asterisks (*) for greater security, and
- presents additional options as shown in the example below.

For a PED-authenticated HSM, the following example changes only the challenge secret of the named partition, and leaves the black PED Key contents unchanged.

Example:

```
[myluna] lunash:>partition changepw -partition mypar1
```

Which part of the partition password do you wish to change?

1. change partition owner (black) PED key data
2. generate new random password for partition owner
3. specify a new password for the partition owner
4. both options 1 and 2

0. abort command

Please select one of the above options: 3

Please enter the password for the partition:

> *****

Please enter a new password for the partition:

> *****

Please re-enter password to confirm:

> *****

Luna PED operation required to activate partition on HSM - use User or Partition Owner (black) PED key.

'partition changePw' successful.

Command Result : 0 (Success)

[myluna] lunash:>

Failed Logins and Forgotten Passwords

"Failed Logins" below.

Appliance

For password changes affecting the appliance, not including the HSM .

Failed Logins

If you fail three consecutive login attempts as HSM Security Officer (or SO), the HSM contents are rendered unrecoverable. This is a security feature (you DO have your important material backed up, don't you?) meant to thwart repeated, unauthorized attempts to access your cryptographic material. The number is **not** adjustable.

Note: The system must actually receive some erroneous/false information before it logs a failed attempt -- if you merely forget to insert a PED Key (for PED-authenticated HSMs), or inserted the wrong color key, that is not counted as a failed attempt.



To fail a login attempt on a Password-authenticated HSM, you would need to type an incorrect password. To fail a login attempt on a PED-authenticated HSM, you would need to insert an incorrect PED Key of the correct color, or to type an incorrect PED PIN, if one had been set for that PED Key.

As soon as you successfully authenticate, the counter is reset to zero.

View a table that compares and contrasts various "deny access" events or actions that are sometimes confused. See "[Comparison of Destruction/Denial Actions](#)" on page 309.

Other roles and functions that need authentication on the HSM have their own responses to too many bad authentication attempts. Some functions do not keep a count of bad attempts; the simple failure of a multi-step or time-consuming operation is considered sufficient deterrent to a brute-force attack. The table in the next section summarizes the responses.

HSM Response When You Reach the Bad-attempt Threshold

Role	Threshold (number of tries)	Result of too many bad login attempts	Recovery
HSM SO	3	HSM is zeroized (all HSM objects identities, and all partitions are gone)	HSM must be reinitialized. Contents can be restored from backup(s).
Partition SO	3	Partition is zeroized.	Partition must be reinitialized. Contents can be restored from backup.
Audit	10	Lockout	Unlocked automatically after 10 minutes.
Crypto Officer [Note 1]	10 (can be decreased by SO.)	Lockout	Must be unlocked/reset by the partition's SO.
Crypto User [Note 2]	10 (can be decreased by SO)	Lockout	Must be unlocked/reset by the partition's CO.
Domain	n/a	Operation fails	Retry the operation with the correct Domain - usually that would be a backup or restore
Remote PED Key	n/a	Operation fails	Retry establishing a Remote PED connection, providing the correct orange PED Key (PED-authenticated only).
Secure Recovery Key	n/a	Recovery from tamper or Secure Transport Mode fails. Entire HSM is locked. The only operation that is not locked out is establishing a Remote PED connection.	Retry recovery from tamper or from STM, providing the correct purple PED Key (PED-authenticated only).

[Note 1] If the policy "SO can reset PIN" is on, then this user is locked. If "SO can reset PIN" is off, then this user is deleted - as is any user that depends upon it, specifically the Crypto User.

[Note 2] The Crypto User is created by the Crypto Officer. Therefore, only the Crypto Officer, and not the SO of the partition, is able to reset the Crypto User. If the policy "SO can reset PIN" is off, then this user is deleted, rather than

Role	Threshold (number of tries)	Result of too many bad login attempts	Recovery
------	-----------------------------	---------------------------------------	----------

locked out when too many bad attempts are made on the CU. Similarly, if too many bad attempts are made on his creator the Crypto Officer, and that role is deleted, then the associated CU is also deleted.

Control the Outcome

The configurable policy “SO/HSM Admin can reset User PIN” [HSM policy #15] allows you to control the outcome of too many consecutive bad authentication attempts. If the policy is “on” then the outcome is that the HSM Partition is locked out. This means that the Partition and its contents can be accessed again after the HSM Admin resets the HSM Partition Owner’s password. If the policy is “off”, then the partition is zeroized after too many bad attempts – meaning that all contents become inaccessible and the partition must be recreated.

“Ignore failed challenge responses” can be set per partition, which ensures that failed HSM Partition Password attempts do not cause the “failed login attempt” counter to increment.

Resetting Passwords

Resetting is normally done by a higher power when an authentication secret is lost/forgotten, or compromised, and is discussed separately from merely changing authentication when the user is in legitimate possession of the current authentication.

HSM

There is no provision to reset the SO or HSM Admin password (for Password Authenticated HSMs) or the blue PED Key (for PED Authenticated or Trusted Path HSMs), except by initializing the HSM, which destroys [zeroizes] the contents of the HSM and of any HSM Partitions. You can change the password (or the secret on the appropriate blue PED Key) with the `hsm changePw` command, but that requires that you know the current password (or have the current blue PED Key).

The assumption, from a security standpoint, is that if you no longer have the ability to authenticate to the HSM (because you forgot the password or lost the PED Key, or because an unauthorized person has changed the password or PED Key), then the HSM is effectively compromised and must be re-initialized. Thus, no explicit “reset” command is provided.

The `hsm init` command does not require a login, and the `hsm login` command is not accepted if the HSM is in zeroized state.

The following are examples of the behavior of the `hsm login` command in various possible circumstances.

Password Authenticated HSM:

One bad login

With or without `-force` (no difference) / interactive password:

Caution: You have only TWO HSM Admin login attempts left. If you fail two more consecutive login attempts (i.e. with no successful logins in between) the HSM will be ZEROIZED!!!

```
Please enter the HSM Administrators' password:
>
```

With or without `-force` / non-interactive password:

```
>hsm login -password userpin -force
Caution: You have only TWO HSM Admin login attempts left. If
          you fail two more consecutive login attempts (i.e.
          with no successful logins in between) the HSM will
          be ZEROIZED!!!
'hsm login' successful.
```

Two bad logins

Without `-force` / interactive password:

```
Caution: This is your LAST available HSM Admin login attempt.
          If the wrong HSM Admin password is provided the HSM will
          be ZEROIZED!!!
          Type 'proceed' if you are certain you have the
          right login credentials or 'quit' to quit now.
          > proceed
Please enter the HSM Administrators' password:
>
```

Without `-force` / non-interactive password:

```
Caution: This is your LAST available HSM Admin login attempt.
          If the wrong HSM Admin password is provided the HSM will
          be ZEROIZED!!!
          Type 'proceed' if you are certain you have the
          right login credentials or 'quit' to quit now.
          > proceed
'hsm login' successful.
```

With `-force` / interactive password:

```
Caution: This is your LAST available HSM Admin login attempt.
          If the wrong HSM Admin password is provided the HSM will
          be ZEROIZED!!!
Please enter the HSM Administrators' password:
> *****
'hsm login' successful.
```

With `-force` / non-interactive password:

```
Caution: This is your LAST available HSM Admin login attempt.
          If the wrong HSM Admin password is provided the HSM will
          be ZEROIZED!!!
'hsm login' successful.
```

Trusted Path Authentication (uses SafeNet PED and PED Keys):

One bad login

With or without `-force` (no difference):

Caution: You have only TWO HSM Admin login attempts left. If you fail two more consecutive login attempts (i.e. with no successful logins in between) the HSM will be ZEROIZED!!!

Use blue PED key?

Two bad logins

Without `-force` :

Caution: This is your LAST available HSM Admin login attempt. If the wrong blue PED key is provided the HSM will be ZEROIZED!!!
Type 'proceed' if you are certain you have the right login credentials or 'quit' to quit now.
> proceed

Use blue PED key?

With `-force` :

Caution: This is your LAST available HSM Admin login attempt. If the wrong HSM Admin password is provided the HSM will be ZEROIZED!!!

Use blue PED key?

'hsm login' successful.

Example when HSM Zeroized:

Error: The HSM is zeroized due to three consecutive failures to login as HSM Administrator.
'hsm login' is not permitted. The HSM must be re-initialized with the 'hsm init' command.
'hsm login' aborted.

Partition

If you lockout your Partition Owner / Crypto Officer with 10 bad logins AND the "SO can Reset Container PIN" policy is ON, then you MUST reset both the partition owner challenge AND the PED pin:

```
[myLuna] lunash:>partition resetPw -partition Partition1
Which part of the partition password do you wish to change?
 1. change black PED key data
 2. generate new random password for partition owner
 3. generate new random password for crypto-user
 4. both options 1 and 2
 0. abort command
Please select one of the above options:
```

For this situation, you must choose option 4.

If the partition was activated prior to this, you must reactivate it after resetting the PED pin.

If you merely wish to change the Partition password or black PED Key, use the `partition changePw` command instead.

Default Challenge Password

Ordinarily, when a PED-authenticated HSM partition is created, a random challenge-secret is generated and displayed by the PED. That secret becomes the authentication secret for your applications accessing that HSM partition. This is a security feature and is useful for that reason in most situations. It enforces a high-security password at the outset. However, in some scenarios, the imposed hands-on activity of reading a 16-character string from the PED screen and recording it, by hand-writing or typing, might be inappropriate.

PED- authenticated SafeNet HSMs 5.4.x and later allow you to specify a default partition password at "partition create" time.

This feature is useful in three situations:

- It allows you to deploy many partitions automatically.
- It allows fully automated testing for PED-authenticated SafeNet HSMs.
- It allows the use of Crypto Command Center (CCC) to create a High Availability group, which requires all member partitions to share the same password.

The automated testing is important to us, for repeatability and reliability of our testing at various stages of development, validation, and production quality control, but many customers might also wish to perform their own automated testing after receiving purchased SafeNet HSMs, before deploying in their own networks, or after pre-configuring HSMs and partitions for shipment/deployment to their own third-party customers.

Security Effects of Administrative Actions

Actions that you take, in the course of administering your SafeNet HSM, can have effects, including destruction, on the roles, the spaces, and the contents of your HSM and its application partition(s). It is important to be aware of such consequences before taking action.

Overt Security Actions

Some actions in the administration of the HSM, or of an application partition, are explicitly intended to adjust specific security aspects of the HSM or partition. Examples are:

- changing a password
- modifying a policy to make a password or other attribute more stringent than the original setting.

Those are discussed in their own sections.

Actions with Security- and Content-affecting Outcomes

Other administrative events have security repercussions as included effects of the primary action, which could have other intent. Some examples are:

- HSM factory reset
- HSM zeroize
- change of a destructive policy
- installation/application of a destructive Capability Update
- HSM initialization
- application partition initialization

This group of administrative actions is compared in this current section ["Summary of Outcomes of Security-affecting Actions" on the next page](#).

Elsewhere

Certain other actions can sometimes cause collateral changes to the HSM, like firmware rollback and update. They usually do not affect contents, unless a partition is full and the action changes the size of partitions or changes the amount of space-per-partition that is taken by overhead/infrastructure, such as when going to HSM firmware 6.22.0 from an earlier version. These are discussed elsewhere.

Summary of Outcomes of Security-affecting Actions

This table lists some major administrative actions that can be performed on the HSM, and compares relevant security-related effects. Use the information in this table to help decide if your contemplated action is appropriate in current circumstances, or if additional preparation (such as backup of partition content, collection of audit data) would be prudent before continuing.

Factory Reset HSM With Firmware <6.22.0

Domain	Destroyed
HSM SO Role	Destroyed
Partition SO Role	Destroyed
Auditor Role	Destroyed
Partition Roles	Destroyed
HSM or Partition/Contents	HSM/Destroyed
HSM Policies	Unchanged
RPV	Unchanged
Messaging	You are about to factory reset the HSM. All contents of the HSM will be destroyed. HSM policies and remote PED vector left unchanged.

Factory Reset HSM With Firmware ≥6.22.0

Domain	Destroyed
HSM SO Role	Destroyed
Partition SO Role	Destroyed
Auditor Role	Destroyed
Partition Roles	Destroyed
HSM or Partition/Contents	HSM/Destroyed
HSM Policies	Reset
RPV	
Messaging	You are about to factory reset the HSM. All contents of the HSM will be destroyed. HSM policies will be reset and the remote PED vector will be erased.

Zeroize HSM With Firmware ≥6.22.0

Domain	Destroyed
HSM SO Role	Destroyed
Partition SO Role	Destroyed
Auditor Role	Unchanged
Partition Roles	Destroyed
HSM or Partition/Contents	HSM/Destroyed
HSM Policies	Unchanged
RPV	Unchanged
Messaging	You are about to zeroize the HSM. All contents of the HSM will be destroyed. HSM policies, remote PED vector and Auditor left unchanged.

Change Destructive HSM Policy

Domain	Unchanged
HSM SO Role	Unchanged
Partition SO Role	Destroyed
Auditor Role	Unchanged
Partition Roles	Destroyed
HSM or Partition/Contents	HSM/Destroyed
HSM Policies	Unchanged except for new policy
RPV	Unchanged
Messaging	You are about to change a destructive HSM policy. All partitions of the HSM will be destroyed.

Apply Destructive CUF Update

Domain	Destroyed
HSM SO Role	Destroyed
Partition SO Role	Destroyed
Auditor Role	Unchanged
Partition Roles	Destroyed

HSM or Partition/Contents	HSM/Destroyed
HSM Policies	Unchanged
RPV	Unchanged
Messaging	You are about to apply a destructive update. All contents of the HSM will be destroyed.

HSM Initialize When Admin Not Initialized

Domain	Destroyed
HSM SO Role	Destroyed
Partition SO Role	Destroyed
Auditor Role	Unchanged
Partition Roles	Destroyed
HSM or Partition/Contents	HSM/Destroyed
HSM Policies	Unchanged
RPV	Unchanged
Messaging	You are about to initialize the HSM. All contents of the HSM will be destroyed.

HSM Initialize When Admin Initialized

Domain	Unchanged
HSM SO Role	Unchanged
Partition SO Role	Destroyed
Auditor Role	Unchanged
Partition Roles	Destroyed
HSM or Partition/Contents	HSM/Destroyed
HSM Policies	Unchanged
RPV	Unchanged
Messaging	You are about to initialize the HSM that is already initialized. All partitions of the HSM will be destroyed. You are required to provide the current SO password.

Non-Admin Partition Initialize When the Partition is Not Initialized

Domain	Unchanged
HSM SO Role	Unchanged
Partition SO Role	Destroyed
Auditor Role	Unchanged
Partition Roles	Destroyed
HSM or Partition/Contents	Partition/Destroyed
HSM Policies	Unchanged
RPV	Unchanged
Messaging	You are about to initialize the partition. All contents of the partition will be destroyed.

Non-Admin Partition Initialize When the Partition is Initialized

Domain	Unchanged
HSM SO Role	Unchanged
Partition SO Role	Destroyed
Auditor Role	Unchanged
Partition Roles	Destroyed
HSM or Partition/Contents	Partition/Destroyed
HSM Policies	Unchanged
RPV	Unchanged
Messaging	You are about to initialize the partition that is already initialized. All contents of the partition will be destroyed. You are required to provide the current Partition SO password.

Secure Transport Mode

This chapter describes Secure Transport Mode, the Master Tamper Key (MTK) and Secure Recovery Key (SRK), and the purple PED key. It contains the following sections:

- ["MTK and SRK" below](#)
- ["Secure Transport Mode \[Local\]" on page 331](#)
- ["Re-Split Required" on page 334](#)
- ["Interrupted SRK Re-split Operation" on page 1](#)

MTK and SRK

Every SafeNet HSM has a Master Tamper Key (MTK) that strongly encrypts all sensitive data generated and stored within the HSM. While the master tamper key remains valid, the HSM uses it to decrypt HSM contents in order to perform cryptographic operations. The master tamper key is unique for each HSM.

When the master tamper key is created, two splits of that secret are stored in separate locations within the HSM.

A tamper event erases the HSM's master tamper key, making the HSM unusable and its contents inaccessible. That is, all contents remain encrypted and the HSM cannot use the master tamper key to decrypt them while the master tamper key does not exist.

The master tamper key can be reconstituted from the two splits for resumption of HSM operation.

The scenarios are:

- a) the two splits remain inside the HSM, and the HSM can use them to immediately recover the master key
- b) one of the splits remains inside the HSM, but the other is moved out to an external device; the HSM cannot recover the HSM's master key until the external split is re-introduced into the HSM via the PED.



Note: The MTK is NOT, in any sense, any kind of "device master key". Its purpose is to implement the tamper behavior and the related Secure Transport Mode. If someone managed to get the MTK for an HSM, they have nothing really usable. All HSM contents are further encrypted by additional layers of authentication and other strong encryption, as described elsewhere in these documents.

Tamper and Recover with Purple Key NOT Enabled

If a tamper event occurs, the event is logged and the HSM stops responding. A restart is required in order to resume. The master key is reconstituted from its component splits and re-validated, making the HSM usable again. The event is recorded in the log.

The intent is to make you aware that a tamper has occurred (the log gets an entry, and the HSM waits for a restart) but not to cause ongoing inconvenience.

That scenario applies to Password-authenticated HSMs, as well as to PED-authenticated HSMs that do not have the purple SRK option invoked - both splits of the master key remain inside the HSM.

```
[myluna] lunash:>hsm srk show
Secure Recovery State flags:

=====
External split enabled: no
SRK resplit required:  no
Hardware tampered:     no
Transport mode:        no
Command Result : 0 (Success)
[myluna] lunash:>
```

Tamper and Recover with Purple PED Key Enabled

Some operational environments require more rigorous response to a tamper event. Your operational and security policy might require that, in addition to logging a tamper event, the HSM must stop performing until you take authoritative action to confirm your acknowledgement and clear the tamper before operations can resume.

PED-authenticated HSMs have the option to store a split of the master key, called the Secure Recovery Vector (SRV) outside of the HSM on a physical device, the Secure Recovery Key (SRK). The situation is as above:

- master key encrypts everything,
- HSM can operate on its contents while the master key is valid,
- master key splits exist,
- master key can be reconstituted and re-validated],

But one split of the master key is imprinted onto a purple PED Key (or Keys, if you choose MofN), the SRK, and not inside the HSM. In that case, the master key component stored inside the HSM is insufficient to reconstitute the master key without the portion stored externally on the purple key(s).

```
[myluna] lunash:>hsm srk show
Secure Recovery State flags:

=====
External split enabled: yes
SRK resplit required:  no
Hardware tampered:     no
Transport mode:        no
Command Result : 0 (Success)
[myluna] lunash:>
```

Where the purple PED Key has been invoked, a tamper event invalidates the master key, but the HSM cannot immediately re-validate because not enough splits are available. You must invoke the LunaSH command `hsm srk transportMode recover`, and present the purple PED Key (or Keys) when prompted. The PED reads the purple key(s) and provides the missing piece (the SRV) to the HSM. The HSM combines the provided external component with the internal component and reconstitutes and validates the master key. You can resume using the HSM with no loss of crypto objects.

Behavior with Purple PED Key Enabled but MISSING or DAMAGED

If SRK is enabled (meaning a split of the MTK has been moved off of the HSM, onto a purple PED Key), and you have lost or destroyed the purple PED Key, then you **cannot** bring the HSM back from tamper or Secure Transport Mode.

Your only option is to return the HSM to SafeNet for re-manufacturing.

However, you CAN run `hsm factoryReset` (from a local console connection), if your security policies require active destruction of HSM objects before it leaves your possession.

When the HSM is returned to you, after re-manufacture, it has a new MTK as if it was a new HSM, so even a copy of the old purple PED Key would be of no use to an unauthorized person. You can freshly initialize the re-manufactured HSM using your existing Domain PED Key, so that you can recover objects from one of your backups onto the re-manufactured HSM.

Secure Transport Mode

If your practice is to configure and prepare your HSMs at a central location before shipping them out to remote locations in your organization, or to configure and prepare your HSMs at a central location before shipping them out to your customers then you can invoke an enhanced security option for shipping.

By switching on Secure Transport Mode (STM), you effectively create a controlled tamper event. The HSM can be shipped in a condition that is completely unusable by anyone who might intercept it if they are not in possession of the unique SRK for that HSM. At the destination, the purple PED Key is used to reconstitute the Secure Recovery Vector, and the MTK is recovered. Your recipient begins using the HSM, with your loaded keys and certificates intact, secure in the knowledge that the HSM has not been attacked in transit.

```
[myluna] lunash:>hsm srk show
      Secure Recovery State flags:

      =====
      External split enabled:  yes
      SRK resplit required:   no
      Hardware tampered:      no
      Transport mode:         yes
Command Result : 0 (Success)
[myluna] lunash:>
```

As a service for customers who request it, SafeNet can invoke Secure Transport Mode at our factory. Your HSM is shipped in anti-tamper packaging via a trusted courier. Meanwhile, the unique purple key is shipped via a separate channel, so that the two are never available together in transit. As a further precaution, we send you the verification code via yet another path. In this way, you enjoy the maximum possible assurance that your HSM has not been attacked between our facility and yours.

Similarly, if your organization ships HSMs to your customers, you can offer them the same service, with new unique purple (SRK) keys generated at each stage, if desired.

Make a New Purple PED Key (SRK external split)

In the event that a SafeNet HSM appliance has been shipped to you, in Secure Transport Mode, and you have recovered from Transport Mode, OR you have decided to invoke Transport Mode to store or ship your HSM, you might wish to generate a unique, new SRK split onto a purple PED Key. That is, the existing MTK is split in two again (different splits than previous) and one new split replaces the internally stored component while the other new split is stored on a new purple PED Key (or keys).

Use the LunaSH command `hsm srk keys resplit` and have a different purple key ready for imprinting.

The existing SRK is needed to validate the operation, but the HSM will not overwrite that purple key with the new split. This is a safety measure, in case the operation was interrupted before completion. If that happens, the old SRK remains valid and the HSM can still be used while you begin the re-split operation again. Once the re-split is

successful, with a new SRK on a new purple key, the old purple key is of no further interest because its contents have been superseded [It could be used in a future re-split operation for this HSM, or it could be imprinted from another HSM, if desired].

For example, when we (SafeNet) ship an appliance in Secure Transport Mode, we make just the one purple key that we ship to you, unless requested otherwise by you. We think that we are trustworthy, but your security auditors might be required by policy or regulation to assume otherwise. For this reason, you can perform `hsm srk keys resplit` as soon as you receive your HSM, and it would not matter if we had kept additional copies of your purple key. They would be rendered useless by the re-splitting operation on your HSM.

Similarly, your customers can do the same once they receive an HSM in Secure Transport Mode from you.

If your application requires that you ship securely, but retain control of the HSM appliances, you can use the Remote PED option to retain all PED Keys (including the purple) and use them from your location when operating the remotely located HSM.

Master Key must be present

You can perform a re-splitting operation only if the HSM's master key is currently valid. That is, for security reasons, you cannot generate a new split and new purple key while an HSM is tampered and not recovered, or while an HSM is in Secure Transport Mode and not recovered. This prevents anyone not in possession of the correct purple PED Key from circumventing the HSM "lock-out" due to tamper or to STM.

What if the purple SRK has been lost?

If the purple SRK is lost, and you do not have a backup of it, then you must obtain an RMA and return the HSM to SafeNet for re-manufacture; any objects that existed on the HSM are lost, unless you have backups. For SafeNet Network HSM, some customers might prefer the additional assurance of pressing the red factory reset button on the back of the appliance, before shipping. For SafeNet PCIe HSM, the equivalent action is shorting/shunting the two pins of the reset header (Tamper 2) on the HSM card, near the battery.

Disabling SRK

If SRK has been enabled (one of the HSM's master key splits has been moved to an external purple PED Key), you can choose to disable the feature.

To do so, you must provide the appropriate purple SRK(s) when you run the command `hsm srk disable`, so that the external split can be read back into a secure internal location.

From that point on, the HSM behaves in the same way as a password-authenticated HSM with respect to tamper events:

- a real tamper event is merely noted in the log and does not hamper HSM operation (beyond requiring restart and re-login)
- Secure Transport Mode is not possible (until SRK is re-enabled).

Compare and Contrast Some "Denial" and Destructive Scenarios

View a table that compares and contrasts various "deny access" events or actions that are sometimes confused - ["Comparison of Destruction/Denial Actions" on page 309.](#)

Secure Transport Mode [Local]

This topic describes what to do if you wish to invoke Secure Transport Mode (STM) on a local SafeNet Network HSM, when shipping the appliance:

- to your customer or
- to your partner organization or
- to your own personnel at another site within your organization.

This page applies to PED Authenticated HSMs only. It does not apply to Password Authenticated HSMs.

As well, you could use STM for securely storing the HSM, where "transport" would take place simply into, and later out of, your warehouse or vault. However, you would also need to manage separate secure storage and handling of the imprinted purple PED Key (SRK) for that HSM until it was time to recover the HSM and return it to service.

Backup

Perform backups of your partitions before you continue with Secure Transport Mode procedure (see "Backup and Restore HSMs and Partitions" on page 1).

First time - no existing SRK (purple key)

This section describes the procedure if you are performing all the actions locally - that is, if your SafeNet PED is connected directly to the SafeNet Network HSM appliance when you invoke STM.

This procedure performs a new split of the HSM's master key and creates a new external split (SRV) to be imprinted on a new purple PED Key. This ensures that nobody could have a pre-existing copy. If you do wish to re-use an existing SRK, do not disable SRK - [re-]enabling SRK includes a re-split of the HSM's Master key, followed by imprinting the new external split on a purple PED Key - any pre-existing SRK becomes invalid.

Have available:

- your SafeNet appliance
- the SafeNet PED and cable
- two "new" purple keys [either they are new and blank from the factory, or they have been used for some other purpose (could include previous use as purple keys for another HSM) that do not contain a currently valid SRV for this HSM - we suggest that you apply purple labels before you start]

Perform the SRK enable operation, and enter Secure Transport Mode now:

1. Connect the SafeNet PED to the SafeNet Network HSM appliance.
2. Login to the appliance as 'admin'.
3. Login to the HSM

```
hsm login
```

4. Run the command

```
hsm srk enable
```

Follow the PED prompts, introducing the purple key (*) and pressing buttons on the PED keypad.

Record the SRV verification string when it is presented.

This command performs a resplit of the MTK before moving one of the (new) splits out to your purple PED Key (s).

5. Run the command

```
hsm srk transportMode enter
```

Follow the PED prompts, presenting the purple key from step 4 (above) when prompted.

6. When the HSM enters Secure Transport Mode, shut it down with


```
sysconf appliance poweroff
```
7. Disconnect the appliance and pack it for shipment.
8. Ship the imprinted purple SRK to your customer or remote location, separately from the appliance shipment.
9. Send the STM verification string via another path.

(* For simplicity, this procedure assumes that you choose Mvalue and Nvalue as "1" in order to have a single imprinted purple key. You can choose other values for M and N, to split the SRV across multiple purple PED Keys. Similarly, you are also given the opportunity to make a copy of the purple SRK - or of the MofN group of purple PED Keys if you elected to use MofN. Your choices in these matters are dictated by convenience and by your security policies.)

Recovery

At the new location, unpack the appliance and connect it to power, network, and PED.

You should also have available the PED Keys for that HSM, including the purple PED Key (*) that you received by separate shipment, as well as the verification string.

10. Power up the appliance and log in.
11. Compare the verification string that was shipped with the string that is presented when you run the command


```
hsm srk keys verify
```

 The PED prompts for the SRK (purple PED Key) and shows the verification string. The displayed string should match the string that was sent by mail or courier.
12. Run the command


```
hsm srk transportMode recover
```

 Respond to the PED prompts (*).

You can now use the HSM's blue PED Key to log in and administer the HSM, and black PED Keys to activate partitions for clients to access.

(*If you invoked MofN when creating the SRK, you must provide quantity M of the purple keys containing the SRV splits.)

No Re-split?

This section describes the same procedure if SRK was previously enabled, and you wish to *re-use* the existing purple PED Key. This might be the case if you know that the most recent re-split of the master key was performed by your organization.

Have available:

- your SafeNet appliance
- the SafeNet PED and cable
- the existing, imprinted purple PED Key[s]
- the verification string for the existing purple key [because you are not re-splitting or enabling at this time, the verification string for the Secure Recovery Vector on the existing key is not presented, so you must already have it in your possession]

Perform the SRK enable operation, and enter Secure Transport Mode now:

1. Connect the SafeNet PED to the SafeNet Network HSM appliance.
2. Login to the appliance as 'admin'.
3. Run the command


```
hsm srk transportMode enter
```

 Follow the PED prompts, presenting the current purple PED Key (or presenting quantity M of the purple PED Keys, if you had invoked MofN during the most recent enable or resplit) when prompted.
4. When the HSM enters Secure Transport Mode, shut it down with


```
sysconf appliance poweroff
```
5. Disconnect the appliance and pack it for shipment.
6. Ship the imprinted purple SRK to your customer or remote location, separately from the appliance shipment.
7. Send the STM verification string via another path.

(* For simplicity, this procedure assumes that you choose Mvalue and Nvalue as "1" in order to have a single imprinted purple key. You can choose other values for M and N, to split the SRV across multiple purple PED Keys. Similarly, you are also given the opportunity to make a copy of the purple SRK - or of the MofN group of purple PED Keys if you elected to use MofN. Your choices in these matters are dictated by convenience and by your security policies.)

Additional Notes

Re-split?

You do not need to re-split if you have recently created a new purple PED Key for this HSM (as happens when you `hsm srk enable`), or if you have maintained rigorous records and are confident that your purple PED Key has not been compromised.

To be absolutely sure, we recommend that you perform a re-split before placing the HSM in Secure Transport Mode, unless your procedures require that you use an existing purple PED Key.

Number of purple keys

The basic procedures above suggest that you need either one or two purple PED Keys.

However, choices that you can make while interacting with the SafeNet PED could require several additional purple keys.

Copies?

During every imprinting transaction, the PED asks if you wish to copy the inserted PED Key. Therefore, ensure that you have enough spare/blank keys ready for the number of copies that you expect to make (examples: alternate key holder, on-site stored back-up, off-site back-up/escrow).

MofN?

At the beginning of an imprinting operation, the PED asks for values of "M" and "N". This is for "MofN" split-secret multi-user (or multi-part) authentication, explained elsewhere in this manual (see ["Using MofN" on page 243](#)). If you input "1" for both "M" and "N", then MofN is not in operation, and only the single (in this case purple) PED Key is needed to carry the authentication secret. But if you specify numbers higher than "1", then the PED splits the secret and imprints the splits onto quantity N different keys. Later, any operation that calls for that secret will need quantity M of those splits to be re-united (presented to the PED upon demand). Thus you must have quantity "N" spare/blank keys ready if you intend to invoke MofN.

Copies and MofN?

Furthermore, you might wish to combine the two scenarios above - in most situations it is prudent to have a backup of any authentication device, against loss or damage. If that is your policy, and if you intend to invoke MofN, then you will need enough spare/blank keys to make at least two full MofN sets of the SRK (purple PED Key(s)). In that case, be careful not to mix the copy sets. If you chose MofN as 3 of 5, then the 5 keys get 5 different splits of the purple-key secret. When you later need to present the secret, you need 3 different splits. If the sets are accidentally mixed, you could have three key holders standing at the PED, but with two of their keys being duplicate splits - the PED refuses to accept what it sees as the same key twice when recreating an MofN secret. To avoid the possibility of inadvertently mixing copies and originals, one option is to simply choose a small "M" and a large "N", and not make copies. For example, if you chose M=3 and N=15, you could arbitrarily select three groups of five splits and use them as though the split was "3 of 5". That is, you could distribute 5 of the 15 splits to operations personnel, so that any three of them could authenticate to the HSM. You could keep another 5 of the 15 splits in on-site lockup as on-site backup, and you could keep the third group of 5 of the 15 splits as your off-site backup. In that case, you avoid the possibility of accidentally mixing copied smaller groups, which could result in two or three of the same split in one group - which the PED would reject.

View a table that compares and contrasts various "deny access" events or actions that are sometimes confused (["Comparison of Destruction/Denial Actions" on page 309](#)).

Re-Split Required

The "SRK resplit required" flag is set only in the event that a failure occurred during a re-split operation, leaving the HSM in an intermediate state. An example might be the user pressing cancel at the wrong time, or a power failure or disconnection during a re-split.

```
Secure Recovery State flags:
=====
External split enabled:  yes
SRK resplit required:  yes
Hardware tampered:     no
Transport mode:        no
```

After an incomplete hsm srk resplit, an attempt to login or to perform other HSM operations would yield an error message about the MTK state. The HSM would process only view/show commands while in that state.

In that situation, it is operationally urgent to issue the command:

```
hsm srk keys resplit
```

which creates a new split of the SRK and places the external portion on a new purple PED Key (or keys, if you choose to invoke MofN).

The HSM is once more usable.

Security

An attacker lacking the proper purple key cannot place the HSM into "SRK resplit required" state. Only the holder of the legitimate purple PED Key can start an hsm srk resplit operation, and is therefore entitled to resume/restart that operation if it is interrupted.

Slot Numbering and Behavior

Administrative partitions and application partitions are identified as PKCS#11 cryptographic slots in SafeNet utilities, such as LunaCM and multitoken, and for applications that use the SafeNet library.

Order of Occurrence for Different SafeNet HSMs

A host computer with SafeNet HSM Client software and SafeNet libraries installed can have SafeNet HSMs connected in any of three ways:

- PCIe embedded/inserted SafeNet PCIe HSM card (one or multiple HSMs installed - administrative partitions and application partitions are shown separately if HSM firmware is version 6.22.0 or newer)
- USB-connected SafeNet USB HSMs (one or multiple - administrative partitions and application partitions are shown separately if HSM firmware is version 6.22.0 or newer)
- SafeNet Network HSM application partitions(*), registered and connected via NTLS or via STC.

Any connected HSM partitions are shown as numbered slots. Slots are numbered from zero or from one, depending on configuration settings (see "[Settings Affecting Slot Order](#)" on the next page, below), and on the firmware version of the HSM(s).

(*One or multiple application partitions. Administrative partitions on SafeNet Enterprise HSMs are not visible via lunacm and other client-side tools. Only registered, connected application partitions are visible, of which multiple-per-HSM, up to 100, can exist. That is, a remote SafeNet Network HSM might support 100 application partitions, but your application and lunacm might see only one or two or fifteen of them if those were the only ones that had established certificate-exchange NTLS links with the current Client computer.)

In lunacm, a slot list would normally show:

- SafeNet Network HSM application partitions for which NTLS links are established with the current host, followed by
- SafeNet PCIe HSM cards, followed by
- SafeNet USB HSMs

For SafeNet Network HSM, as seen from a client (via NTLS), only application partitions are visible. The HSM administrative partition of a remote SafeNet Network HSM is never seen by a SafeNet HSM Client. The SafeNet Network HSM slots are listed in the order they are polled, dictated by the entries in the [SafeNet Network HSM] section of the Crystoki.ini / chrystoki.conf file, like this:

```
ServerName00=192.20.17.200
ServerPort00=1792
ServerHt100=0
ServerName01=192.20.17.220
ServerPort01=1793
ServerHt101=
```

For SafeNet PCIe HSM and SafeNet USB HSM, if you have multiple of either HSM type connected on a single host, then the order in which they appear is the hardware slot number, as discovered by the host computer.

For SafeNet PCIe HSM and SafeNet USB HSM, the HSM administrative slot always appears immediately after the application partition. If no application partition has yet been created, a space is reserved for it, in the slot numbering.

Settings Affecting Slot Order

Settings in the [Presentation] section of the configuration file (Chrystoki.conf for UNIX/Linux, crystoki.ini for Windows) can affect the numbering that the API presents to SafeNet tools (like lunacm) or to your application.

[Presentation]

ShowUserSlots=<slot><serialnumber>

- Sets starting slot for the identified partition.
- Default, when ShowUserSlots is not specified, is that all available partitions are visible and appear in default order.
- Can be applied, individually, to multiple partitions, by a single entry containing a comma-separated list like: ShowUserSlots=1(351970018022),2(351970018021),3(351970018020),....
- Affects only PPSO partitions (f/w 6.22.0 or newer)
- If multiple partitions on the same HSM are connected to the SafeNet HSM Client host computer, redirecting one of those partitions with ShowUserSlots= causes all the others to disappear from the slot list, unless they are also explicitly re-ordered by the same configuration setting.

ShowAdminTokens=yes

- Default is yes. Admin partitions of local HSMs are visible in a slot listing.
- Remotely connected partitions (SafeNet Network HSM) are not affected by this setting, because NTLS connects only application partitions, not HSM SO (Admin) partitions to clients, so a SafeNet Network HSM SO administrative partition would never be visible in a client-side slot list, regardless.

ShowEmptySlots=1

- Controls how C_GetSlotList - as used by lunacm slot list command, or ckdemo command 14, and by your PKCS#11 application - displays, or does not display unused potential slots, when the number of partitions on an HSM is not at the limit.

OneBaseSlotId=1

- Causes basic slot list to start at slot number 1 (one) instead of default 0 (zero). (Any submitted number other than zero is treated as "1". Any letter or other non-numeric character is treated as "0".)

Effects of Settings on Slot List

Say, for example, you have multiple HSMs connected to your host computer (or installed inside), with any combination of firmware 6.22.0 (and newer) or pre-6.22.0 firmware, and no explicit entries exist for slot order in the config file. The defaults prevail and the slot list would start at zero.

If you set OneBaseSlotId=1 in the configuration file, then the slot list starts at "1" instead of at "0". You could set this for personal preference, or according to how your application might expect slot numbering to occur (or if you have

existing scripted solutions that depend on slot numbering starting at zero or starting at one). OneBaseSlotId affects the starting number for all slots, regardless of firmware.

If you set ShowUserSlots=20(17923506), then the identified token or HSM or application partition would appear at slot 20, regardless of the locations of other HSMs and partitions, but only if the indicated partition is firmware 6.22.0 or newer and is a PPSO partition.

Effects of New Firmware on Slot Login State

Note: Slots retain login state when current-slot focus changes.



For HSMs with firmware earlier than version 6.22.0, when you used **slot set** to move the focus from an HSM partition or slot with logged in session(s), to another partition or slot, any sessions on the original slot were automatically closed (thus logged out).

For HSMs with firmware version 6.22.0 or newer, you can use **slot set** to repeatedly shift focus among slots, and whatever login state was in force when you were previously focused on a slot is still in effect when you return to that slot.

SNMP Monitoring

This chapter describes Simple Network Management Protocol (SNMP v3) support for remote monitoring of conditions on a local HSM that might require administrative attention. It contains the following sections:

- ["Overview and Installation" below](#)
- ["The SafeNet Chrysalis-UTSP MIB" on page 340](#)
- ["The SAFENET HSM MIB" on page 341](#)
- ["Frequently Asked Questions" on page 349](#)

Overview and Installation

This section provides an overview of the SNMP implementation and describes how to install the SNMP subagent.



Note: This feature is not currently supported for use with IPv6 networks.

MIB

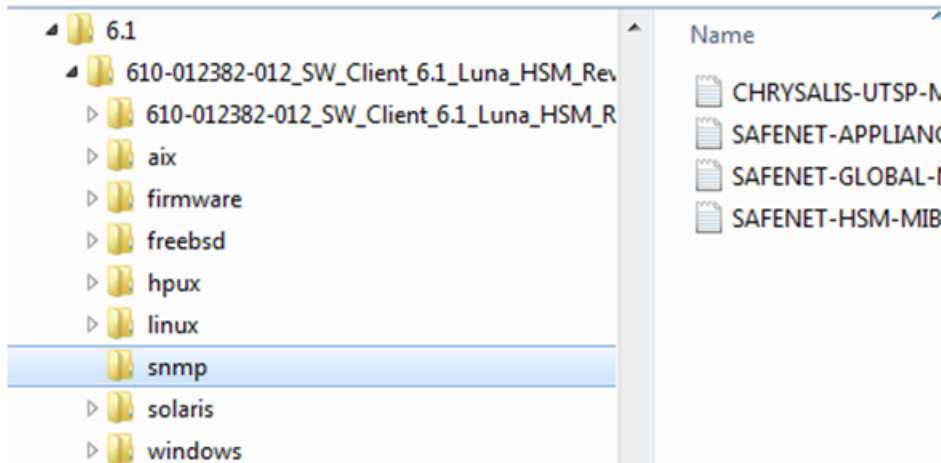
We provide the following MIBs (management information base):

MIB Name	Description
CHRYSALIS-UTSP-MIB.txt	Defines SNMP access to information about the SafeNet appliance.
SAFENET-HSM-MIB.txt	Defines SNMP access to information about the SafeNet HSM.
SAFENET-GLOBAL-MIB.txt	Must be found in your system path so that symbols can be resolved.
SAFENET-APPLIANCE-MIB.txt	Reports the software version of SafeNet Network HSM appliance.

Copy all MIBs in <luna client install dir > to the MIB directory on your system.

For SafeNet Network HSM, the host is the appliance, so all the above MIBs are in the appliance, to support SNMP.

In case of later need, the LunaClient installation medium contains all the supplied MIB files in an SNMP folder, like this example for Windows:



SafeNet SNMP Subagent

We find that most customers choosing to use SNMP already have an SNMP infrastructure in place. Therefore, we provide a subagent that you can install on your managed workstations, and which can point to your agent via the socket created by the agent. This applies to SafeNet USB HSM and SafeNet PCIe HSM - for SafeNet Network HSM, the subagent is already on the appliance.

The SNMP subagent (luna-snmp) is an AgentX SNMP module that extends an existing SNMP agent with support for SafeNet HSM monitoring. It is an optional component of the SafeNet HSM client installation. The subagent has been tested against net-snmp, but should work with any SNMP agent that supports the AgentX protocol.

To install the SNMP subagent

After selecting one or more products from the main SafeNet HSM Client installation menu, you are presented with a list of optional components, including the SNMP subagent. It is not selected by default, but can be installed with any product except the SafeNet Network HSM client installed in isolation.

1. In the installation media, go to the appropriate folder for your operating system.
2. Run the installer (install.sh for Linux and UNIX, LunaClient.msi for Windows).
3. Choose the SafeNet products that you wish to install, and include SNMP among your selections. The subagent is installed for any SafeNet product except SafeNet Network HSM in isolation.
4. Proceed to Post-installation configuration.

Post-installation configuration

After the SafeNet HSM client is installed, complete the following steps to configure the SNMP subagent:

1. Copy the SafeNet MIBs from <install dir>/snmp to the main SNMP agent's MIB directory. Or copy to another computer (your SNMP computer) if you are not running SNMP from the same computer where SafeNet Client software is installed.
2. If running on Windows, configure the subagent via the file <install dir>/snmp/luna-snmp.conf to point to the AgentX port where the main SNMP agent is listening. The file must then be copied to the same directory as snmpd.conf. (This assumes net-snmp is installed; the setup might differ if you have another agent.)

If running on a UNIX-based platform, the subagent should work without extra configuration assuming that the primary SNMP agent is listening on the default local socket (/var/agentx/master). You still have the option of editing and using luna-snmp.conf.

- After configuration is complete, start the agent. Then start the subagent via the service tool applicable to your platform (for example, “service luna-snmp start” on Linux, or start SafeNet SNMP Subagent Service from the services in Windows).

Normally the agent is started first. However, the subagent periodically attempts to connect to the agent until it is successful. The defaults controlling this behavior are listed below. They can be overridden by changing the appropriate entries in **luna-snmp.conf**.

Configuration Options In the luna-snmp.conf File

Option	Description	Default
agentXSocket [<transport-specifier>:]<transport-address>[,...]	Defines the address to which the subagent should connect. The default on UNIX-based systems is the Unix Domain socket "/var/agentx/master". Another common alternative is tcp:localhost:705. See the section LISTENING ADDRESSES in the snmpd man page for more information about the format of addresses (http://www.net-snmp.org/docs/man/snmpd.html).	The default, for Linux, is "/var/agentx/master". In the file, you can choose to un-comment "tcp:localhost:705" which is most commonly used with Windows.
agentXPingInterval <NUM>	Makes the subagent try to reconnect every <NUM> seconds to the master if it ever becomes (or starts) disconnected.	15
agentXTimeout <NUM>	Defines the timeout period (NUM seconds) for an AgentX request.	1
agentXRetries <NUM>	Defines the number of retries for an AgentX request.	5

The SafeNet Chrysalis-UTSP MIB



Note: The Chrysalis MIB is the SafeNet MIB for all SafeNet HSM products - the Chrysalis name is retained for historical continuity.

To illustrate accessing data, the command "snmpwalk -v 3 -u admin -l authPriv -a SHA1 -A 12345678 -x AES -X 87654321 myLuna19 private" produced this output:

- CHRYSALIS-UTSP-MIB::hsmOperationRequests.0 = Counter64: 3858380
- CHRYSALIS-UTSP-MIB::hsmOperationErrors.0 = Counter64: 385838
- CHRYSALIS-UTSP-MIB::hsmCriticalEvents.0 = Counter64: 0
- CHRYSALIS-UTSP-MIB::hsmNonCriticalEvents.0 = Counter64: 5
- CHRYSALIS-UTSP-MIB::ntlsOperStatus.0 = INTEGER: up(1)
- CHRYSALIS-UTSP-MIB::ntlsConnectedClients.0 = Gauge32: 0
- CHRYSALIS-UTSP-MIB::ntlsLinks.0 = Gauge32: 0
- CHRYSALIS-UTSP-MIB::ntlsSuccessfulClientConnections.0 = Counter64: 16571615927115620

- CHRYSALIS-UTSP-MIB::ntlsFailedClientConnections.0 = Counter64: 1657161592711562

The various counts are recorded since the last restart.

Item	Description
hsmOperationRequests	The total number of HSM operations that have been requested.
hsmOperationErrors	The total number of HSM operations that have been requested, that have resulted in errors.
hsmCriticalEvents	The total number of critical HSM events that have been detected (Tamper, Zeroization, SO creation, or Audit role creation)
hsmNonCriticalEvents	The total number of NON-critical HSM events that have been detected (any that are not among the critical list, above).
ntlsOperStatus	The current operational status of the NTL service, where the options are: 1 = up, 2 = not running, and 3 = status cannot be determined.
ntlsConnectedClients	The current number of connected clients using NTLS.
ntlsLinks	The current number of links in NTLS - can be multiple per client, depending on processes.
ntlsSuccessfulClientConnections	The total number of successful client connections.
ntlsFailedClientConnections	The total number of UNsuccessful client connections.

The SAFENET HSM MIB

The SAFENET-HSM-MIB defines HSM status information and HSM Partition information that can be viewed via SNMP.

To access tables, use a command like:

```
snmptable -a SHA -A snmppass -u snmpuser -x AES -X snmppass -l authPriv -v 3 192.20.11.59 SAFENET-HSM-MIB::hsmTable
```

The information is defined in tables, as detailed in the following sections:

SNMP Table Updates

The SNMP tables are updated and cached every 60 seconds. Any changes made on the HSM may therefore take up to 60 seconds to be included in the tables. When a query is received to view the tables, the most recent cached version is displayed. If a change you were expecting is not displayed, wait 60 seconds and try again.



Note: Some values may not get updated automatically, such as the HSM firmware version (hsmFirmwareVersion) following a firmware upgrade. To force an update, restart the SNMP agent.

hsmTable

This table provides a list of all the HSM information on the managed element.

Item	Type	Description	Values
hsmSerialNumber	DisplayString	Serial number of the HSM - used as an index into the tables.	From factory
hsmFirmwareVersion	DisplayString	Version of firmware executing on the HSM.	As found
hsmLabel	DisplayString	Label associated with the HSM.	Provided by SO at init time
hsmModel	DisplayString	Model identifier for the HSM.	From factory
hsmAuthenticationMethod	INTEGER	Authentication mode of the HSM.	unknown(1), -- not known password(2), -- requires passwords pedKeys(3) -- requires PED
hsmRpvInitialized	INTEGER	Remote ped vector initialized flag of the HSM.	notSupported(1), -- rpv not supported uninitialized(2), -- rpv not initialized initialized(3) -- rpv initialized
hsmFipsMode	TruthValue	FIPS 140-2 operation mode enabled flag of the HSM.	Factory set
hsmPerformance	INTEGER	Performance level of the HSM.	
hsmStorageTotalBytes	Unsigned32	Total storage capacity in bytes of the HSM	Factory set
hsmStorageAllocatedBytes	Unsigned32	Number of allocated bytes on the HSM	Calculated
hsmStorageAvailableBytes	Unsigned32	Number of available bytes on the HSM	Calculated
hsmMaximumPartitions	Unsigned32	Maximum number of partitions allowed on the HSM	2, 5, 10, 15, or 20, per license
hsmPartitionsCreated	Unsigned32	Number of partitions created on the HSM	As found
hsmPartitionsFree	Unsigned32	Number of partitions that can still be created on the HSM	Calculated
hsmBackupProtocol	INTEGER	Backup protocol used on the HSM	unknown(1), none(2),

Item	Type	Description	Values
			cloning(3), keyExport(4)
hsmAdminLoginAttempts	Counter32	Number of failed Administrator login attempts left before HSM zeroized	As found, calculated
hsmAuditRoleInitialized	INTEGER	Audit role is initialized flag	notSupported (0), yes(1), no(2)
hsmManuallyZeroized	TruthValue	Was HSM manually zeroized flag	As found
hsmUpTime	Counter64	Up time in seconds since last HSM reset	Counted
hsmBusyTime	Counter64	Busy time in seconds since the last HSM reset	Calculated
hsmCommandCount	Counter64	HSM commands processed since last HSM reset	Counted

The hsmPartitionTable

This table provides a list of all the partition information on the managed element.

Item	Type	Description	Values
hsmPartitionSerialNumber	DisplayString	Serial number for the partition	Generated
hsmPartitionLabel	DisplayString	Label assigned to the partition	Provided at partition creation
hsmPartitionActivated	TruthValue	Partition activation flag	Set by policy
hsmPartitionStorageTotalBytes	Unsigned32	Total storage capacity in bytes of the partition	Set or calculated at partition creation or re-size
hsmPartitionStorageAllocatedBytes	Unsigned32	Number of allocated (in use) bytes on the partition	Calculated
hsmPartitionStorageAvailableBytes	Unsigned32	Number of available (unused) bytes on the partition	Calculated
hsmPartitionObjectCount	Unsigned32	Number of objects in the partition	Counted

hsmLicenseTable

This table provides a list of all the license information on the managed element. More than one HSM might be connected to a Host, so they are accessed with two indices; the first index identifies the HSM for which the license entry corresponds (hsmSerialNumber), the second is the index for the corresponding license (hsmLicenseID).

Item	Type	Description	Values
hsmLicenseID	DisplayString	License identifier	Set at factory or at capability update
hsmLicenseDescription	DisplayString	License description	Set at factory or at capability update

hsmPolicyTable

This table provides a list of all the HSM policy information on the managed element.

Item	Type	Description	Values
hsmPolicyType	INTEGER	Type of policy	capability(1), policy(2)
hsmPolicyID	Unsigned32	Policy identifier	Numeric value identifies policy and is used as a index into the policy table
hsmPolicyDescription	DisplayString	Description of the policy	Brief text description of what the policy does
hsmPolicyValue	DisplayString	Current value of the policy	Brief text description to show current state/value of policy

hsmPartitionPolicyTable

This table provides a list of all the partition policy information on the managed element.

Item	Type	Description	Values
hsmPartitionPolicyType	INTEGER	Capability or policy	capability(1), policy(2)
hsmPartitionPolicyID	Unsigned32	Policy identifier	Numeric value identifies policy and is used as a index into the policy table
hsmPartitionPolicyDescription	DisplayString	Description of the policy	Brief text description of what the policy does
hsmPartitionPolicyValue	DisplayString	Current value of the policy	Brief text description to show current state/value of policy

hsmClientRegistrationTable

This table provides a list of registered clients.

Item	Type	Description	Values
hsmClientName	DisplayString	Name of the client	Name provided on client cert
hsmClientAddress	DisplayString	Address of the client	IP address of the client
hsmClientRequiresHTL	TruthValue	Flag specifying if HTL required for the client	Flag set at HSM host side to control client access
hsmClientOTTEpiry	INTEGER	OTT expiry time (-1 if not provisioned)	Expiry time, in seconds, for HTL OneTimeToken (range is 0-3600); -1 indicates not provisioned, 0 means never expires

hsmClientPartitionAssignmentTable

This table provides a list of assigned partitions for a given client.

Item	Type	Description	Values
hsmClientHsmSerialNumber	DisplayString	Index into the HSM table	--
hsmClientPartitionSerialNumber DisplayString	DisplayString	Index into the Partition Table	--

SNMP output compared to SafeNet tools output

For comparison, the following shows lunacm or lunash command outputs that provide HSM information equivalent to the SNMP information depicted in the tables above (from the HSM MIB).

HSM Information

At the HSM level the information in the outputs of "hsm show" and "hsm showp" and "hsm di" includes the following :

- SW Version
- FW Version
- HSM label
- Serial #
- HW Model
- Authentication Method
- RPV state
- FIPS mode
- HSM storage space (bytes)
- HSM storage space used (bytes)
- HSM storage free space (bytes)

- Performance level
- Max # of partitions
- # of partitions created
- # of free partitions
- Configuration (Cloning/CKE)
- License information similar to the output of the "hsm displayLicenses" command
- Policies as shown below.

```

Description Value
=====
Enable PIN-based authentication Allowed
Enable PED-based authentication Disallowed
Performance level 15
Enable domestic mechanisms & key sizes Allowed
Enable masking Disallowed
Enable cloning Allowed
Enable special cloning certificate Disallowed
Enable full (non-backup) functionality Allowed
Enable non-FIPS algorithms Allowed
Enable SO reset of partition PIN Allowed
Enable network replication Allowed
Enable Korean Algorithms Allowed
FIPS evaluated Disallowed
Manufacturing Token Disallowed
Enable Remote Authentication Allowed
Enable forcing user PIN change Allowed
Enable portable masking key Allowed
Enable partition groups Disallowed
Enable remote PED usage Disallowed
Enable External Storage of MTK Split Disallowed
HSM non-volatile storage space 2097152
Enable HA mode CGX Disallowed
Enable Acceleration Allowed
Enable unmasking Allowed
Enable FW5 compatibility mode Disallowed
Unsupported Disallowed
Unsupported Disallowed
Enable ECIES support Disallowed
The following policies are set due to current configuration of
this HSM and cannot be altered directly by the user.

```

```

Description Value
=====
PIN-based authentication True
The following policies describe the current configuration of
this HSM and may be changed by the HSM Administrator.
Changing policies marked "destructive" will zeroize (erase
completely) the entire HSM.

```

Description	Value	Code	Destructive
Allow cloning	On	7	Yes
Allow non-FIPS algorithms	On	12	Yes
SO can reset partition PIN	On	15	Yes
Allow network replication	On	16	No
Allow Remote Authentication	On	20	Yes

Force user PIN change after set/reset	Off	21	No
Allow offboard storage	On	22	Yes
Allow Acceleration	On	29	Yes
Allow unmasking	On	30	Yes

Partition Information

At the HSM Partition level the information in the outputs of "partition show" and "partition showp" includes the following :

- Partition Name
- Partition Serial #
- Activation State
- AutoActivation State
- Partition storage space (bytes)
- Partition storage space used (bytes)
- Partition storage free space (bytes)
- Partition Object Count
- Partition Policies from the Partition showpolicies command

```
lunash:> partition showPolicies -partition mypartition
```

```
Partition Name: mypartition
Partition Num: 65038002
```

The following capabilities describe this partition and can never be changed.

Description	Value
=====	=====
Enable private key cloning	Allowed
Enable private key wrapping	Disallowed
Enable private key unwrapping	Allowed
Enable private key masking	Disallowed
Enable secret key cloning	Allowed
Enable secret key wrapping	Allowed
Enable secret key unwrapping	Allowed
Enable secret key masking	Disallowed
Enable multipurpose keys	Allowed
Enable changing key attributes	Allowed
Enable PED use without challenge	Allowed
Allow failed challenge responses	Allowed
Enable operation without RSA blinding	Allowed
Enable signing with non-local keys	Allowed
Enable raw RSA operations	Allowed
Max failed user logins allowed	10
Enable high availability recovery	Allowed
Enable activation	Allowed
Enable auto-activation	Allowed
Minimum pin length (inverted: 255 - min)	248
Maximum pin length	255
Enable Key Management Functions	Allowed
Enable RSA signing without confirmation	Allowed

Enable Remote Authentication	Allowed
Enable private key unmasking	Allowed
Enable secret key unmasking	Allowed
Enable RSA PKCS mechanism	Allowed
Enable CBC-PAD (un)wrap keys of any size	Allowed
Enable private key SFF backup/restore	Disallowed
Enable secret key SFF backup/restore	Disallowed
Enable Secure Trusted Channel	Allowed

The following policies are set due to current configuration of this partition and may not be altered directly by the user.

Description	Value
=====	=====
Challenge for authentication not needed	False

The following policies describe the current configuration of this partition and may be changed by the HSM Administrator.

Description	Value	Code
=====	=====	=====
Allow private key cloning	On	0
Allow private key unwrapping	On	2
Allow secret key cloning	On	4
Allow secret key wrapping	On	5
Allow secret key unwrapping	On	6
Allow multipurpose keys	On	10
Allow changing key attributes	On	11
Ignore failed challenge responses	On	15
Operate without RSA blinding	On	16
Allow signing with non-local keys	On	17
Allow raw RSA operations	On	18
Max failed user logins allowed	10	20
Allow high availability recovery	On	21
Allow activation	Off	22
Allow auto-activation	Off	23
Minimum pin length (inverted: 255 - min)	248	25
Maximum pin length	255	26
Allow Key Management Functions	On	28
Perform RSA signing without confirmation	On	29
Allow Remote Authentication	On	30
Allow private key unmasking	On	31
Allow secret key unmasking	On	32
Allow RSA PKCS mechanism	On	33
Allow CBC-PAD (un)wrap keys of any size	On	34
Force Secure Trusted Channel	Off	37

Command Result : 0 (Success)
[myluna] lunash:>

Frequently Asked Questions

This section provides additional information by answering questions that are frequently asked by our customers.

We want to use SNMP to remotely monitor and manage our installation – why do you not support such standard SNMP traps as CPU and Memory exhaustion?

Those sorts of traps were specifically excluded because they can be used to establish a covert channel (an illicit signaling channel that can be used to communicate from a high assurance “area” to a lower assurance one in an effort to circumvent the security policy). Resource exhaustion events/alerts are the oldest known form of covert channel signaling. Exercise care with any HSM product that does allow such traps - what other basic security holes might be present?

Software Maintenance and Updates

This chapter describes how to maintain and update the functionality of your HSMs by performing the following tasks:

- ["About Updating SafeNet HSM" below](#)
- ["Advanced Configuration Upgrades" on page 352](#)
- ["Apply a Capability Upgrade/Update to HSM" on page 357](#)
- ["Firmware Rollback" on page 359](#)
- ["Serial Number Handling" on page 361](#)

About Updating SafeNet HSM

Your SafeNet HSM system consists of components that might, from time to time, require updating to newer versions. The newer version might have fixes or functional improvements that are useful or important for your application. The components that might be affected are:

- Client software
- SafeNet HSM firmware

Some new features are implemented entirely in the Lunaclient software, and have no dependency on HSM firmware. Some new features are implemented entirely in the HSM firmware, and are independent of the associated client software.

Some new features require that both the HSM firmware and the client software be updated, to take full advantage of the feature.

The instructions that accompany the update detail the dependencies.

In addition, you might wish to add purchased capability upgrades, which is a separate procedure; see ["Apply a Capability Upgrade/Update to HSM" on page 357](#).

How Firmware Updates Affect Agency Validation

In the case of FIPS 140, cryptographic devices are evaluated as a combination of hardware and firmware. Therefore, if either of those elements changes, the device is no longer covered by the current validation certificate. If you require that equipment used in your application be FIPS 140-2 level 3 validated (for example), you can use the most recent of our relevant HSM products that has been validated - which applies to a specific hardware and firmware combination. If we release a newer version of firmware, your own security or compliance policies would not permit you to install that update until we have submitted the updated HSM for [re-] evaluation, and a new validation certificate has been issued.

As a general rule (exceptions are possible) we submit HSMs with new firmware versions. If the changes are small or do not affect areas that concern the FIPS evaluators, then the re-evaluation is performed on a delta basis and therefore occurs relatively quickly. For a completely new product or major revision, the evaluators require a complete re-submission and the process takes roughly a year from submission to certificate. Therefore, when a FIPS-

candidate firmware version exists, our practice is to ship the respective HSM product with the most recent FIPS-validated firmware version installed, and with the candidate version as a standby update file (ready to install, but not yet installed). This ensures that customers who require validated systems continue to get them, and that customers who do not require validated systems are able to easily and quickly apply the update if they choose to do so.

The obvious trade-off is that customers who elect to remain with the as-shipped installed firmware version are maintaining the FIPS compliance at the cost of any upgraded capabilities or any security or functional fixes that are part of the firmware update. Similarly, customers who choose to perform the update benefit from the improved capabilities and any security or functional fixes, but at the cost of moving out of FIPS compliance.

Updating the Client Software

To update the software on a Client, you simply remove the older version and Install the newer, using the same procedure (for your operating system) that you used for the original software installation. That applies to SafeNet Network HSM Client software itself, as well as to the SDK material.

SafeNet HSM Lunaclient installers attempt to preserve existing configuration files, but also edit a version if updated settings are required by the options that you choose to install.

Upgrading the SafeNet PCIe HSM or SafeNet USB HSM/SafeNet Backup HSM firmware

To upgrade the firmware on a SafeNet PCIe HSM or SafeNet USB HSM/SafeNet Backup HSM, run a LunaCM command on a SafeNet HSM client computer

- that contains a copy of the firmware upgrade (.fuf) file with its associated firmware authentication code (.txt) file, and
 - contains the SafeNet PCIe HSM, or
 - is connected to the SafeNet USB HSM/SafeNet Backup HSM that you want to upgrade.
1. Copy the firmware file (<fw_filename>.fuf) from the firmware folder on the media (usually a downloaded archive) to the client root directory:
 - Windows: C:\Program Files\SafeNet\LunaClient
 - Linux/AIX: /usr/safenet/lunaclient/bin
 - Solaris/HP-UX: /opt/safenet/lunaclient/bin
 2. Obtain the firmware authorization code:
 - a. Contact Gemalto Customer Support (support@safenet-inc.com). The firmware authorization code is provided as a .txt file.
 - b. Copy the <fw_auth_code>.txt file to the SafeNet HSM client root directory:
 - Windows: C:\Program Files\SafeNet\LunaClient
 - Linux/AIX: /usr/safenet/lunaclient/bin
 - Solaris/HP-UX: /opt/safenet/lunaclient/bin
 3. Launch the LunaCM utility:

Windows

 - a. Open a Command Prompt window
(Start > Programs > Accessories > Command Prompt).
 - b. Change to the SafeNet HSM client root directory:

```
cd C:\Program Files\SafeNet\LunaClient
```

- c. Enter the following command

```
Lunacm
```

Linux/AIX

- a. Open a terminal window and change to the SafeNet HSM client root directory:

```
/usr/safenet/lunaclient/bin
```

- b. Enter the following command:

```
./lunacm
```

HP-UX/Solaris

- a. Open a terminal window and change to the SafeNet HSM client root directory:

```
/opt/safenet/lunaclient/bin
```

- b. Enter the following command:

```
./lunacm
```

4. If more than one HSM is installed, note which slot is assigned to that HSM and select it.

```
slot set -slot <number>
```

5. Enter the following command, to log in to the HSM. (For a PED-authenticated HSM, omit the password; you are prompted to respond to the Luna PED):

For legacy systems

```
hsm login [-password <password>]
```

For current systems (firmware 6.22.0 or newer)

```
role login -name <name of role> -password <password>
```

6. Enter the following command to upgrade the firmware on the HSM:

```
hsm -updateFirmware -fuf <fw_filename>.fuf -authcode <fw_authcode_filename>.txt
```

Advanced Configuration Upgrades

This page describes configuration upgrades, how they work and interact, etc.

For instructions to apply a Configuration Upgrade to your HSM, see ["Apply a Capability Upgrade/Update to HSM" on page 357](#).

SafeNet offers advanced configuration upgrades for its HSM products, some examples of which are listed in the following tables.

SafeNet delivers advanced configuration upgrades for SafeNet Network HSM as a secure package update. Follow the steps of ["Apply a Capability Upgrade/Update to HSM" on page 357](#) to apply the update. These are sometimes referred to as CUFs, but those refer to the USB HSM and the PCIe HSM; for the Network HSM, CUFs must be packaged as secure packages in order for the appliance to recognize them and handle them properly.

For SafeNet PCIe HSM and SafeNet USB HSM, you receive a firmware update file (FUF) or a capability update file (CUF¹).

¹Capability Update File - see "Advanced Configuration Upgrades".



Note: This is not necessarily a complete list, please check with your sales representative for the full list.



Note: Part numbers shown here are for field upgrades. The same upgrades are often available for factory installation when you purchase a new SafeNet HSM product. Those have different part numbers (ask your sales representative). Not all field upgrades have an equivalent factory-applied version, because we ship HSMs with the most recent FIPS-validated firmware version, and some newer upgrades might require more recent firmware, so they cannot be installed at the factory.

Table 1: SafeNet Network HSM configuration upgrades

Configuration upgrade	Part number
Maximum memory	908-000086-001
Korean algorithms	908-000139-002
ECIES acceleration	908-000175-001
5 partitions	908-000201-001
10 partitions	908-000202-001
15 partitions	908-000203-001
20 partitions	908-000204-001
35 partitions	908-000379-001
50 partitions	908-000235-001
75 partitions	908-000280-001
100 partitions	908-000232-001
Enable Small Form-factor Backup (SA)	908-000220-001
Enable Per-Partition Security Officer (PPSO)	908-000263-001



Note: Increasing the number of partitions is not destructive; it does not erase existing partitions and objects. However, simply increasing the number of partition licenses does not increase memory. Depending on the size of the original partitions (did you re-size them to use large amounts of memory, or "all available memory"?), you might need to resize the existing partitions to make room for the additional partitions. If a partition is occupied when it is to be resized, you might need to move some objects before resizing.



Note: You can apply 100 partitions without also upgrading to Maximum Memory, but this leaves very little memory for each partition. Usefulness depends upon your application, and the sizes of keys and objects that you would store in each partition.

Also, if you are using STC, then that requires 2 KB of partition space for each STC client that is registered to a given partition.



Note: If you are both

- upgrading from an earlier firmware version to HSM firmware 6.22.0 (or newer)

AND

- applying the Per-Partition SO (PPSO) capability update,

be aware that the PPSO capability update is destructive. Therefore, there is no need to re-size partitions.

Instead, to avoid unnecessary duplication of effort, you should

- safeguard (archive) any existing partition contents,
- then zeroize the HSM for a clean update,
- then perform both the firmware AND capability updates,
- and finally restore to new partitions.

Table 2: SafeNet PCIe HSM capability upgrades

Configuration upgrade	Part number
Korean algorithms	908-000138-002
ECIES acceleration	908-000177-001
Enable Small Form-factor Backup (PCIe)	908-000223-001

Table 3: SafeNet USB HSM configuration upgrades

Configuration upgrade	Part number
Korean algorithms	908-000156-002
ECIES acceleration	908-000179-001

Table 4: SafeNet Backup HSM configuration upgrades

Configuration upgrade	Part number
5 partitions	908-000083-001
10 partitions	908-000287-001
20 partitions	908-000085-001

Configuration upgrade	Part number
35 partitions	908-000281-001
50 partitions	908-000282-001
75 partitions	908-000283-001
100 partitions	908-000284-001

NOTE: SafeNet Remote Backup HSM comes with maximum memory and does not require a separate memory upgrade for larger numbers of partitions.

ECIES Acceleration

SafeNet offers ECIES support via a client-library shim. With the shim, ECIES 386-bit performance is approximately 40 operations per second. The ECIES acceleration configuration upgrade improves performance. This upgrade provides an approximately 5x performance increase compared to using the shim. If you choose to apply and use the configuration upgrade, you must remove the shim from your system configuration for the upgrade to have effect: shim use overrides acceleration.

Applying the ECIES advanced configuration upgrade is a destructive operation: objects already created on the HSM are destroyed. Therefore, you should apply this update when you first configure your HSM, before putting it into production (alternatively, you can back up any important objects and restore them onto the HSM after the upgrade).



Note: The full ECIES suite of mechanisms is not approved by NIST (that is, not all are FIPS 140-2 algorithms). Applying EITHER the ECIES shim OR this configuration upgrade option means that you can use all the available ECIES mechanisms when the HSM is **not** in the FIPS 140-2 mode of operation; however if FIPS 140-2 mode **is** asserted then some ECIES mechanisms are blocked.

Rollback Behavior

When it became possible to roll HSM firmware updates¹ back to an earlier version, some additional concerns became evident. The order in which you perform some activities becomes important.

An HSM that receives a firmware update arrives at that condition with any capabilities/features that were part of the HSM before the update was installed. The pre-update record of <firmware version+configuration> is set. If you rollback, you rollback² to exactly the state that was recorded, prior to the update. All the same capabilities/features would be available, because they were present before the firmware update.

Any capability that you added after a firmware update would be lost, if you then rolled back the firmware, because the pre-update record of <firmware version+configuration> did not include any capability that you added only post-update. In that case:

- If the late-installed capability **is not** dependent on the newer firmware, then you can simply install it again, on the

¹A newer version of client software, appliance software, or HSM firmware, to fix defects, or to improve security, or to modify/improve existing features, or to add enhancements. Updates are provided as needed, or as the product develops, for a hardware version.

²To return the HSM to its previous firmware version. This gives up any enhancements or fixes that were gained by the newer firmware version, as well as any upgrades that were installed after the firmware update (that is to be rolled back).

HSM at the rolled-back firmware version, and it will become part of the pre-update record the next time you update firmware.

- If the late-installed capability **is** dependent on the newer firmware, then you must do without that feature/capability until you once more update to a firmware version that can support it. At that time, you will need to re-install that capability upgrade¹.

The following table summarizes the options comparatively.

	Start with this	If you do this...	Result is this	If you next do this...	Result is this	If you next do this...	Result is this	If you next do this...	Result is this
Example 1 (Read across ==>)	f/w X and Capabilities A, B, C	Update to f/w Y	f/w Y and Capabilities A, B, C	Roll back to f/w X	f/w X and Capabilities A, B, C				
Example 2 (Read across ==>)		Add Capability D (no dependency)	f/w X and Capabilities A, B, C, D	Update to f/w Y	f/w Y and Capabilities A, B, C, D	Roll back to f/w X	f/w X and Capabilities A, B, C, D		
Example 3 (Read across ==>)		Update to f/w Y	f/w Y and Capabilities A, B, C	Add Capability D (no dependency)	f/w Y and Capabilities A, B, C, D	Roll back to f/w X	f/w X and Capabilities A, B, C		
Example 4 (Read across ==>)		Capability E depends on f/w Y; Attempt to add Capability E fails	f/w X and Capabilities A, B, C (unchanged)	Update to f/w Y	f/w Y and Capabilities A, B, C	Add Capability E (depends on f/w Y)	f/w Y and Capabilities A, B, C, E	Roll back to f/w X	f/w X and Capabilities A, B, C

¹A secure package that can be applied to the HSM to grant new capability or to enhance existing function.

	Start with this	If you do this...	Result is this	If you next do this...	Result is this	If you next do this...	Result is this	If you next do this...	Result is this

In Example 1, above, no capabilities change; only the firmware version.

In Example 2, above, D is added **before** firmware update; therefore the pre-update record includes capability D, so **D survives** firmware update and firmware rollback.

In Example 3, above, D is added **after** firmware update, the pre-update record does not include capability D, so **D does not survive** firmware rollback.

In Example 4, above, the pre-update record does not include capability E, so E does not survive firmware rollback.

We advise you to retain a copy of any in-field configuration upgrades.

Apply a Capability Upgrade/Update to HSM

SafeNet HSMs are shipped from the factory in specific configurations with specific sets of capabilities, to suit your requirements. It can happen that your requirements change over time. To future-proof your SafeNet HSM investment, you have the option to purchase Secure Capability Updates to enhance the performance or extend the capability of SafeNet systems already in your possession, as described in "[Advanced Configuration Upgrades](#)" on [page 352](#). The Secure Capability Update accomplishes system upgrades while safeguarding the integrity of your sensitive key material and of the system software.

Preparing to Upgrade

To ensure a trouble-free installation, you must prepare for the upgrade.

To prepare for the upgrade

1. Backup the application partition by cloning the contents to another SafeNet USB HSM.
2. On the host computer, acquire the capability update software files.
 - a. Follow the FTP instructions that are supplied in e-mail from SafeNet Customer Support (support@safenet-inc.com).
 - b. Unzip the files (as directed in the ftp instructions).

In some Windows configurations, you might not have authority to copy or unzip files directly into C:\Program Files\.... In that case, put the files in a known location that can be referenced in a lunacm command.

Installing the Upgrade Package

Once the files are unpacked and available on the host computer, open a command-prompt session.

To install the upgrade package

1. Go to the SafeNet Client directory and launch lunacm.

2. Log into the HSM:

For HSM with pre-6.22.0 firmware

```
lunacm:> hsm login
```

For HSM with version 6.22.0 or newer firmware

```
lunacm:> role login -name Administrator
```

3. Apply the new capability:

```
lunacm:>hsm updatecap -cuf \Users\me\Downloads\621-000099-001.CUF -authcode \User-
s\me\Downloads\621-000099-001_authcode.TXT
```

```
You are about to apply a destructive update.
All contents of the HSM will be destroyed.
All partition roles will be destroyed.
The domain will be destroyed.
```

```
Are you sure you wish to continue?
```

```
Type 'proceed' to continue, or 'quit' to quit now ->proceed
```

```
Capability update passed.
```

```
Command Result : No Error
```

```
lunacm:>hsm
```

4. Check that the new capability is in place:

```
lunacm:>hsm showpolicies
HSM Capabilities
 0: Enable PIN-based authentication : 0
 1: Enable PED-based authentication : 1
 2: Performance level : 15
 4: Enable domestic mechanisms & key sizes : 1
 6: Enable masking : 0
 7: Enable cloning : 1
 8: Enable special cloning certificate : 0
 9: Enable full (non-backup) functionality : 1
12: Enable non-FIPS algorithms : 1
15: Enable SO reset of partition PIN : 1
16: Enable network replication : 1
17: Enable Korean Algorithms : 1
18: FIPS evaluated : 0
19: Manufacturing Token : 0
20: Enable Remote Authentication : 1
21: Enable forcing user PIN change : 1
22: Enable offboard storage : 1
23: Enable partition groups : 0
25: Enable remote PED usage : 1
26: Enable External Storage of MTK Split : 0
27: HSM non-volatile storage space : 2097152
28: Enable HA mode CGX : 0
29: Enable Acceleration : 1
30: Enable unmasking : 1
31: Enable FW5 compatibility mode : 0
```

```

33: Maximum number of partitions : 100
34: Enable ECIES support : 0
35: Enable Single Domain : 1
36: Enable Unified PED Key : 1
37: Enable MofN : 1
38: Enable small form factor backup/restore : 0
39: Enable Secure Trusted Channel : 1
40: Enable decommission on tamper : 0
41: Enable Per-Partition SO : 1 <<=====
42: Enable partition re-initialize : 1

```

HSM Policies

```

0: PIN-based authentication : 0
1: PED-based authentication : 1
6: Allow masking : 0
7: Allow cloning : 1
12: Allow non-FIPS algorithms : 1
15: SO can reset partition PIN : 1
16: Allow network replication : 1
20: Allow Remote Authentication : 1
21: Force user PIN change after set/reset : 0
22: Allow offboard storage : 1
23: Allow partition groups : 0
25: Allow remote PED usage : 1
26: Store MTK Split Externally : 0
29: Allow Acceleration : 1
30: Allow unmasking : 1
31: Allow FW5 compatibility mode : 0
33: Current maximum number of partitions : 100
34: Allow ECIES support : 0
35: Force Single Domain : 0
36: Allow Unified PED Key : 0
37: Allow MofN : 1
38: Allow small form factor backup/restore : 0
39: Allow Secure Trusted Channel : 0
40: Allow decommission on tamper : 0
42: Allow partition re-initialize : 0

```

Command Result : No Error

lunacm:>

Firmware Rollback

When you perform a firmware update operation, a newer firmware version is installed in the HSM, and the firmware that was previously active is retained in case you wish to roll back to that previous version. This allows you to try out a new version, without being committed to it. At any time there can be no more than one active firmware and one potential rollback firmware.

From the factory, normally only the active firmware is installed, and there is no rollback option until you have updated firmware at least once.

If the HSM contains a rollback firmware version (call it 'B') and a currently active firmware version (call it 'C'), and you then perform a firmware update operation to raise the current version to a newer version (call it 'D'), then the 'C'

firmware assumes the rollback status and the 'B' version is now gone from the HSM. If you do perform rollback, then 'C' becomes the current version, and there is no rollback option from there.

CAUTION: The rollback operation is destructive to application partitions and contents, so perform backups, as necessary, before rolling back.



After rollback, the no-longer-valid client/partition assignment configuration files remain, and must be cleared before you create any new partitions. HSM initialization clears those files and is a **required** operation following firmware rollback.

For SafeNet PCIe HSM and SafeNet USB HSM, you can have newer firmware in the host file system, ready to install.

To roll back HSM firmware

1. Ensure that the host computer and, if applicable, any attached USB HSM or Backup HSM, are connected to an uninterruptible power supply.
2. Launch LunaCM.
3. Use **slot list** command to see the slot number for the desired HSM.
4. Use **slot set** command to select the slot corresponding to the HSM that is to have its firmware rolled back.
5. Use the **hsm showinfo** command to see the current firmware version and the rollback firmware version:

```
lunacm:> hsm showinfo  lunacm:> hsm showinfo

Partition Label -> myusbhsm
Partition Manufacturer -> Safenet, Inc.
Partition Model -> G5 Base
Partition Serial Number -> 150022
Partition Status -> OK
Token Flags ->
    CKF_RESTORE_KEY_NOT_NEEDED
    CKF_PROTECTED_AUTHENTICATION_PATH
    CKF_TOKEN_INITIALIZED
RPV Initialized -> Yes
Slot Id -> 1
Tunnel Slot Id -> 2
Session State -> CKS_RW_PUBLIC_SESSION
Role Status ->  none logged in
Token Flags ->
    TOKEN_KCV_CREATED
Partition OUID: 0000000000000000064a0200

Partition Storage:
    Total Storage Space:  262144
    Used Storage Space:   0
    Free Storage Space:  262144
    Object Count:         0
    Overhead:              9280

*** The HSM is NOT in FIPS 140-2 approved operation mode. ***

Firmware Version -> 6.27.0
Rollback Firmware Version -> 6.10.9
```



```
HSM Storage:
  Total Storage Space: 2097152
  Used Storage Space: 174288
  Free Storage Space: 1922864
  Allowed Partitions: 1
  Number of Partitions: 1
```

```
License Count -> 9
  1. 621000026-000 K6 base configuration
  1. 620127-000 Elliptic curve cryptography
  1. 620114-001 Key backup via cloning protocol
  1. 620109-000 PIN entry device (PED) enabled
  1. 621010358-001 Enable a split of the master tamper key to be s
tored externally
  1. 621010089-001 Enable remote PED capability
  1. 621000021-001 Performance level 15
  1. 621000079-001 Enable Small Form Factor Backup
  1. 621000099-001 Enable per-partition Security Officer
```

```
Command Result : No Error
```

6. Login if you have not already done so, and run the **hsm rollbackfw** command.

```
lunacm:> hsm login
```

```
Please attend to the PED.
```

```
Command Result : No Error
```

```
lunacm:> hsm rollbackFW
```

```
You are about to rollback the firmware.
The HSM will be reset.
Are you sure you wish to continue?
```

```
Type 'proceed' to continue, or 'quit' to quit now -> proceed
```

```
Rolling back firmware. This may take several minutes.
```

```
Firmware rollback passed. Resetting HSM
```

```
Command Result : No Error
```

7. Following rollback, initialize the HSM with command **hsm init** :

```
.
```

Serial Number Handling

Serial numbers of partitions changed after SafeNet HSM firmware 6.22.0.

The HSM serial number remains the HSM serial number, unchanged.

Application partition serial numbers are derived differently and are longer than they were previously.

Serial numbers for pre-existing application partitions (that were created on an HSM with firmware older than 6.22.0) are preserved when HSM firmware is updated.

New application partitions, created while an HSM is at firmware version 6.22.0 or newer, are assigned new, longer serial numbers, regardless of whether the partition is created as “legacy” or as PPSO type.

HA operation with SafeNet HSMs is unaffected by older serial numbers, new-style serial numbers, or any mix of the two. A failing member of an HA group can be replaced by a member with a different style of serial number.

If you have applications or scripts that rely on parsing HSM partition serial numbers (for example, log-file post-processing), some re-coding might be needed before you update.

Standards and Validations

When appropriate, we submit SafeNet products for validation/certification against international standards in the security, crypto, and data protection fields. This chapter discusses the major certifications that we routinely seek.



Note: If your application or your milieu does not require that HSM products carry validations like NIST's FIPS, or Common Criteria EAL, then this section would be of limited interest, except possibly for the reassurance that we design, test, manufacture, and handle our products

This chapter contains the following sections:

- ["About FIPS Validation" below](#)
- ["About HSM NOT in FIPS140-2 Approved Mode" on the next page](#)
- ["NIST SP 800-131A: Changes to FIPS-Supported Algorithms " on page 377](#)
- ["Common Criteria" on page 384](#)

About FIPS Validation

In many areas of the information security industry, validations against independent or government standards are considered a desirable or essential attribute of a product. The pre-eminent standard in the field is the FIPS (Federal Information Processing Standards) 140 standard from the United States government's NIST (National Institute of Standards and Technology) at <http://www.nist.gov>.

SafeNet routinely seeks FIPS validation for our HSM products. We were among the very first to achieve FIPS 140-1 validation for an HSM, and have since re-submitted our products when the standard changed (to 140-2) and whenever improvements to our product (or the introduction of new products) introduced enough change that the previous validation did not cover. Since the first, we have never failed to achieve the validation whenever we submit a product or variant.

In the case of appliance products, such as the SafeNet Network HSM validation is performed against the HSM Keycard inside the appliance, and not against the entire appliance. Furthermore, the validation of the HSM is for a particular firmware version, only.

However, the process of re-validation, though shorter than a new, first-time validation is nevertheless lengthy and involved. Therefore, whenever we introduce a new product, or a product variant that is sufficiently modified to require a new validation, there is a delay of several months until the validation certificate is granted.

Also to be considered is that validation can be a moving target. A product that received validation against the standard several years ago might not pass today, not because the standard has changed, but because interpretation has evolved or because testing organizations have revised their emphasis in some areas.

Finally, older standards give way to newer. Due to the expense and time constraints, companies tend to stay with a previous version of an optional standard until there is sufficient market demand for validation to the newer standard.

What does this mean to me?

Check with SafeNet to learn the most current validation status of any product, or click the NIST link, above. If FIPS validation is a primary concern for your application, you may need to use a previous, validated version of a product, and forego the latest improvements and features. During the lifetime of a product, we try to ensure that a FIPS-validated version remains available for purchase, despite the existence of newer firmware and hardware versions (see note below).

If the features of a new release are critical to your application, and FIPS validation is not a gating requirement, then you can use the newest product release. Based on past performance, the most recent release is likely to receive validation within months.



Note: It can happen that external circumstances make it impossible to fulfill the availability policy. An example from our own history was the introduction of the RoHS (Reduction of Hazardous Substances) legislation in the EU. Even where a company retained the capability to manufacture older-version units, they were no longer permitted to sell those older-design products to any country that abides by the RoHS regulations. In other cases, suppliers might no longer have stocked the non-RoHS parts that were used in the older design, and demand might not have justified their creating equivalent parts that meet RoHS standards. In such a situation, we could sell only the newer, RoHS-compliant product, while waiting for it to achieve FIPS (or other) validation.

"[About HSM NOT in FIPS140-2 Approved Mode](#)" below for more information.

About HSM NOT in FIPS140-2 Approved Mode

This is an option. You can change it.

If you run the `hsm show` command, you **might** see this text under the heading "FIPS 140-2 Operation":

The HSM is NOT in FIPS 140-2 approved operation mode

There is nothing wrong with your HSM. The message refers to an option (HSM policy code 12) that you can select according to your needs.

Here is how it works.

The FIPS-Approved Algorithms

The HSM is capable of a comprehensive set of cryptographic algorithms, allowing it to address the needs of a world data security market. However, some governments and agencies specify a restricted set of the available algorithms that suit their requirements or that limit the scope of testing that they are required to perform. FIPS is a very prominent set of standards in the industry, so we provide an option to exclude some algorithms from availability, so that an HSM owner can operate confidently in compliance with the FIPS 140-2 standard.

However, as technology advances and the cryptographic landscape shifts, agencies like NIST need to update their standards (like FIPS), dropping older, less effective options and adding newer ones as they become important and are vetted in the test labs. The list of FIPS-approved algorithms is subject to change.

For the most current list of FIPS approved algorithms, please visit the NIST web site at <http://csrc.nist.gov/>

For FIPS, you might be interested in:

- Modules in Process: <http://csrc.nist.gov/groups/STM/cmvp/documents/140-1/140InProgress.pdf>
- Completed Validations - Vendor List: <http://csrc.nist.gov/groups/STM/cmvp/documents/140-1/1401vend.htm>

For Common Criteria EAL, you can check:

- Completed CC evaluations: <http://www.commoncriteriaportal.org/products/>

For Payment Card Industry standards:

- https://www.pcisecuritystandards.org/approved_companies_providers/approved_pin_transaction_security.php then click the "HSM" tab (the third item in the product category headings)

Because we make available more algorithms and mechanisms than are covered under FIPS testing and certification, we are required by the FIPS 140-2 standard to provide the **option** for FIPS approved algorithms **only**, and to warn users if they are in a mode of operation that allows non-FIPS approved algorithms.

See for summary table of our supported mechanisms, and their FIPS-approved status.

What Does This Mean For Your Application?

The only difference between the two modes is that when Non-FIPS algorithms are **disallowed**, then you are operating in fully FIPS-compliant mode, and you have access to **only** the algorithms listed by NIST for the standard (the algorithms that satisfy the standard). When Non-FIPS algorithms are allowed, then you have exactly the same HSM appliance, except that it now offers you the possibility to use many more algorithms that are not FIPS 140-2 approved. This is useful where FIPS 140-2 standard is not a requirement (in many countries and organizations around the world). That is, the HSM treats FIPS-allowed algorithms identically when in either mode, but in non-FIPS mode the HSM offers additional capability that is not available in FIPS mode.

Your choice, then, is which is more important in your situation:

- that you have the ability to use those additional algorithms with your application or,
- that you restrict yourself to using only the FIPS-approved algorithms (perhaps for policy or regulatory compliance)?

Thus, whoever is setting your policy must understand that it is the FIPS standard, and not SafeNet, that decides which algorithms are permissible. You then have the option to comply or not. The appliance remains just as highly secure in either mode, but you must choose the mode that satisfies your auditors and/or your application requirements.

What Are the Implications of Changing This Policy Setting?

Other than the compliance issue, described above, the major issue is that this is a destructive policy. That is, if you change this policy, the Partitions and their contents are lost. If you have any important keys, certificates, or other material stored on the SafeNet HSM, you will want to back them up before changing this, or any other destructive policy.

["About FIPS Validation" on page 363](#)

Migrating from Non-FIPS HSM to FIPS HSM

If you have objects stored in an application partition of an HSM with FIPS mode off, you might encounter a situation where you need to begin using those objects on an HSM with FIPS mode in force. Depending on the HSM and other factors, you have some options.

One approach is to clone the needed objects to a different HSM that has the FIPS mode policy set on (or to archive (back up) the objects onto a Backup HSM and restore them onto the second HSM).

Another option, requiring fewer HSMs, is to change the policy on the current HSM from off to on. The policy change is destructive, meaning that all your objects would be destroyed. Therefore, you would need to archive (back up) the application partition contents before changing the HSM policy on the primary HSM, then create a new partition on that HSM, after the policy change, and restore the objects into that partition.

The exact sequence and commands to perform that operation might vary, depending on circumstances and on the firmware level of the HSM. As well, the commands are slightly different if you are using lunash on SafeNet Network HSM (for a legacy partition), or if you are using lunacm to perform the task on a SafeNet PCIe HSM or on an NTLS-connected SafeNet Network HSM PPSO partition.



CAUTION: Before migrating to FIPS mode, check whether your Cryptoki application and the objects it uses for its crypto operations are suitable to run in FIPS mode for the particular version of SafeNet HSM firmware you are using. You can confirm that an application works in FIPS mode by setting up a test environment with a development HSM configured in FIPS mode. Specifically you will need to confirm that your application uses only FIPS approved mechanisms and key sizes. (See ["Supported Mechanisms" on page 1.](#))

Be aware that FIPS object type and object size restrictions vary from firmware version to firmware version, usually becoming more restrictive in newer releases. If our documentation said that a particular mechanism or object type/size was acceptable in FIPS-mode for firmware that accompanied a previous release, that might no longer be the case if you are updating to a more current firmware version.

Start with objects in partition on Non-FIPS HSM	End with objects in partition on FIPS HSM	Lunash	Lunacm
SafeNet USB HSM	A different SafeNet USB HSM - (Backup HSM is not supported with SafeNet USB HSM)	n/a	- Clone the FIPS-approved objects individually from the HSM with FIPS policy OFF to the HSM with FIPS policy ON
SafeNet PCIe HSM	A different SafeNet PCIe HSM card in the same host	n/a	- Clone the FIPS-approved objects individually from the HSM with FIPS policy OFF to the HSM with FIPS policy ON

Start with objects in partition on Non-FIPS HSM	End with objects in partition on FIPS HSM	Lunash	Lunacm	
SafeNet PCIe HSM	The same SafeNet PCIe HSM card with the policy changed	n/a	<ul style="list-style-type: none"> - note the HSM and partition settings before proceeding, so you can replicate them after the change to FIPS mode - list the objects on the source partition - for releases earlier than 6.0, review the "Mechanisms" chapter of the "SDK Reference Guide", to determine which (if any) objects on the source partition are not FIPS-approved. - for releases earlier than 6.0, delete from the source partition any objects that are not FIPS approved; you will no longer be able to store or use them when the HSM is in FIPS mode - Archive (back up) the objects from the primary HSM, with FIPS policy OFF, to a SafeNet Backup HSM - Switch the FIPS policy of the primary HSM to ON (which destroys the partition and all objects on the HSM) - re-initialize the HSM - create a new partition on the primary HSM - Restore all of the FIPS-acceptable objects from the backup HSM to the new partition on the HSM with FIPS policy ON - begin using FIPS-acceptable mechanisms to operate on/with the objects in the new partition 	
SafeNet Network HSM PPSO partition	A PPSO partition on the same SafeNet Network HSM with the policy changed	<ul style="list-style-type: none"> - note the HSM settings before proceeding, so you can replicate them after the change to FIPS mode [continue at client in lunacm] =====>> 	<ul style="list-style-type: none"> - note the partition settings before proceeding, so you can replicate them after the change to FIPS mode 	

Start with objects in partition on Non-FIPS HSM	End with objects in partition on FIPS HSM	Lunash	Lunacm	
		<ul style="list-style-type: none"> - Switch the FIPS policy of the primary HSM to ON (which destroys the partition and all objects on the HSM) - re-initialize the HSM - create a new partition on the primary HSM - restore client/partition assignments <p>[conclude at client in lunacm] =====>></p>	<ul style="list-style-type: none"> - list the objects on the source partition - review the "Mechanisms" chapter of the "SDK Reference Guide", to determine which (if any) objects on the source partition are not FIPS-approved. - delete from the source partition any objects that are not FIPS approved; you will no longer be able to store or use them when the HSM is in FIPS mode - Archive (back up) the objects from the source partition (on the HSM with FIPS policy OFF) to a SafeNet Backup HSM <p><<===== [continue at SafeNet Network HSM, in lunash]</p> <ul style="list-style-type: none"> - Restore all of the FIPS-acceptable objects from the backup HSM to the new partition on the HSM with FIPS policy ON - begin using FIPS-acceptable mechanisms to operate on/with the objects in the new partition 	
SafeNet Network HSM legacy partition	A legacy partition on the same SafeNet Network HSM with the policy changed	<ul style="list-style-type: none"> - note the HSM and partition settings before proceeding, so you can replicate them after the change to FIPS mode - Archive (back up) the objects from the primary HSM with FIPS policy 	n/a	

Start with objects in partition on Non-FIPS HSM	End with objects in partition on FIPS HSM	Lunash	Lunacm	
		OFF to a SafeNet Backup HSM - Switch the FIPS policy of the primary HSM to ON (which destroys the partition and all objects on the HSM) - re-initialize the HSM - create a new partition on the primary HSM - Restore all of the FIPS-acceptable objects from the backup HSM to the new partition on the HSM with FIPS policy ON - restore client/partition assignments - begin using FIPS-acceptable mechanisms to operate on/with the objects in the new partition		

Note: After the HSM has been set to FIPS mode, only FIPS-approved objects, as allowed by the HSM's current version of firmware, can be restored onto the new partition.



For the KE (Key Export) HSM configuration, private keys cannot be backed up/restored. They can only be unwrapped back onto the new FIPS-mode partitions.

SafeNet Network HSM or SafeNet PCIe HSM application partition (f/w 6.22.0 or newer) in LunaCM

The following instructions are for

- a SafeNet PCIe HSM application partition (locally installed), or
- a SafeNet Network HSM application partition via secure network connection (NTLS or STC)

In both cases, the HSM firmware is version 6.22.0 or newer. The SafeNet Network HSM application partition is a PPSO partition (not legacy), which can be administered only via client connection (lunacm).

The following instructions assume that a SafeNet Backup HSM is connected locally to the SafeNet HSM Client host computer.

The following instructions assume that you have a number of objects on the application partition that you have been using while the HSM is non-FIPS, and that you wish to continue using after you switch the HSM to FIPS mode.

For illustrative purposes, the example source partition in the following instructions includes a few objects that are not permitted in an HSM running in FIPS mode. This includes verifying the HSM's FIPS mode,

For SafeNet Network HSM, the initial action to set up the application partition must be performed by the HSM SO at the SafeNet Network HSM appliance using LunaSH, via SSH or via local serial console.

For SafeNet PCIe HSM, the HSM SO creates the partition in lunacm.

Thereafter, either for a SafeNet PCIe HSM local application partition (with PSO), or for a SafeNet Network HSM application partition (with PSO) viewed over NTLS connection, the actions are the same and are performed in lunacm as follows.

1. Use **slot set slot** command to set the current-slot focus to the source application partition on the non-FIPS-mode HSM. Type:

```
lunacm:> slot set slot 1

Current Slot Id:      1      (Luna User Slot 6.27.0 (PED) Signing With Cloning Mode)

Command Result : No Error

lunacm:>
```

2. Use **partition showinfo** command to view the HSM information for the SafeNet HSM that contains the source partition. Confirm that the HSM is "...NOT in FIPS 140-2 approved operation mode" before the HSM is switched to FIPS mode.

```
lunacm:> partition showinfo

Partition Label -> Cryptoki User
Partition Manufacturer -> Safenet, Inc.
Partition Model -> K6 Base
Partition Serial Number -> 450039014
Partition Status -> OK
Token Flags ->
    CKF_LOGIN_REQUIRED
    CKF_USER_PIN_INITIALIZED
    CKF_RESTORE_KEY_NOT_NEEDED
    CKF_PROTECTED_AUTHENTICATION_PATH
    CKF_TOKEN_INITIALIZED
RPV Initialized -> Yes
Slot Id -> 1
Tunnel Slot Id -> 4
Session State -> CKS_RW_USER_FUNCTIONS
Role Status -> Crypto Officer logged in
Token Flags ->
    TOKEN_KCV_CREATED
Partition OUID: 40000006f3040000f7dd0600
```

```

Partition Storage:
  Total Storage Space: 2087864
  Used Storage Space: 0
  Free Storage Space: 2087864
  Object Count: 0
  Overhead: 9288

```

*** The partition is in FIPS 140-2 approved operation mode. ***

Command Result : No Error

```
lunacm:> slot set slot 6
```

```

Current Slot Id: 6 (SafeNet USB HSM 6.10.9 (PED) Backup Device)

```

Command Result : No Error

```
lunacm:> hsm si
```

```

HSM Label -> G5Backupup
HSM Manufacturer -> Safenet, Inc.
HSM Model -> G5Backupup
HSM Serial Number -> 475292
HSM Status -> OK
Token Flags ->
  CKF_RNG
  CKF_LOGIN_REQUIRED
  CKF_USER_PIN_INITIALIZED
  CKF_RESTORE_KEY_NOT_NEEDED
  CKF_PROTECTED_AUTHENTICATION_PATH
  CKF_TOKEN_INITIALIZED
Firmware Version -> 6.10.9
Rollback Firmware Version -> Not Available

```

```

HSM Auditor: Not Supported
HSM Logging: Not Supported

```

```

RPV Initialized -> No
Slot Id -> 6
Session State -> CKS_RW_PUBLIC_SESSION

```

```
SO Status-> Not Logged In
```

```

SO Failed Logins-> 0
SO Flags ->
  CONTAINER_KCV_CREATED

```

```

HSM Storage:
  Total Storage Space: 16252928
  Used Storage Space: 147352
  Free Storage Space: 16105576
  Allowed Partitions: 20
  Number of Partitions: 7

```

```

SO Storage:
  Total Storage Space: 262144

```

```

Used Storage Space: 0
Free Storage Space: 262144
Object Count: 0

```

*** The HSM is NOT in FIPS 140-2 approved operation mode. ***

License Count -> 5

```

1. 621010355-000 SafeNet Remote Backup HSM base configuration
1. 621000006-001 Enabled for 15.5 megabytes of object storage
1. 621000007-001 Enable the master tamper key to be stored externally
1. 621000008-001 Enable remote PED capability
1. 621000005-001 Maximum 20 partitions

```

Command Result : No Error

lunacm:>

3. Use **slot set -slot** command to set the current-slot focus to the slot with the Backup HSM. Type:

```
lunacm:> slot set -s 6
```

```
Current Slot Id: 6 (SafeNet USB HSM 6.0.8 (PED) Backup Device)
```

Command Result : No Error

lunacm:>

4. Use **partition archive list** to show the contents of the backup partition. Type:

```
lunacm:> partition archive list -slot 6
```

HSM Storage Information for slot 6:

```

Total HSM Storage Space: 16252928
Used HSM Storage Space: 147352
Free HSM Storage Space: 16105576
Allowed Partitions: 20
Number Of Partitions: 7

```

Partition list for slot 6

Number of partition: 7

```

Name: John2Backup2
Total Storage Size: 1684
Used Storage Size: 1643
Free Storage Size: 41
Number Of Objects: 2

```

```

Name: JohnK6backup
Total Storage Size: 41340
Used Storage Size: 40210
Free Storage Size: 1130
Number Of Objects: 85

```

```
Name: 6106Backup
```

```

Total Storage Size:      41340
Used Storage Size:      40210
Free Storage Size:      1130
Number Of Objects:      85

Name:                    UserPartitionBackup
Total Storage Size:      2436
Used Storage Size:      2353
Free Storage Size:      83
Number Of Objects:      6

Name:                    UserPartitionBackup1
Total Storage Size:      2436
Used Storage Size:      2353
Free Storage Size:      83
Number Of Objects:      6

Name:                    John2Backup
Total Storage Size:      1684
Used Storage Size:      1643
Free Storage Size:      41
Number Of Objects:      2

Name:                    6101Backup
Total Storage Size:      41340
Used Storage Size:      40210
Free Storage Size:      1130
Number Of Objects:      85

```

Command Result : No Error

lunacm:>

5. Use **hsm showinfo** to verify that the newly initialized HSM is now in FIPS mode. Type:

```

lunacm:> hsm showinfo

HSM Label -> G5Backkup
HSM Manufacturer -> Safenet, Inc.
HSM Model -> G5Backup
HSM Serial Number -> 475292
HSM Status -> OK
Token Flags ->
    CKF_RNG
    CKF_LOGIN_REQUIRED
    CKF_USER_PIN_INITIALIZED
    CKF_RESTORE_KEY_NOT_NEEDED
    CKF_PROTECTED_AUTHENTICATION_PATH
    CKF_TOKEN_INITIALIZED
Firmware Version -> 6.0.8
Rollback Firmware Version -> Not Available

HSM Auditor: Not Supported
HSM Logging: Not Supported

RPV Initialized -> No
Slot Id -> 6

```

```
Session State -> CKS_RW_PUBLIC_SESSION
```

```
SO Status-> Not Logged In
```

```
SO Failed Logins-> 0
```

```
SO Flags ->
```

```
CONTAINER_KCV_CREATED
```

```
HSM Storage:
```

```
Total Storage Space: 16252928
Used Storage Space: 147352
Free Storage Space: 16105576
Allowed Partitions: 20
Number of Partitions: 7
```

```
SO Storage:
```

```
Total Storage Space: 262144
Used Storage Space: 0
Free Storage Space: 262144
Object Count: 0
```

```
*** The HSM is NOT in FIPS 140-2 approved operation mode. ***
```

```
License Count -> 5
```

```
1. 621010355-000 SafeNet Remote Backup HSM base configuration
1. 621000006-001 Enabled for 15.5 megabytes of object storage
1. 621000007-001 Enable the master tamper key to be stored externally
1. 621000008-001 Enable remote PED capability
1. 621000005-001 Maximum 20 partitions
```

```
Command Result : No Error
```

```
lunacm:> slot set -s 1
```

```
Current Slot Id: 1 (Luna User Slot 6.22.0 (PED) Signing With Cloning Mode)
```

```
Command Result : No Error
```

```
lunacm:> role show -name Crypto Officer
```

```
State of role 'Crypto Officer':
```

```
Primary authentication type: PED
Secondary authentication type: PIN
Failed login attempts before lockout: 10
```

```
Command Result : No Error
```

```
lunacm:>
```

6. Use **partition showinfo** to verify that the new partition is in FIPS mode. Type:

```
lunacm:> partition showinfo
```

```
Partition Label -> Cryptoki User
Partition Manufacturer -> Safenet, Inc.
Partition Model -> K6 Base
```

```

Partition Serial Number -> 450039014
Partition Status -> OK
Token Flags ->
    CKF_LOGIN_REQUIRED
    CKF_USER_PIN_INITIALIZED
    CKF_RESTORE_KEY_NOT_NEEDED
    CKF_PROTECTED_AUTHENTICATION_PATH
    CKF_TOKEN_INITIALIZED
RPV Initialized -> Yes
Slot Id -> 1
Tunnel Slot Id -> 4
Session State -> CKS_RW_USER_FUNCTIONS
Role Status -> Crypto Officer logged in
Token Flags ->
    TOKEN_KCV_CREATED
Partition OID: 40000006f3040000f7dd0600

```

```

Partition Storage:
    Total Storage Space: 2087864
    Used Storage Space: 0
    Free Storage Space: 2087864
    Object Count: 0
    Overhead: 9288

```

*** The partition is in FIPS 140-2 approved operation mode. ***

Command Result : No Error

lunacm:>

lunacm:>

7. Use **partition archive restore** to restore the contents of the backup partition onto the newly created application partition on the now FIPS-mode HSM. Type:

```
lunacm:> partition archive restore -slot 6 -partition 6106Backup
```

Logging in to partition 6106Backup on slot 6 as the user.

Please attend to the PED.

Verifying that all objects can be restored...

85 objects will be restored.

Restoring objects...

Cloned object 122 from partition 6106Backup (new handle 20).

Cloned object 123 from partition 6106Backup (new handle 25).

Cloned object 124 from partition 6106Backup (new handle 26).

Cloned object 125 from partition 6106Backup (new handle 27).

Cloned object 126 from partition 6106Backup (new handle 28).

Cloned object 127 from partition 6106Backup (new handle 29).

Failed to clone object 128 from partition 6106Backup (CKR_ATTRIBUTE_VALUE_INVALID).

Cloned object 129 from partition 6106Backup (new handle 30).

Failed to clone object 130 from partition 6106Backup (CKR_ATTRIBUTE_VALUE_INVALID).

Failed to clone object 131 from partition 6106Backup (CKR_KEY_TYPE_INCONSISTENT).

Cloned object 132 from partition 6106Backup (new handle 31).

Cloned object 133 from partition 6106Backup (new handle 32).

Cloned object 136 from partition 6106Backup (new handle 35).

Cloned object 137 from partition 6106Backup (new handle 36).

Cloned object 138 from partition 6106Backup (new handle 37).

Cloned object 139 from partition 6106Backup (new handle 38).

Cloned object 143 from partition 6106Backup (new handle 42).

Cloned object 144 from partition 6106Backup (new handle 43).

Cloned object 145 from partition 6106Backup (new handle 44).

Cloned object 148 from partition 6106Backup (new handle 47).

Cloned object 149 from partition 6106Backup (new handle 48).

Cloned object 150 from partition 6106Backup (new handle 49).

Cloned object 151 from partition 6106Backup (new handle 50).

Cloned object 152 from partition 6106Backup (new handle 51).

Failed to clone object 153 from partition 6106Backup (CKR_KEY_TYPE_INCONSISTENT).

Cloned object 154 from partition 6106Backup (new handle 52).

Cloned object 158 from partition 6106Backup (new handle 56).

Cloned object 159 from partition 6106Backup (new handle 57).

Cloned object 160 from partition 6106Backup (new handle 58).

Cloned object 161 from partition 6106Backup (new handle 59).

Failed to clone object 162 from partition 6106Backup (CKR_ATTRIBUTE_VALUE_INVALID).

Cloned object 163 from partition 6106Backup (new handle 60).

Cloned object 164 from partition 6106Backup (new handle 61).

Cloned object 165 from partition 6106Backup (new handle 62).

Failed to clone object 166 from partition 6106Backup (CKR_ATTRIBUTE_VALUE_INVALID).

Cloned object 167 from partition 6106Backup (new handle 63).

Cloned object 168 from partition 6106Backup (new handle 64).

Cloned object 169 from partition 6106Backup (new handle 65).

Cloned object 170 from partition 6106Backup (new handle 66).

Cloned object 171 from partition 6106Backup (new handle 67).

Cloned object 175 from partition 6106Backup (new handle 71).

Failed to clone object 176 from partition 6106Backup (CKR_KEY_TYPE_INCONSISTENT).

Cloned object 177 from partition 6106Backup (new handle 72).

Cloned object 180 from partition 6106Backup (new handle 75).

Cloned object 181 from partition 6106Backup (new handle 76).

Cloned object 182 from partition 6106Backup (new handle 77).

Cloned object 183 from partition 6106Backup (new handle 78).

Failed to clone object 184 from partition 6106Backup (CKR_ATTRIBUTE_VALUE_INVALID).

Cloned object 185 from partition 6106Backup (new handle 79).

Cloned object 186 from partition 6106Backup (new handle 80).

Cloned object 187 from partition 6106Backup (new handle 81).

Cloned object 188 from partition 6106Backup (new handle 82).

Cloned object 189 from partition 6106Backup (new handle 83).

Failed to clone object 190 from partition 6106Backup (CKR_ATTRIBUTE_VALUE_INVALID).

Failed to clone object 191 from partition 6106Backup (CKR_ATTRIBUTE_VALUE_INVALID).

Cloned object 192 from partition 6106Backup (new handle 84).

Cloned object 193 from partition 6106Backup (new handle 85).

Cloned object 194 from partition 6106Backup (new handle 86).

Cloned object 198 from partition 6106Backup (new handle 90).


```

Cloned object 199 from partition 6106Backup (new handle 91).
Cloned object 200 from partition 6106Backup (new handle 92).
Cloned object 201 from partition 6106Backup (new handle 93).
Cloned object 204 from partition 6106Backup (new handle 96).
Cloned object 205 from partition 6106Backup (new handle 97).
Cloned object 206 from partition 6106Backup (new handle 98).
Cloned object 207 from partition 6106Backup (new handle 99).
Cloned object 208 from partition 6106Backup (new handle 100).
Cloned object 209 from partition 6106Backup (new handle 101).
Cloned object 210 from partition 6106Backup (new handle 102).
Failed to clone object 211 from partition 6106Backup (CKR_KEY_TYPE_INCONSISTENT).
Failed to clone object 212 from partition 6106Backup (CKR_ATTRIBUTE_VALUE_
INVALID).
Failed to clone object 213 from partition 6106Backup (CKR_ATTRIBUTE_VALUE_
INVALID).
Cloned object 214 from partition 6106Backup (new handle 103).
Cloned object 215 from partition 6106Backup (new handle 104).
Cloned object 216 from partition 6106Backup (new handle 105).
Cloned object 217 from partition 6106Backup (new handle 106).
Failed to clone object 218 from partition 6106Backup (CKR_ATTRIBUTE_VALUE_
INVALID).
Cloned object 219 from partition 6106Backup (new handle 107).
Cloned object 220 from partition 6106Backup (new handle 108).
Cloned object 221 from partition 6106Backup (new handle 109).
Cloned object 222 from partition 6106Backup (new handle 110).
Cloned object 223 from partition 6106Backup (new handle 111).
Cloned object 227 from partition 6106Backup (new handle 115).
Failed to clone object 228 from partition 6106Backup (CKR_KEY_TYPE_INCONSISTENT).
Cloned object 229 from partition 6106Backup (new handle 116).

Not all objects can be cloned. Please verify HSM configuration.

Restore Complete.

70 objects have been restored from partition 6106Backup on slot 6.

Command Result : No Error

lunacm:>

```

The example output above shows that some objects are not accepted into the now FIPS-mode HSM, because those objects are not permitted by FIPS mode. Other objects that are compliant are successfully transferred.

NIST SP 800-131A: Changes to FIPS-Supported Algorithms

As a result of the NIST SP 800-131A algorithm transitions, the list of algorithms that are supported in FIPS mode changes periodically.

Summary 2014

To comply with changes that came into effect on 01 January 2014, the following algorithms are not supported in SafeNet HSM 5.4, and higher, when the HSM is operated in FIPS mode:

- All digital signature and mac generation algorithms that use SHA-1 will no longer be supported, digital signature verification and mac verification will still be supported using SHA-1 for legacy purposes
- DSA Key Pair Generation and Signature Generation with a key size of less than 2048 bits is no longer supported
- DSA Signature Verification of 1024 bit keys is still supported for legacy purposes
- RSA Key Pair Generation and Signature Generation with a key size of less than 2048 bits is no longer supported
- RSA Signature Verification of 1024 bit keys is still supported for legacy purposes
- ECDSA DSA Key Pair Generation and Signature Generation with a curve size of less than 224 bits is no longer supported
- ECDSA Signature Verification with a curve size of less than 224 is still supported for legacy purposes
- RSA Key wrapping with an RSA Key of less than 2048 bits is no longer supported, however key unwrapping is still supported for legacy purposes
- RSA encryption with an RSA key of less than 2048 bits is no longer supported, however decryption is still supported for legacy purposes
- Diffie-Hellman key agreement with a key size of less than 2048 bits is no longer supported
- EC Diffie-Hellman key agreement with a curve size of less than 224 bits is no longer supported
- HMAC Generation with a key size less than 112 bits is no longer supported
- HMAC Verification with a key size less than 112 bits is supported for legacy purposes



Note: Use of SHA-1 is allowed for use in FIPS Approved mode, with the exception of digital signature/ MAC generation applications, for which is it not allowed in FIPS Mode.

Affected Algorithms

These changes affect the following algorithms:

Digital Signature Changes

Digital Signature	Key Pair Generation	Signature Generation	Signature Verification
DSA < 2048 with SHA-1	OFF	OFF	LEGACY
DSA < 2048 with SHA-2	OFF	OFF	LEGACY
RSA < 2048 with SHA-1	OFF	OFF	LEGACY
RSA < 2048 with SHA-2	OFF	OFF	LEGACY
ECDSA n < 224 with SHA-1	OFF	OFF	LEGACY
ECDSA n < 224 with SHA-2	OFF	OFF	LEGACY

Key Transport Changes

	Key Wrapping	Key Unwrapping
RSA < 2048	OFF	LEGACY

Encryption Changes

	Encryption	Decryption
RSA < 2048	OFF	LEGACY

Key Agreement Changes

	Key Agreement
Diffie-Hellman < 2048	OFF
EC Diffie-Hellman with n < 224	OFF

2-Key Triple DES Changes (before release 6.2.1)

	Encryption	Decryption	Key Wrapping	Key Unwrapping	CMAC KDF	HMAC KDF	CMAC Generation	CMAC Verification
2-Key Triple-DES	RESTRICTED	LEGACY	RESTRICTED	LEGACY	DEPRECATED	ACCEPTABLE	DEPRECATED	LEGACY

HMAC Changes

	MAC Generation	MAC Verification
HMAC < 112	OFF	LEGACY



Note: SHA-1 is allowed except for digital signature/MAC Generation

Impact on your operations

You can restore keys having legacy bit lengths from a backup. Legacy keys are retained on the HSM after the upgrade to SafeNet HSM 5.4 or later, and function in 'legacy' mode, only.

If you still wish to use the 'legacy' keys fully, you must exit FIPS mode:

- Backup your keys

- Switch off FIPS mode (change the policy), wiping out all keys
- Restore keys to the HSM that is no longer in FIPS mode

Mechanisms Affected

These changes affect the following mechanisms:

RSA FIPS Mechanisms

RSA FIPS Mechanism	FIPS	Changes in FIPS mode
CKM_RSA_PKCS_KEY_PAIR_GEN	YES	LEGACY less than 2048 bit
CKM_RSA_PKCS	YES	LEGACY less than 2048 bit
CKM_SHA1_RSA_PKCS	YES	LEGACY
CKM_RSA_PKCS_OAEP	YES	LEGACY less than 2048 bit
CKM_RSA_X9_31_KEY_PAIR_GEN	YES	LEGACY less than 2048 bit
CKM_RSA_FIPS_186_3_AUX_PRIME_KEY_PAIR_GEN	YES	LEGACY less than 2048 bit
CKM_RSA_FIPS_186_3_PRIME_KEY_PAIR	YES	NO, Already enforced at 2048 bit
CKM_RSA_X9_31_KEY_PAIR_GEN	YES	LEGACY less than 2048 bit
CKM_SHA1_RSA_X9_31	YES	LEGACY
CKM_SHA224_RSA_X9_31	YES	LEGACY less than 2048 bit
CKM_SHA256_RSA_X9_31	YES	LEGACY less than 2048 bit
CKM_SHA384_RSA_X9_31	YES	LEGACY less than 2048 bit
CKM_SHA512_RSA_X9_31	YES	LEGACY less than 2048 bit
CKM_RSA_PKCS_PSS	YES	LEGACY less than 2048 bit
CKM_SHA1_RSA_PKCS_PSS	YES	LEGACY
CKM_SHA224_RSA_PKCS	YES	LEGACY less than 2048 bit
CKM_SHA224_RSA_PKCS_PSS	YES	LEGACY less than 2048 bit
CKM_SHA256_RSA_PKCS	YES	LEGACY less than 2048 bit
CKM_SHA256_RSA_PKCS_PSS	YES	LEGACY less than 2048 bit
CKM_SHA384_RSA_PKCS	YES	LEGACY less than 2048 bit
CKM_SHA384_RSA_PKCS_PSS	YES	LEGACY less than 2048 bit
CKM_SHA512_RSA_PKCS	YES	LEGACY less than 2048 bit
CKM_SHA512_RSA_PKCS_PSS	YES	LEGACY less than 2048 bit

DSA FIPS Mechanisms

DSA FIPS Mechanism	FIPS	Changes in FIPS mode
CKM_DSA_KEY_PAIR_GEN	YES	LEGACY
CKM_DSA	YES	LEGACY
CKM_DSA_PARAMETER_GEN	YES	LEGACY
CKM_SHA1_DSA	YES	LEGACY
CKM_SHA224_DSA	YES	LEGACY
CKM_SHA256_DSA	YES	LEGACY

ECDSA Mechanisms

ECDSA Mechanism	FIPS	Changes in FIPS mode
CKM_EC_KEY_PAIR_GEN	YES	LEGACY for n < 224
CKM_ECDSA	YES	LEGACY for n < 224
CKM_SHA1_ECDSA	YES	LEGACY
CKM_SHA224_ECDSA	YES	LEGACY for n < 224
CKM_SHA256_ECDSA	YES	LEGACY for n < 224
CKM_SHA384_ECDSA	YES	LEGACY for n < 224
CKM_SHA512_ECDSA	YES	LEGACY for n < 224

HMAC Mechanisms

HMAC Mechanism	FIPS	Changes in FIPS mode
CKM_HMAC_SHA224	YES	LEGACY for key length less than 112 bits
CKM_HMAC_SHA256	YES	LEGACY for key length less than 112 bits
CKM_HMAC_SHA384	YES	LEGACY for key length less than 112 bits
CKM_HMAC_SHA512	YES	LEGACY for key length less than 112 bits
CKM_HMAC_SHA1	YES	LEGACY for key length less than 112 bits – ALSO HMAC based KDF is acceptable using an approved hash function including SHA-1

Diffie-Hellman Mechanisms

Diffie-Hellman Mechanisms	FIPS	Changes in FIPS mode
CKM_ECDH1_DERIVE	YES	LEGACY, for n < 224
CKM_ECDH1_COFACTOR_DERIVE	YES	LEGACY, for n < 224

Summary 2016 (Triple DES)

NIST document SP 800-131A places restrictions on the usage of Triple-DES, a.k.a. DES3, in FIPS mode.

As of 01 January 2016, 2-key Triple-DES is restricted to legacy operations (decryption, unwrapping, and CMAC verification) when the HSM is in FIPS mode. All other operations for Triple-DES now require the use of the 24-byte three-key variant.

The HSM refuses non-legacy operations when in FIPS mode. The restriction on 16-byte two-key Triple-DES is enforced by the module in firmware versions 6.22.0 and higher. The restriction on 24-byte two-key Triple-DES with non-unique keys is now enforced by the module in firmware versions 6.24.2 and higher.

To illustrate, Triple DES has three keying options (table below):

	Common Names	Key Size	Key Format	FIPS Status	Notes
Keying Option 1	3-Key Triple DES or 3DES / DES3	24-byte DES3 key (3 x 8-byte keys)	K1≠K2≠K3 3 keys, all unique	Approved	
Keying Option 2	2-Key Triple DES or 2DES / DES2	16-byte DES2 key (2 x 8-byte keys)	K1≠K2 2 keys, K1 is reused for K3	Legacy as of 1 Jan 2016	Restricted in FW 6.22.0
		24-byte DES3 key (3 x 8-byte keys)	K1=K3≠K2 3 keys, with K1 and K3 identical		Restricted in FW 6.24.2
Keying Option 3	1-Key Triple DES or DES	24-byte DES3 key (3 x 8-byte keys)	K1=K2=K3 3 keys, all identical	Disallowed	
		8-byte DES key	K1 1 key, K1 is reused for K2 and K3		

Note: The FIPS Status and Notes columns in the above table refer to the HSM when it is in FIPS mode.



Only when the HSM is **not** in FIPS mode, can the 2-key and the non-unique 3-key DES3 variants be used freely.

Note: These Triple DES restrictions are enforced by the HSM at release HSM 6.2.1 and firmware 6.24.2 and above; firmware 6.24.2 is currently (July 2016) on track to be the next FIPS-validation candidate.



The FIPS-validated version at this time is firmware 6.10.9 which was released before the SP 800-131A Revision 1 adjustment. Therefore, exclusion of 2-key, or 3-key non-unique, Triple DES is enforced by firmware only if you update to firmware versions shown in the table. If your HSM remains at version 6.10.9, then the SP800-131A revision 1 restriction must be enforced by your application if you wish to do so.

Other Effects

In addition to acceptable key sizes, some algorithms now limit the size of data that can be processed. For example, RSA sign/verify operations, even with sufficiently large key sizes selected, will not run if the input data chunk is too small, when FIPS mode is active. If using an application that is unaware of FIPS-mode limitations, you might encounter errors if you do not adjust the instructions. Using multitoken, as an example, allowing it to use its default data size of 16 bytes, you might see something like this:

```
C:\Program Files\SafeNet\LunaClient>multitoken.exe -mode rsasigver -key 2048 -slots 1
Initializing library...Finished Initializing
...done.
Do you wish to continue?
Enter 'y' or 'n': y
Constructing thread objects.
Logging in to tokens...
slot 2... Enter password:
Serial Number 151363
Please wait, creating test threads.
Error 0x21 (CKR_DATA_LEN_RANGE) on C_Sign
Aborting tests due to error 0x00000021 (CKR_DATA_LEN_RANGE) on thread 0, slot 1, serial number
150022!
Waiting for threads to terminate.
```

You would correct by including the additional parameter "-packet 32" in the command.

```
C:\Program Files\SafeNet\LunaClient>multitoken -mode rsasigver -key 2048 -slots 1 -packet 32
Initializing library...Finished Initializing
...done.

Do you wish to continue?

Enter 'y' or 'n': y

Constructing thread objects.
Logging in to tokens...
slot 1... Enter password: *****
Serial Number 150022
```

Please wait, creating test threads.

Test threads created successfully. Press ENTER to terminate testing.

RSA sign/verify 2048-bit : (packet size = 32 bytes)

		operations/second		elapsed
1, 0	total	average	time (secs)	
-----	-----	-----	-----	-----
111.2	111.2	111.259*		45
111.2	111.2	111.253*		50

Waiting for threads to terminate.

C:\Program Files\SafeNet\LunaClient>

Modification to DES3 Algorithm for NIST Compliance (2015)

In accordance with NIST document SP 800-131A Revision 1, when the HSM is in FIPS mode, two-key DES3 is now restricted to legacy operations (Decryption, Unwrapping, and CMAC verification). All other operations for DES3 must use the three-key variant.

If you are still using Two-key Triple DES, we suggest that you begin adapting your operational work-flow for the following changes that are in effect as of year 2015.

- Encryption, Disallowed
- Decryption, Legacy
- Wrapping, Disallowed
- Unwrapping, Legacy
- CMAC Sign, Disallowed
- CMAC Verification, Legacy

UPDATE Affecting Triple DES (SP 800-67)

SP 800-67 specifies the TDES standard and is static. SP 800-131A is a living document, updated frequently, that places additional requirements on existing standards as the crypto and threat environments evolve.

A late 2015 change in 131A is implemented in SafeNet General Purpose HSM firmware 6.24.2 as of July 2016.

Please refer to "[NIST SP 800-131A: Changes to FIPS-Supported Algorithms](#)" on page 377 above for details.

.

Common Criteria

If you are concerned with Common Criteria validation for products that you use, this section has some information that might be useful in your decisions and planning.

Background

Common Criteria is an international standard for computer security certification that is becoming important worldwide (<http://www.commoncriteriaportal.org>). CC includes categories that are applicable to SafeNet HSM products. The process to have a product validated against CC rules is lengthy and expensive, so we have tended to submit our products when they are mature and when there is a demand.

The process is quite different from US government's FIPS validation, but from the perspective of SafeNet products it has this in common: the unit that is validated is the HSM. That is, a SafeNet Network HSM (or other) appliance would not receive FIPS or CC validation - rather the HSM card that is the core of the appliance, and is used in several products, is what actually gets validated, and only for a particular firmware version. In FIPS evaluation, the HSM card at a certain firmware version is validated. In CC evaluation, the HSM card, as it is used in the SafeNet appliance is evaluated, a subtle difference.

Due to market requirements, the K5 generation (with firmware 4.6.1) as used in SafeNet Network HSM 4.x was CC evaluated and achieved certification. You can check the Common Criteria portal (see link above), or contact SafeNet to find out the most current CC status of any of our products. The K5 is used in the 4.x series 1U rack-mount-format SafeNet Network HSM, and is also the core of some other SafeNet HSM products.

The Common Criteria evaluation process has been started for the **K6** HSM. The earliest expected result would be some time in 2014. From time to time, you can check with the CC site, or with your SafeNet representative to learn the status of SafeNet products in various evaluations.

Trade-offs

As a general practice, when a product receives one of the major validations, and our customers invest their resources in that version of the product, we try to keep supporting that version even while the "state of the art" advances. Thus, as the SafeNet Network HSM product advances (newer software and firmware versions being released) it would be our general practice to make available (sell) the "frozen" CC'd version as long as possible for customers who wish to match units that they already own, while also offering the newer versions for those customers who did not require CC compliance.

What you might notice is that - when possible - a newer release of software is made compatible with an older milestone release of firmware (such as the Common Criteria-validated or FIPS-validated version). Features or fixes to the overall system can take the form of software-only, firmware-only, and combinations that involve changes to both software and firmware to achieve the fix or the new functionality. Thus, it becomes apparent that, if you need to retain strict Common Criteria compliance, you cannot take advantage of any fix or any functional improvement (feature) that has a firmware component to it [since it is the specific hardware and firmware combination that is evaluated and certified].

If you choose to use newer software (Client, Appliance, or both) while retaining an older firmware, be aware that only those new features (and fixes) that are implemented entirely in software would become available to you. Any newer product features that depended on newer firmware could not be accessed without updating the firmware, which would then invalidate that HSM from its strict compliance with Common Criteria - it would no longer be the exact version that was validated. One example is when new encryption algorithms are added, they are implemented in firmware. The software might allow you to call for a new algorithm, but if you have retained older firmware that didn't include such an algorithm, the response (naturally) would be an error message, such as "Mechanism Invalid".



Note: Due to the way that CC rules work, a validated product must be shipped from the factory. If you already own a SafeNet appliance that has the proper hardware, you cannot simply apply an upgrade/update and achieve CC compliance. Naturally, any competing product faces the same constraints.

So, What Are the Options?

If you absolutely need CC-validated equipment, because your own organization's rules require it, or because your customers require it, then you should purchase a CC-validated version.

If you simply prefer the peace-of-mind associated with the CC blessing - unbiased third-party confirmation that we have conformed to certain rigorous standards with our product - but you also require the newer features or algorithms, consider the following possibility. You might accept that there is some "halo" effect attached to our other products, especially in the same product lines that have been submitted and validated, because we follow the same procedures in our design, testing, sourcing, manufacturing, and other handling for all our HSM products that we do for those that go into the Common Criteria submission pipeline.

In other words, we believe that any given product that we make is likely to meet a CC standard if submitted because we make them all that way. Then we select the one(s) that we believe are early enough in their life-window and their customer appeal to be worth submitting to the year(s?)-long evaluation process.