

# SafeNet Luna Network HSM 7.0

Utilities Reference Guide

## Document Information

<b>Product Version</b>	7.0
<b>Document Part Number</b>	007-013576-002
<b>Release Date</b>	02 June 2017

## Revision History

<b>Revision</b>	<b>Date</b>	<b>Reason</b>
Rev. A	02 June 2017	Initial release.

## Trademarks, Copyrights, and Third-Party Software

Copyright 2001-2017 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.

**Table 1: Third-party software used in this product**

<b>Software</b>	<b>License and copyright</b>
editline	This product incorporates editline licensed under Apache v2.0 Open Software. Copyright 1992, 1993 Simmule Turner and Rich Salz. All rights reserved. You can obtain the full text of the Apache v2.0 Open Software license at the following URL: <a href="https://www.apache.org/licenses/LICENSE-2.0">https://www.apache.org/licenses/LICENSE-2.0</a>
libFDT	Dual License Choice of BSD or GPL-2.0 Copyright (C) 2006 David Gibson, IBM Corporation.
libsodium	ISC License (ISCL) Copyright (C) 2013-2016
Linux Kernel	GPL-2.0
OpenSSH	This product uses a derived version of OpenSSH Copyright 1995 Tatu Ylonen , Espoo, Finland. All rights reserved . Copyright 1995, 1996 by David Mazieres . Copyright 1983, 1990, 1992, 1993, 1995 The Regents of the University of California. All rights reserved You can obtain the full text of the OpenSSH license at the following URL: <a href="https://www.openbsd.org/policy.html">https://www.openbsd.org/policy.html</a>
OpenSSL	SSLeay License Copyright (C) 1995-1998 Eric Young (eay@cryptsoft.com) OpenSSL license

Software	License and copyright
	Copyright (C) 1998-2002 The OpenSSL Project
Software implementation of SHA2	Proprietary license Copyright (C) 2002, Dr Brian Gladman, Worcester, UK.
Software implementation of AES	Proprietary license Copyright (C) 2001, Dr Brian Gladman <brg@gladman.uk.net>, Worcester, UK.

## 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 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.

### 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 2014/30/EU. This product satisfies the CLASS B limits of EN55032.

# CONTENTS

<b>PREFACE</b>	<b>About the Utilities Reference Guide</b>	<b>10</b>
Customer Release Notes		10
Audience		10
Document Conventions		11
Notes		11
Cautions		11
Warnings		11
Command Syntax and Typeface Conventions		11
Support Contacts		12
<b>1</b>	<b>Certificate Management Utility (CMU)</b>	<b>13</b>
About the CMU Functions		13
Authentication		14
cmu certify		15
cmu delete		17
cmu export		18
cmu generatekeypair		19
cmu getattribute		23
cmu getpkc		24
cmu import		25
cmu importkey		26
cmu list		28
cmu requestcertificate		30
cmu selfsigncertificate		33
cmu setattribute		36
cmu verifypkc		38
<b>2</b>	<b>CKdemo</b>	<b>39</b>
Accessing the ckdemo Utility		39
Using the ckdemo Menu		40
Executing a Menu Function		41
The TOKEN Menu Functions		42
(1) Open Session		42
(2) Close Session		42
(3) Login		42
(4) Logout		42
(5) Change PIN		42
(6) Init Token		42
(7) Init PIN		43
(8) Mechanism List		43
(9) Mechanism Info		43
(10) Get Info		43
(11) Slot Info		43

12) Token Info .....	43
13) Session Info .....	43
14) Get Slot List .....	44
15) Wait for Slot Event .....	44
18) Factory Reset .....	44
19) Clone MofN .....	44
The OBJECT MANAGEMENT Menu Functions .....	45
(20) Create Object .....	45
(21) Copy Object .....	45
(22) Destroy Object .....	45
(23) Object Size .....	45
(24) Get Attribute .....	45
(25) Set Attribute .....	45
(26) Find Objects .....	45
(27) Display Object .....	46
(30) Modify Usage Count .....	46
(31) Destroy Multiple Objects .....	46
(32) Extract Public Key .....	46
The SECURITY Menu Functions .....	46
(40) Encrypt File .....	46
(41) Decrypt File .....	46
(42) Sign .....	46
(43) Verify .....	47
(44) Hash .....	47
(45) Simple Generate Key .....	47
(46) Digest Key .....	47
The HIGH AVAILABILITY RECOVERY Menu Functions .....	47
(50) HA Init .....	47
(51) HA Login .....	47
(52) HA Status .....	47
The KEY Menu Functions .....	48
60) Wrap Key .....	48
61) Unwrap Key .....	48
62) Generate Random Number .....	48
63) Derive Key .....	48
64) PBE Key Generation .....	48
65) Create Known Keys .....	48
66) Seed RNG .....	48
67) EC User Defined Curves .....	49
The CA Menu Functions .....	49
(70) Set Domain .....	49
(71) Clone Key .....	49
(72) Set MofN .....	49
(73) Generate MofN .....	49
(74) Activate MofN .....	49
(75) Generate Token Keys .....	49
(76) Get Token Cert .....	50
(77) Sign Token Cert .....	50
(78) Generate CertCo Cert .....	50

(79) Modify MofN .....	50
(86) Dup. MofN Keys .....	50
(87) Deactivate MofN .....	50
(88) Get Token Certificates .....	50
(112) Set Legacy Cloning Domain .....	50
The OTHERS Menu Functions .....	50
90) Self Test .....	51
94) Open Access .....	51
95) Close Access .....	51
97) Set App ID .....	51
98) Options .....	51
100) LKM Commands .....	52
The OFFBOARD KEY STORAGE Menu Functions .....	52
(101) Extract Masked Object .....	52
(102) Insert Masked Object .....	52
(103) Multisign With Value .....	53
(104) Clone Object .....	53
(105) SIMExtract .....	53
(106) SIMInsert .....	53
(107) SimMultiSign .....	53
(118) Extract Object .....	53
(119) Insert Object .....	53
The SCRIPT EXECUTION Menu Functions .....	53
The CLUSTER EXECUTION Menu Functions .....	53
The PED INFO Menu Functions .....	53
120) Set PED Info .....	54
121) Get PED Info .....	54
122) Init RPV .....	54
123) Delete RPV .....	54
The AUDIT/LOG Menu Functions .....	54
(130) Get Config .....	54
(131) Set Config .....	54
(132) Verify Logs .....	54
(133) Get Time .....	54
(134) Set Time .....	54
(135) Import Secret .....	55
(136) Export Secret .....	55
(137) Init Audit .....	55
(138) Get Status .....	55
(139) Log External .....	55
The SRK Menu Functions .....	55
(200) SRK Get State .....	55
(201) SRK Restore .....	55
(202) SRK Resplit .....	55
(203) SRK Zeroize .....	55
(204) SRK Enable/Disable .....	56
The POLICY Menu Functions .....	56
(53) Show Partition Policies .....	56
(54) Set Partition Policies .....	56

(55) Show HSM Policies .....	56
(56) Set HSM Policies .....	56
(57) Set Destructive HSM Policies .....	56
<b>3 Multitoken .....</b>	<b>57</b>
Accessing Multitoken .....	57
Using Multitoken .....	57
Operating Modes .....	59
Named and User-defined Curves .....	63
<b>4 Remote Backup Service (RBS) .....</b>	<b>66</b>
RBS Overview .....	66
rbs .....	67
rbs config .....	68
rbs daemon .....	69
rbs genkey .....	70
rbs nopassword .....	71
<b>5 SLogin .....</b>	<b>72</b>
Using the salogin Utility .....	72
The salogin Command .....	72
Examples .....	73
Other options .....	74
<b>6 SCP and PSCP .....</b>	<b>75</b>
Using the scp and pscp Utilities .....	75
<b>7 VTL .....</b>	<b>77</b>
VTL Overview .....	78
vtl addServer .....	80
vtl backup .....	81
vtl backup append .....	83
vtl backup delete .....	85
vtl backup restore .....	86
vtl backup token .....	87
vtl backup token factoryreset .....	88
vtl backup token init .....	89
vtl backup token resize .....	90
vtl backup token show .....	91
vtl backup token show licenses .....	92
vtl backup token update .....	93
vtl cklogsupport .....	94
vtl createCert .....	95
vtl deleteServer .....	97
vtl examineCert .....	98
vtl fingerprint .....	100
vtl haAdmin .....	101
vtl haAdmin addMember .....	102



---

vtl haAdmin autoRecovery .....	104
vtl haAdmin deleteGroup .....	105
vtl haAdmin haLog .....	107
vtl haAdmin HAOnly Client .....	109
vtl haAdmin HAOnly disable .....	110
vtl haAdmin HAOnly enable .....	111
vtl haAdmin HAOnly show .....	112
vtl haAdmin newGroup .....	113
vtl haAdmin recover group .....	115
vtl haAdmin removeMember .....	116
vtl haAdmin show .....	117
vtl haAdmin standbyMembers .....	118
vtl haAdmin standbyMembers -remove .....	119
vtl haAdmin standbyMembers -set .....	120
vtl haAdmin synchronize .....	121
vtl listServers .....	122
vtl listSlots .....	123
vtl logging configure .....	124
vtl logging show .....	125
vtl replaceserver .....	126
vtl supportInfo .....	127
vtl verify .....	128

# PREFACE

## About the Utilities Reference Guide

This document describes how to use the various utilities included with the SafeNet Luna HSM client. It contains the following chapters:

- "Certificate Management Utility (CMU)" on page 13
- "CKdemo" on page 39
- "Multitoken" on page 57
- "Remote Backup Service (RBS)" on page 66
- "SALogin" on page 72
- "SCP and PSCP" on page 75
- "VTL" on page 77

This preface also includes the following information about this document:

- "Customer Release Notes" below
- "Audience" below
- "Document Conventions" on the next page
- "Support Contacts" on page 12

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 from the Technical Support Customer Portal at <https://supportportal.gemalto.com>.

---

## Audience

This document is intended for personnel responsible for maintaining your organization's security infrastructure. This includes SafeNet Luna 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
<b>bold</b>	<p>The bold attribute is used to indicate the following:</p> <ul style="list-style-type: none"> <li>• Command-line commands and options (Type <b>dir /p</b>.)</li> <li>• Button names (Click <b>Save As</b>.)</li> <li>• Check box and radio button names (Select the <b>Print Duplex</b> check box.)</li> <li>• Dialog box titles (On the <b>Protect Document</b> dialog box, click <b>Yes</b>.)</li> <li>• Field names (<b>User Name</b>: Enter the name of the user.)</li> <li>• Menu names (On the <b>File</b> menu, click <b>Save</b>.) (Click <b>Menu &gt; Go To &gt; Folders</b>.)</li> <li>• User input (In the <b>Date</b> box, type <b>April 1</b>.)</li> </ul>
<i>italics</i>	<p>In type, the italic attribute is used for emphasis or to indicate a related document. (See the <i>Installation Guide</i> for more information.)</p>
<variable>	<p>In command descriptions, angle brackets represent variables. You must substitute a value for command line arguments that are enclosed in angle brackets.</p>

Format	Convention
[optional] [<optional>]	Represent optional <b>keywords</b> 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> <b> <c>}	Represent required alternate <b>keywords</b> 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> <b> <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	
<b>Phone</b> (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
<b>Web</b>	<a href="https://safenet.gemalto.com">https://safenet.gemalto.com</a>	
<b>Technical Support Customer Portal</b>	<a href="https://supportportal.gemalto.com">https://supportportal.gemalto.com</a> 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 <b>Register</b> link at the top of the page. You will need your Customer Identifier number.	

# Certificate Management Utility (CMU)

This chapter provides a detailed description of each of the functions available in the SafeNet Certificate Management Utility. It contains the following topics:

- "About the CMU Functions" below
- "cmu certify" on page 15
- "cmu delete" on page 17
- "cmu export" on page 18
- "cmu generatekeypair" on page 19
- "cmu getattribute" on page 23
- "cmu getpkc" on page 24
- "cmu import" on page 25
- "cmu importkey" on page 26
- "cmu list" on page 28
- "cmu requestcertificate" on page 30
- "cmu selfsigncertificate" on page 33
- "cmu setattribute" on page 36
- "cmu verifypkc" on page 38

## About the CMU Functions

---

This section provides a detailed description of each function available in the Certificate Management Utility.

The command function is the first parameter on the command line that invokes the CMU application. It does not require a leading dash character. All options follow the command function and do employ leading dashes. Only a single command function can be specified with each invocation of the CMU application.

```
cmu <function> <-parameter_name[=parameter_value]>
```

Most functions take parameters, some of which may be mandatory, and some optional. Parameters may, in turn, take values. If a parameter takes a value, then the general syntax is to write the command **cmu**, followed by a space, followed by a function name, followed by a space, followed by a leading dash "-" and parameter name and an equal sign "=" and a value, with no spaces from the dash to the end of the parameter value. Multiple parameters are separated by spaces.

## Authentication

Where an operation requires authentication, you must provide the appropriate password (for a Password Authenticated HSM) or the appropriate PED key (via Luna PED, for a Trusted Path HSM).

## cmu certify

This function creates an X.509 V3 certificate from a PKCS #10 certificate request. The parent certificate and corresponding private key must already exist on the token or HSM. The private key is located on the token using the public key info inside the parent certificate.

### Syntax

**cmu certify** <parameters>

### Mandatory Parameters

Parameter	Description
<b>-handle=&lt;handle#&gt;</b>	This is a mandatory parameter that defines the handle to parent certificate. If this parameter is omitted and there is only one certificate on the HSM, that certificate is automatically selected. If this parameter is omitted and there are multiple certificates on the HSM, the user is asked to select the certificate.
<b>-inputfile</b>	This parameter defines the name of the file that contains the PKCS #10 certificate request.
<b>-startDate</b>	This parameter defines the validity start of the certificate, in the format YYYYMMDD.
<b>-endDate</b>	This parameter defines the validity end of the certificate, in the format YYYYMMDD.
<b>-serialNumber</b>	This parameter defines the serial number of the certificate, in big-endian hexadecimal form.

### Optional Parameters

Parameter	Description
<b>-keyusage</b>	This is an optional parameter that defines the key usage extension for the certificate. It can be set to any of the following: digitalsignature, nonrepudiation, keyencipherment, dataencipherment, keyagreement, keycertsign, crlsign, encipheronly, decipheronly. This parameter may appear more than once in the parameter set to define multiple usages, or it can be used once with a comma separated list of usage types.
<b>-md5WithRsa</b>	This is an optional parameter that defines the signature algorithm for the certificate to be pkcs-1-MD5withRSAEncryption. The default is to use sha1WithRsa.
<b>-sha1WithRsa</b>	This is an optional parameter that defines the signature algorithm for the certificate to be pkcs-1-SHA1withRSAEncryption. The default is to use sha1WithRsa.
<b>-label</b>	This is an optional parameter that defines the label attribute for the certificate object that gets created on the HSM. If omitted, the common name of the subject DN is used instead.
<b>-sha224withrsa</b>	This is an optional parameter that defines the signature algorithm for the certificate to be pkcs-1-SHA224withRSAEncryption. The default is to use sha1WithRsa.

Parameter	Description
<b>-sha256withrsa</b>	This is an optional parameter that defines the signature algorithm for the certificate to be pkcs-1-SHA256withRSAEncryption. The default is to use sha1WithRsa.
<b>-sha384withrsa</b>	This is an optional parameter that defines the signature algorithm for the certificate to be pkcs-1-SHA384withRSAEncryption. The default is to use sha1WithRsa.
<b>-sha512withrsa</b>	This is an optional parameter that defines the signature algorithm for the certificate to be pkcs-1-SHA512withRSAEncryption. The default is to use sha1WithRsa.
<b>-sha1withdsa</b>	This is an optional parameter that defines the signature algorithm for the certificate to be pkcs-1-SHA1withDSAEncryption. The default is to use sha1WithRsa.
<b>-sha1withecdsa</b>	This is an optional parameter that defines the signature algorithm for the certificate to be pkcs-1-SHA1withECDSAEncryption. The default is to use sha1WithRsa.
<b>- sha224withecdsa</b>	This is an optional parameter that defines the signature algorithm for the certificate to be pkcs-1-SHA224withECDSAEncryption. The default is to use sha1WithRsa.
<b>-sha256withecdsa</b>	This is an optional parameter that defines the signature algorithm for the certificate to be pkcs-1-SHA256withECDSAEncryption. The default is to use sha1WithRsa.
<b>-sha384withecdsa</b>	This is an optional parameter that defines the signature algorithm for the certificate to be pkcs-1-SHA384withECDSAEncryption. The default is to use sha1WithRsa.
<b>- sha512withecdsa</b>	This is an optional parameter that defines the signature algorithm for the certificate to be pkcs-1-SHA512withECDSAEncryption. The default is to use sha1WithRsa.
<b>-id</b>	This optional parameter defines the CKA_ID attribute for the certificate object that gets created on the HSM. If omitted, the CKA_ID attribute of the private key is used instead.
<b>-certificatepolicy</b>	This optional parameter defines the certificate policy to be used.
<b>-keyids</b>	This optional parameter indicates to use a subject key identifier from the parent. Set to True or False (or 1 or 0).

## Example

```
cmu certify -input=testCert.req -h=8
```

Create and sign a new certificate using certificate 8 as the parent.



## cmu delete

This function deletes a key, certificate, or generic data object on the token. A confirmation message is presented to the user, describing the class and label of the object about to be deleted.

### Syntax

**cmu delete** <parameters>

### Required Parameters

Parameter	Description
<b>-handle</b> =<handle#>	The handle of the object to be deleted. The parameter "-handle" is followed by an equal sign "=", followed by the handle of the object (no spaces).

### Optional Parameters

Parameter	Description
<b>-force</b>	This optional parameter can be used to suppress the user confirmation step.

### Example

The following command deletes the key or certificate referenced by object handle 14 without a request for confirmation of the delete operation:

```
cmu delete -handle=14 -force
```

The following command queries the user for a handle of an object to delete. The user is asked to confirm the deletion operation:

```
cmu delete
```

## cmu export

This function exports an X.509 certificate from the token or HSM to a file. The supported formats are Raw (binary) and PEM (base 64 encoding).

### Syntax

**cmu export**

### Required Parameters

Parameter	Description
<b>-handle=&lt;handle#&gt;</b>	The handle to the X.509 certificate to be exported from the HSM to a file. The parameter "-handle" is followed by an equal sign "=", followed by the handle of the object (no spaces).
<b>-outputfile</b>	This mandatory parameter defines the name of the file that receives the certificate.

### Optional Parameters

Parameter	Description
<b>-binary</b>	This is an optional parameter that defines the exported certificate format to be raw binary instead of the default PEM (base64) encoding.

### Example

The following command outputs the certificate with handle 7 to file test.cer in PEM format:

```
cmu export -handle=7 -output=test.cer
```

## cmu generatekeypair

This function generates an asymmetric key pair on the token or HSM. An optional input filename can be used to specify a file from which mandatory and optional attributes are to be read.

For DSA key generation, the domain parameters (Prime, Subprime, and Base) are required, and must be provided either as part of the command, or as responses to interactive prompting. If one is provided at the command line, then all three must be provided in that manner. If none are provided at the command line, then all three are prompted for interactive entry.

You may not provide only one or two of the parameters at the command line. Providing just one or two domain parameters is considered an error, and the command halts with an error message.

### Syntax

**cmu generatekeypair <parameters>**

### Required Parameters

Parameter	Description
<b>-modulusBits</b>	This mandatory parameter defines the length in bits of the modulus value for the generation of RSA key pairs. It must be set to a value between 1024 and 4096 that is a multiple of 64 bits. If the HSM policy 12 "Allow non-FIPS algorithms" is set to "No", then RSA key size is limited to 2048 bits or 3072 bits.
<b>-publicExponent</b>	This mandatory parameter defines the public exponent value to use in the generation of RSA key pairs. It must be set to a value of 3, 17 or 65537.

### Optional Parameters

Parameter	Description
<b>-binary</b>	This is an optional parameter that defines the exported certificate format to be raw binary instead of the default PEM (base64) encoding.
<b>-inputFile</b>	This optional parameter defines the name of a file from which to obtain additional parameter settings, one per line, of the form name=value.
<b>-keyType</b>	This optional parameter defines the type of asymmetric keys to generate. This parameter is not required if the key type can be established by the presence of other parameters. (e.g. If modulusBits and/or publicExponent parameters are present, then -keyType=RSA is redundant). Currently, only RSA key pairs are supported.
<b>-label</b>	This optional parameter defines a label to be applied to both of the newly generated keys. If a multiple word label is required, the label value must be enclosed within quotation marks.
<b>-labelPublic</b>	This optional parameter defines a label to apply to the public key from the newly generated key pair.

Parameter	Description
<b>-labelPrivate</b>	This optional parameter defines a label to apply to the private key from the newly generated key pair.
<b>-modifiable</b>	This optional parameter defines the modifiable setting for each of the keys in the key pair. It must be set to True or False (or 1 or 0).
<b>-id</b>	This optional parameter defines the ID field for the newly generated keys. It must be set to a big-endian hexadecimal integer value.
<b>-startDate</b>	This optional parameter defines the startDate field for the newly generated keys. The format for the value is YYYYMMDD.
<b>-endDate</b>	This optional parameter defines the endDate field for the newly generated keys. The format for the value is YYYYMMDD.
<b>-subject</b>	This optional parameter defines the subject field for the newly generated keys. It must be set to a big-endian hexadecimal integer value. The subject field is typically set to the DER encoding of the subject distinguished name for the key.
<b>-encrypt</b>	This optional parameter defines the encrypt setting for the public key in the newly generated key pair. It must be set to True or False (or 1 or 0), with False being the default. If this parameter is set to True, then the decrypt setting for the private key should also be set to True. Note that an HSM is often configured such that no key can have multiple functions. Thus if encrypt is set True, then wrap and verify need to be False.
<b>-decrypt</b>	This optional parameter defines the decrypt setting for the private key in the newly generated key pair. It must be set to True or False (or 1 or 0), with False being the default. If this parameter is set to True, then the encrypt setting for the public key should also be set to True. Note that an HSM is often configured such that no key can have multiple functions. Thus if decrypt is set True, then unwrap and sign need to be False.
<b>-sign</b>	This optional parameter defines the sign setting for the private key in the newly generated key pair. It must be set to True or False (or 1 or 0), with False being the default. If this parameter is set to True, then the verify setting for the public key should also be set to True. Note that an HSM is often configured such that no key can have multiple functions. Thus if sign is set True, then unwrap and decrypt need to be False.
<b>-verify</b>	This optional parameter defines the verify setting for the public key in the newly generated key pair. It must be set to True or False (or 1 or 0), with False being the default. If this parameter is set to True, then the sign setting for the private key should also be set to True. Note that an HSM is often configured such that no key can have multiple functions. Thus if verify is set True, then encrypt and wrap need to be False.
<b>-wrap</b>	This optional parameter defines the wrap setting for the public key in the newly generated key pair. It must be set to True or False (or 1 or 0), with False being the default. If this parameter is set to True, then the unwrap setting for the private key should also be set to True. Note that an HSM is often configured such that no key can have multiple functions. Thus if wrap is set True, then encrypt and verify need to be False.

Parameter	Description
<b>-unwrap</b>	This optional parameter defines the unwrap setting for the private key in the newly generated key pair. It must be set to True or False (or 1 or 0), with False being the default. If this parameter is set to True, then the wrap setting for the public key should also be set to True. Note that an HSM is often configured such that no key can have multiple functions. Thus if unwrap is set True, then decrypt and sign need to be False.
<b>-extractable</b>	This optional parameter defines the extractable setting for the private key in the newly generated key pair. It must be set to True or False (or 1 or 0), with False being the default.
<b>-curvetype</b>	This optional parameter defines the name of a curve type for ECDSA keys. Enter values 1-5 (1-NISTP 192 / 2-NISTP 224 / 3-NISTP 256 / 4-NISTP 384 / 5-NISTP 521).
<b>-prime</b>	This optional parameter defines a prime length for DSA key generation.
<b>-subprime</b>	This optional parameter defines a subprime bits length for DSA key generation.
<b>-base</b>	This optional parameter defines a base length for DSA key generation.

## Example

### RSA

```
C:\Program Files\SafeNet\LunaClient>cmu gen -modulusBits=2048 -publicExp=65537 -sign=T -verify=T
Select token
[1] Token Label: myPartition1
[2] Token Label: myPartition1
Enter choice: 2
Please enter password for token in slot 2 : *****
C:\Program Files\SafeNet\LunaClient>cmu list
Select token
[1] Token Label: myPartition1
[2] Token Label: myPartition1
Enter choice: 2
Please enter password for token in slot 2 : *****
handle=14          label=NewPublicVerifyingKey
handle=15          label=NewPrivateSigningKey
C:\Program Files\SafeNet\LunaClient>
```

### DSA - Domain Parameters at Command Line

```
cmu generatekeypair -keytype DSA -slot 6 -prime 0xfcec6182e-
b206b43c03e36c0ead-
abff56a0c2e79def44bc8f2e53699096d1ff270f159785d756921dbff9773ae08483b662fc07df7512ff68b2e5565fd7982e20c244
-subprime 0xd3807353b51c5f71b22ac3d0c7e394148fcedc61 -base
0x42e3778e6ec31b0d-
b07a6b370d7f-
b6fb4a0bca6deaac371f6adbcbeba38ddf76a47c3c3d79276a0e579ce4e347180fd9b4ad461d6cf0eac51fb08cf452f624570051e5
```

## DSA - Domain Parameters Entered Interactively

```
cmu generatekeypair -keytype DSA -slot 6
```

The prime, subprime and base values must be entered as a HEX byte array.

For example, to enter a 1024-bit prime value, enter a 128-byte HEX byte array using this format:  
0xa0383ee692f8...

The prime value can be a 1024-bit, 2048-bit or 3072-bit value.

Enter a prime value: 0xfceec6182e-

```
b206b43c03e36c0eadabff56a0c2e79def44bc8f2e53699096dlff270f159785d7 56921d-
```

```
bff9773ae08483b662fc07df7512ff68b2e5565fd7982e20c244832aba121cc0799cc09f2d5414d5f3966211365f
```

```
51b83e9ffcccb3d88cdf238f7c2739131ca7aadff662fec1fb0e1d311a404260376fd011fe00d0204c3
```

Enter a 160 bit subprime value: 0xd3807353b51c5f71b22ac3d0c7e394148fcedc61

Enter a 1024-bit base value: 0x42e3778e6ec31b0d-

```
b07a6b370d7fb6fb4a0bca6deaac371f6adbcbeba38ddf76a47 c3c3d79276a0e579ce4e347180f-
```

```
d9b4ad461d6cf0eac51fb08cf452f624570051e518a75a5bb9c3578a14fd4f27f795b22
```

```
acea62b1fdf1032c1266da081c7fb99c4266626587093fd381617238ee1578fc325548dc1c08e5f9322c3b1205e
```

## cmu getattribute

This function outputs any viewable attributes for an object. An optional output filename can be used to direct the output to a file.

### Syntax

`cmu getAttribute <parameters>`

### Required Parameters

Parameter	Description
<code>-handle=&lt;handle#&gt;</code>	The handle to the object. The parameter "-handle" is followed by an equal sign "=", followed by the handle of the object (no spaces). If this parameter is omitted and there is only one object on the HSM, that object is automatically selected. If this parameter is omitted and there are multiple objects on the HSM, you are prompted to select the object.

### Optional Parameters

Parameter	Description
<code>-attributes</code>	This optional parameter lists the attributes to be displayed for the object as a comma separated list. Multiple instances of this option can also be used to define multiple attributes. If this parameter is omitted, all viewable attributes are displayed.
<code>-outputFile</code>	This optional parameter defines the filename to which the attribute set is written. If this parameter is omitted, the attribute set is written to the display.

### Example

The following command outputs all of the viewable attributes for the object with handle 46:

```
cmu getAttribute -handle=46
```

The following command outputs the label, public exponent and modulus of key 9 to file **keydata.txt**:

```
cmu getAttribute -handle=9 -attribute=label,publicExponent,modulus -outputFile=keydata.txt
```

## cmu getpkc

Retrieve a Public Key Confirmation from the HSM.

### Syntax

**cmu getpkc**

### Optional Parameters

Parameter	Description
<b>-handle=&lt;handle#&gt;</b>	The handle to the corresponding private key for the PKC. The parameter "-handle" is followed by an equal sign "=", followed by the handle of the object (no spaces).
<b>-outputfile</b>	This mandatory parameter defines the name of the file that receives the PKC.
<b>-pkctype</b>	This mandatory parameter defines the PKC type (1 - TC-TrustCenter, 2 - Chrysalis-ITS).
<b>-verify</b>	This optional parameter sets a flag to verify the PKC against the certificate that signed the PKC. It must be set to True or False (or 1 or 0), with False being the default.

### Example

```
cmu getpkc -handle=5 -pkctype=1
```



## cmu import

This function imports X.509 certificates from a file to the token or HSM. The file may include a single DER encoded binary certificate or a CMSS PKCS #7 certificate or certificate set. Either type of certificate can be binary or PEM (base 64) encoded. An optional label can be defined as a function parameter. If omitted, the common name of the certificate subject is chosen as the label.

### Syntax

**cmu import**

### Required Parameters

Parameter	Description
<b>-inputFile</b>	This parameter defines the name of the file containing the certificate to import.

### Optional Parameters

Parameter	Description
<b>-label</b>	This parameter defines a label to apply to the imported certificate. If no label is defined, the Common Name portion of the certificate Subject distinguished name is used instead.

### Example

The following example inputs the certificate in **test.cer**

## cmu importkey

This function unwraps an RSA, DSA, or ECDSA private key onto the selected token or HSM. The key file may be in any of the following formats:

- PKCS #12(PFX) RSA in a DER-encoded format (.pfx file)
- PKCS #8(Unencrypted PrivatekeyInfo) in RSA or DSA in base 64 PEM, or binary DER format
- PKCS #1 (RSA in base64 PEM, or binary DER) format.

### Syntax

**cmu importkey** <parameters>

### Required Parameters

Parameter	Description
<b>-in (Filename)</b>	This parameter defines the full path to the file containing the key to import.
<b>-keyalg (DSA RSA ECDSA)</b>	Specifies the key's algorithm.

### Optional Parameters

Parameter	Description
<b>-PKCS8</b>	Indicates that the key to import is formatted according to the PKCS#8 standard.
<b>-PKCS12</b>	Indicates that the key to import is formatted according to the PKCS#12 standard. *Note that only the private key portion is unwrapped onto the token. Any certificates in this file are simply ignored. It is assumed that you properly export a PKCS #12 key from Windows keystore (or other source, as appropriate).
<b>-wrapkey (handle)</b>	The handle of the existing key that is to be used as the wrapping key. *Note that this key must have the CKA_WRAP attribute set to true. If this flag is not specified the default behaviour is to autogenerate a 3DES key for the sole purpose of unwrapping the key onto the HSM.
<b>-setkeyattr</b>	Allows the user to manually enter the imported key's attributes. Modifiable key attributes are CKA_DECRYPT, CKA_SIGN, CKA_EXTRACTABLE, and CKA_UNWRAP. The defaults are always 1=true.

### Example

```
cmu importkey -in rawrsa1028.pem -keyalg RSA -wrapkey 11 -setkeyattr
```

```
cmu importkey -pkcs8 -in pk8privkey.pem -keyalg DSA-keyalg DSA
```

```
cmu importkey -in rsakey.pem -keyalg RSA -wrapkey 11
```

```
cmu importkey -in rsakey.pem -keyalg RSA
```

```
cmu importkey -PKCS12 -in p12.pfx -keyalg RSA
```

## Notes

1. Ideally the private key should be in PKCS#8 format (privatekeyinfo) and not encrypted.  
To convert a private key of either RSA or DSA type: (see PKCS#1 for RSA and PKCS#11 (11.9) for DSA) into a PKCS#8 structure, use the following openssl command:  
`openssl pkcs8 -in key.pem -topk8 -nocrypt -out noenckey.pem`
2. In the option to the command, the "PKCS" should be in all uppercase letters, as "cmu importkey -PKCS8" or "cmu importkey -PKCS12".
3. If the PKCS#8 structure is already encrypted according to the PKCS#5-PBE standard, then to import via CMU, use the following command:  
`openssl pkcs8 -in pk8.pem -out key.pem`  
\*You will be prompted for the password to decrypt the PrivateKeyInfo.
4. You can export the PrivatekeyInfo contents of a .pfx file by using the following openssl command: `openssl pkcs12 -in p12.pfx -out pk12_privkey.pem -nocerts -nodes`  
\*You will be prompted for the password to decrypt the PrivateKeyInfo.

## cmu list

This function lists all objects (keys, certificates and other general data objects) on the HSM that match an optional set of search criteria and that are accessible given the authentication state of the HSM. Search criteria can include many of the object attributes that are available for searching via the PKCS #11 API. If no search criteria are defined, all accessible objects are returned. The content of the entries in the returned list is definable and can include the object handle and/or any combination of viewable object attributes. The default is to include the handle and the label (CKA\_LABEL).

### Syntax

**cmu list** <parameters>

### Required Parameters

None

### Optional Parameters

Parameter	Description
<b>-display</b>	This is a comma-separated list of attributes to be displayed for each returned object in the list. Multiple attributes can also be specified by repeated use of the display option instead of using the comma-separated list. The attributes supported with the display option are index, handle, class, keyType, label and value. If this parameter is omitted, only the handle and the label are displayed.
<b>-class</b>	This option defines the class of object to list. It can be set to any of data, certificate, public, private and secret.
<b>-keyType</b>	This option specifies the type of keys to list. It can be set to any of rsa, dsa, dh, des, 2des, 3des, rc2, rc4, rc5, cast3, cast5 and generic.
<b>-certificateType</b>	This option specifies the type of certificate to list. It can only be set to x.509 if used.
<b>-label</b>	This option specifies the label that objects must match in order to be listed.
<b>-application</b>	This option specifies the application attribute that objects must match in order to be listed.
<b>-value</b>	This option specifies the value that objects must match in order to be listed.
<b>-issuer</b>	This option specifies the issuer that objects must match in order to be listed.
<b>-serialNumber</b>	This option specifies the serial number that objects must match in order to be listed.
<b>-subject</b>	This option specifies the subject that objects must match in order to be listed.
<b>-id</b>	This option specifies the id that objects must match in order to be listed.
<b>-token</b>	This option specifies whether permanent or temporary objects are to be listed. It can be set to True or 1 for permanent objects and False or 0 for temporary objects.

Parameter	Description
<b>-private</b>	Set to True or False (or 1 or 0).
<b>-sensitive</b>	Set to True or False (or 1 or 0).
<b>-alwaysSensitive</b>	Set to True or False (or 1 or 0).
<b>-extractable</b>	Set to True or False (or 1 or 0).
<b>-neverExtractable</b>	Set to True or False (or 1 or 0).
<b>-local</b>	Set to True or False (or 1 or 0).
<b>-encrypt</b>	Set to True or False (or 1 or 0).
<b>-decrypt</b>	Set to True or False (or 1 or 0).
<b>-sign</b>	Set to True or False (or 1 or 0).
<b>-verify</b>	Set to True or False (or 1 or 0).
<b>-wrap</b>	Set to True or False (or 1 or 0).
<b>-unwrap</b>	Set to True or False (or 1 or 0).
<b>-derive</b>	Set to True or False (or 1 or 0).
<b>startDate</b>	This option specifies the start date that objects must match in order to be listed.
<b>endDate</b>	This option specifies the end date that objects must match in order to be listed.
<b>modulusBits</b>	This option specifies the modulus size that RSA keys must match in order to be listed.
<b>publicExponent</b>	This option specifies the public exponent value that RSA keys must match in order to be listed. It can only be set to 3, 17 or 65537.
<b>- modifiable</b>	Set to True or False (or 1 or 0).

## Example

The following example displays the handle and label of each certificate that is accessible on the HSM:

```
cmu list -class=certificate
```

The following example displays the handles of all locally generated RSA private signing keys on the HSM:

```
cmu list -keyType=rsa -local=True -sign=True -display=handle
```

The following example displays the class, type and label of all signing keys on the HSM:

```
cmu list -display=class,keyType,label -sign=True
```

## cmu requestcertificate

This function creates a PKCS #10 certificate request for an RSA/DSA/ECDSA key pair on the token or HSM. It must be provided with the handle either to the public key or to the corresponding private key (all of the public key components are contained within the private key). The private key must have Signing capability because it is used to sign the certificate request structure. The signature is done using any of the mechanisms listed below. The subject name is defined by a series of optional RDN components.

If none of these components are provided on the command line, the CKA\_SUBJECT of the private key is used as the subject of the certificate request. If the private key does not have its CKA\_SUBJECT attribute set, the user will be queried for each of the RDN components. The Subject DN should contain at least the country, organization and common name components.

The signed certificate request is output to the specified file.

### Syntax

**cmu requestCertificate** <parameters>

### Required Parameters

Parameter	Description
- <b>publichandle=</b> <pubkeyhandle#>	This is a mandatory parameter that defines the handle to the public key from an RSA key pair to be certified. If this parameter is omitted and there is only one public signing key on the HSM, that key is automatically selected. If this parameter is omitted and there are multiple public signing keys on the HSM, the user is asked to select the public signing key.
- <b>privatehandle</b> =<privkeyhandle#>	This is a mandatory parameter that defines the handle to the private key from an RSA key pair to be certified. If this parameter is omitted and there is only one private signing key on the HSM, that key is automatically selected. If this parameter is omitted and there are multiple private signing keys on the HSM, the user is asked to select the private signing key.
<b>-outputFile</b>	This mandatory parameter defines the file that receives the certificate request.

### Optional Parameters

Parameter	Description
<b>-binary</b>	This is an optional parameter that defines the certificate request format to be raw binary instead of the default PEM (base64) encoding.
<b>-sha1WithRsa</b>	This is an optional parameter that defines the signature algorithm for the certificate request to be pkcs-1-SHA1withRSAEncryption. The default is to use sha1WithRsa.
- <b>sha224withrsa</b>	This is an optional parameter that defines the signature algorithm for the certificate request to be pkcs-1-sha224withRSAEncryption. The default is to use sha1WithRsa.
- <b>sha256withrsa</b>	This is an optional parameter that defines the signature algorithm for the certificate

Parameter	Description
	request to be pkcs-1-sha256withRSAEncryption. The default is to use sha1WithRsa.
<b>- sha384withrsa</b>	This is an optional parameter that defines the signature algorithm for the certificate request to be pkcs-1-sha384withRSAEncryption. The default is to use sha1WithRsa.
<b>- sha512withrsa</b>	This is an optional parameter that defines the signature algorithm for the certificate request to be pkcs-1-sha512withRSAEncryption. The default is to use sha1WithRsa.
<b>- sha1withdsa</b>	This is an optional parameter that defines the signature algorithm for the certificate request to be pkcs-1-sha1withDSAEncryption. The default is to use sha1WithRsa.
<b>- sha1withecdsa</b>	This is an optional parameter that defines the signature algorithm for the certificate request to be pkcs-1-sha1withECDSAEncryption. The default is to use sha1WithRsa.
<b>- sha224withecdsa</b>	This is an optional parameter that defines the signature algorithm for the certificate request to be pkcs-1-sha224withECDSAEncryption. The default is to use sha1WithRsa.
<b>- sha256withecdsa</b>	This is an optional parameter that defines the signature algorithm for the certificate request to be pkcs-1-sha256withECDSAEncryption. The default is to use sha1WithRsa.
<b>- sha384withecdsa</b>	This is an optional parameter that defines the signature algorithm for the certificate request to be pkcs-1-sha384withECDSAEncryption. The default is to use sha1WithRsa.
<b>- sha512withecdsa</b>	This is an optional parameter that defines the signature algorithm for the certificate request to be pkcs-1-sha512withECDSAEncryption. The default is to use sha1WithRsa.
<b>-C</b>	This optional parameter defines the two-letter country name for the subject distinguished name (DN) of the certificate request. This parameter should be present in the subject DN.
<b>-S</b>	This optional parameter defines the state or province name for the subject distinguished name of the certificate request. This parameter may be present in the Subject DN.
<b>-L</b>	This optional parameter defines the locality (typically the city) for the subject distinguished name of the certificate request. This parameter may be present in the Subject DN.
<b>-O</b>	This optional parameter defines the organization name for the subject distinguished name (DN) of the certificate request. This parameter should be present in the subject DN.
<b>-OU</b>	This optional parameter defines the organization unit name for the subject distinguished name (DN) of the certificate request. This parameter may be present in the subject DN.
<b>-CN</b>	This optional parameter defines the common name for the subject distinguished name (DN) of the certificate request. This parameter should be present in the subject DN.

## Example

The following example creates a PEM encoded PKCS #10 certificate request for key 6:

```
cmu requestCert -publichandle=6 -privatehandle=7 -C=CA -L=Ottawa -O="Rainbow-Chrysalis" -  
CN="Test Certificate" -outputFile=testCert.req
```



## cmu selfsigncertificate

This function creates a self-signed X.509 certificate for an RSA, DSA, or ECDSA key pair on the token or HSM. It must be provided with the handles to both the public key and the corresponding private key (all of the public key components are contained within the private key). The private key must have Signing capability since it is used to sign the certificate request structure. The signature is done with any of the mechanisms listed below. The subject name is defined by a series of optional RDN components.

If none of these components are provided on the command line, the CKA\_SUBJECT of the private key is used as the subject of the certificate. If the private key does not have its CKA\_SUBJECT attribute set, the user will be queried for each of the RDN components. The Subject DN should contain at least the country, organization and common name components.

The certificate will, by default, have a keyUsage setting of keycertsign. The certificate is stored as a PKCS #11 certificate object on the token. The CKA\_ID attribute of the certificate is defined by an optional parameter. If this parameter is omitted, the CKA\_ID of the private key is used.

### Syntax

**cmu selfSignCertificate** <parameters>

### Required Parameters

Parameter	Description
- <b>publichandle=</b> <pubkeyhandle#>	This is a mandatory parameter that defines the handle to the public key from an RSA key pair to be certified. If this parameter is omitted and there is only one public signing key on the HSM, that key is automatically selected. If this parameter is omitted and there are multiple public signing keys on the HSM, the user is asked to select the public signing key.
- <b>privatehandle</b> =<privkeyhandle#>	This is a mandatory parameter that defines the handle to the private key from an RSA key pair to be certified. If this parameter is omitted and there is only one private signing key on the HSM, that key is automatically selected. If this parameter is omitted and there are multiple private signing keys on the HSM, the user is asked to select the private signing key.
<b>-startDate</b>	This parameter defines the validity start of the certificate, in the format YYYYMMDD.
<b>-endDate</b>	This parameter defines the validity end of the certificate, in the format YYYYMMDD.
<b>-serialNumber</b>	This parameter defines the serial number of the certificate, in big-endian hexadecimal form.

### Optional Parameters

Parameter	Description
<b>-keyusage</b>	This is an optional parameter that defines the key usage extension for the certificate. It can be set to any of the following: digitalsignature, nonrepudiation, keyencipherment,

Parameter	Description
	dataencipherment, keyagreement, keycertsign, crlsign, encipheronly, decipheronly. This parameter may appear more than once in the parameter set to define multiple usages, or it can be used once with a comma separated list of usage types. If no key usage is specified, a default setting of keycertsign is used.
<b>-label</b>	This is an optional parameter that defines the CKA_LABEL attribute for the certificate object that gets created on the HSM. If omitted, the common name of the issuer and subject DN is used instead.
<b>-id</b>	This is an optional parameter that defines the CKA_ID attribute for the certificate object that gets created on the HSM. If omitted, the CKA_ID attribute of the private key is used instead.
<b>-md5WithRsa</b>	This is an optional parameter that defines the signature algorithm for the certificate request to be pkcs-1-MD5withRSAEncryption. The default is to use sha1WithRsa.
<b>-sha1WithRsa</b>	This is an optional parameter that defines the signature algorithm for the certificate request to be pkcs-1-SHA1withRSAEncryption. The default is to use sha1WithRsa.
<b>- sha224withrsa</b>	This is an optional parameter that defines the signature algorithm for the certificate request to be pkcs-1-sha224withRSAEncryption. The default is to use sha1WithRsa.
<b>- sha256withrsa</b>	This is an optional parameter that defines the signature algorithm for the certificate request to be pkcs-1-sha256withRSAEncryption. The default is to use sha1WithRsa.
<b>- sha384withrsa</b>	This is an optional parameter that defines the signature algorithm for the certificate request to be pkcs-1-sha384withRSAEncryption. The default is to use sha1WithRsa.
<b>- sha512withrsa</b>	This is an optional parameter that defines the signature algorithm for the certificate request to be pkcs-1-sha512withRSAEncryption. The default is to use sha1WithRsa.
<b>- sha1withdsa</b>	This is an optional parameter that defines the signature algorithm for the certificate request to be pkcs-1-sha1withDSAEncryption. The default is to use sha1WithRsa.
<b>- sha1withecdsa</b>	This is an optional parameter that defines the signature algorithm for the certificate request to be pkcs-1-sha1withECDSAEncryption. The default is to use sha1WithRsa.
<b>- sha224withecdsa</b>	This is an optional parameter that defines the signature algorithm for the certificate request to be pkcs-1-sha224withECDSAEncryption. The default is to use sha1WithRsa.
<b>- sha256withecdsa</b>	This is an optional parameter that defines the signature algorithm for the certificate request to be pkcs-1-sha256withECDSAEncryption. The default is to use sha1WithRsa.
<b>- sha384withecdsa</b>	This is an optional parameter that defines the signature algorithm for the certificate request to be pkcs-1-sha384withECDSAEncryption. The default is to use sha1WithRsa.
<b>- sha512withecdsa</b>	This is an optional parameter that defines the signature algorithm for the certificate request to be pkcs-1-sha512withECDSAEncryption. The default is to use

Parameter	Description
	sha1WithRsa.
<b>-C</b>	This optional parameter defines the two-letter country name for the subject distinguished name (DN) and issuer Distinguished Name of the certificate. This parameter should be present in each DN.
<b>-S</b>	This optional parameter defines the state or province name for the subject DN and issuer DN of the certificate. This parameter may be present in each DN.
<b>-L</b>	This optional parameter defines the locality (typically the city) for the subject DN and issuer DN of the certificate. This parameter MAY be present in each DN.
<b>-O</b>	This optional parameter defines the organization name for the subject DN and issuer DN of the certificate. This parameter SHOULD be present in each DN.
<b>-OU</b>	This optional parameter defines the organization unit name for the subject DN and issuer DN of the certificate. This parameter MAY be present in each DN.
<b>-CN</b>	This optional parameter defines the common name for the subject DN and issuer DN of the certificate. This parameter SHOULD be present in each DN.

## Example

The following example creates a self-signed certificate for RSA key 4:

```
cmu selfSign -publichandle=4 -privatehandle=5 -C=CA -O=Rainbow-Chrysalis -CN="Test Root Certificate" -startDate=20120101 -endDate=20151231 -serialNum=0133337f
```

## cmu setattr

This function sets any modifiable attributes for an object. An optional input filename can be used to specify a file from which the new attribute values are to be read.

### Syntax

**cmu setattr** <parameters>

### Required Parameters

Parameter	Description
<b>-handle</b> =<handle#>	This is a mandatory parameter that defines the handle to the object on the HSM. If this parameter is omitted and there is only one object on the HSM, that object is automatically selected. If this parameter is omitted and there are multiple objects on the HSM, the user is asked to select the object

### Optional Parameters

Parameter	Description
<b>-inputFile</b>	This optional parameter names a file from which to obtain additional attribute settings. The settings in this file shall be one per line and of the form: attributeName=attributeValue.
<b>-label</b>	This optional parameter defines a new value for the label of an object on the HSM.
<b>-application</b>	This optional parameter defines a new value for the application attribute of a data object on the HSM.
<b>-value</b>	This optional parameter defines a new value attribute for an object on the HSM. It must be set to a big-endian hexadecimal integer value. Note that the value attribute can be changed only for data objects, not for certificates or keys.
<b>-issuer</b>	This optional parameter defines a new issuer attribute for a certificate on the HSM. It must be set to a big-endian hexadecimal integer value. Note that this field is informational, typically set to the DER encoding of the issuer field within the certificate, and changing it does not affect the actual issuer field within the certificate itself.
<b>-serialNumber</b>	This optional parameter defines a new serial number attribute for a certificate on the HSM. It must be set to a big-endian hexadecimal integer value. Note that this field is informational, typically set to the DER encoding of the serial number of the certificate, and changing it does not affect the actual serial number field within the certificate itself.
<b>-subject</b>	This optional parameter defines a new subject field for an object on the HSM. It must be set to a big-endian hexadecimal integer value. The subject field is typically set to the DER encoding of the subject distinguished name for the key or certificate. Note that the subject is not modifiable for certificate objects once they are created.

Parameter	Description
<b>-id</b>	This optional parameter defines a new ID field for a key or certificate on the HSM. It must be set to a big-endian hexadecimal integer value.
<b>-extractable</b>	This optional parameter defines a new extractable setting for a private key on the HSM. This setting can only be changed from True to False (or from 1 to 0).
<b>-encrypt</b>	Set to True or False (or 1 or 0). Note that an HSM is typically configured such that functional key attributes cannot be changed, so attempting to change this attribute will be rejected by the HSM.
<b>-decrypt</b>	Set to True or False (or 1 or 0). Note that an HSM is typically configured such that functional key attributes cannot be changed, so attempting to change this attribute will be rejected by the HSM.
<b>-sign</b>	Set to True or False (or 1 or 0). Note that an HSM is typically configured such that functional key attributes cannot be changed, so attempting to change this attribute will be rejected by the HSM.
<b>-verify</b>	Set to True or False (or 1 or 0). Note that an HSM is typically configured such that functional key attributes cannot be changed, so attempting to change this attribute will be rejected by the HSM.
<b>-wrap</b>	Set to True or False (or 1 or 0). Note that an HSM is typically configured such that functional key attributes cannot be changed, so attempting to change this attribute will be rejected by the HSM.
<b>-unwrap</b>	Set to True or False (or 1 or 0). Note that an HSM is typically configured such that functional key attributes cannot be changed, so attempting to change this attribute will be rejected by the HSM.
<b>-derive</b>	Set to True or False (or 1 or 0). Note that an HSM is typically configured such that functional key attributes cannot be changed, so attempting to change this attribute will be rejected by the HSM.
<b>-startDate</b>	This optional parameter defines a new startDate field for a key on the HSM. The format for the value is YYYYMMDD.
<b>-endDate</b>	This optional parameter defines a new endDate field for a key on the HSM. The format for the value is YYYYMMDD.
<b>-sensitive</b>	Set to True or False (or 1 or 0). Note that an HSM is typically configured such that functional key attributes cannot be changed, so attempting to change this attribute will be rejected by the HSM.

## Example

The following example changes the key with handle 43 to be unextractable:

```
cmu setAttribute -handle=43 -extractable=False
```

## cmu verifypkc

Verify a Public Key Confirmation from the HSM.

### Syntax

```
cmu verifypkc <parameters>
```

### Required Parameters

Parameter	Description
<b>-inputFile</b>	This parameter defines the name of the file that contains the PKC.
<b>-pkctype</b>	This parameter defines the PKC type (1 - TC-TrustCenter, 2 - Chrysalis-ITS).

### Optional Parameters

None.

### Example

```
cmu verifypkc -inputFile=test.pkc -pkctype=1
```

This chapter describes how to access and use the ckdemo demonstration utility. The ckdemo utility is a simple console-based tool that provides a menu of functions that perform operations based on the PKCS#11 API. The ckdemo utility is included with the SafeNet Luna HSM client and can be used with any SafeNet Luna HSM.

This chapter contains the following sections:

- ["Accessing the ckdemo Utility" below](#)
- ["Using the ckdemo Menu" on the next page](#)
- ["The TOKEN Menu Functions" on page 42](#)
- ["The OBJECT MANAGEMENT Menu Functions" on page 45](#)
- ["The SECURITY Menu Functions" on page 46](#)
- ["The HIGH AVAILABILITY RECOVERY Menu Functions" on page 47](#)
- ["The KEY Menu Functions" on page 48](#)
- ["The CA Menu Functions" on page 49](#)
- ["The OTHERS Menu Functions" on page 50](#)
- ["The OFFBOARD KEY STORAGE Menu Functions" on page 52](#)
- ["The SCRIPT EXECUTION Menu Functions" on page 53](#)
- ["The CLUSTER EXECUTION Menu Functions" on page 53](#)
- ["The PED INFO Menu Functions" on page 53](#)
- ["The AUDIT/LOG Menu Functions" on page 54](#)
- ["The SRK Menu Functions" on page 55](#)
- ["The POLICY Menu Functions" on page 56](#)

## Accessing the ckdemo Utility

---

The ckdemo utility is included with the SafeNet Luna HSM client. How you access it depends on whether you are using Windows or Linux/UNIX.

### To access ckdemo from a Linux client:

1. Go to the SafeNet Luna HSM client binary directory.

```
cd /usr/safenet/lunaclient/bin
```

2. Launch the ckdemo utility.

```
./ckdemo
```

The ckdemo main menu is displayed. See ["Using the ckdemo Menu" on the next page](#).

**To access ckdemo from a Windows client:**

1. Navigate to the SafeNet Luna HSM client installation folder (**C:\Program Files\SafeNet\LunaClient**).
2. Double-click on **ckdemo** to open a console window with the ckdemo interface.

The ckdemo main menu is displayed. See "Using the ckdemo Menu" on page 1.

## Using the ckdemo Menu

When you launch the ckdemo utility, the ckdemo menu is displayed. The ckdemo menu provides access to numerous functions in several categories, as illustrated below:

**Figure 1: The ckdemo menu**

```

TOKEN:
  ( 1) Open Session   ( 2) Close Session ( 3) Login
  ( 4) Logout         ( 5) Change PIN    ( 6) Init Token
  ( 7) Init Pin       ( 8) Mechanism List ( 9) Mechanism Info
  (10) Get Info       (11) Slot Info     (12) Token Info
  (13) Session Info  (14) Get Slot List (15) Wait for Slot Event
  (16) Token Status  (18) Factory Reset (19) CloneMofN
  (33) Token Insert  (34) Token Delete
  (36) Show Roles    (37) Show Role Configuration Policies
  (38) Show Role State (39) Get OUID
  (58) HSM Zeroize   (59) Token Zeroize

OBJECT MANAGEMENT:
  (20) Create object (21) Copy object   (22) Destroy object
  (23) Object size   (24) Get attribute (25) Set attribute
                   (26) Find object   (27) Display Object
  (30) Modify Usage Count (31) Destroy Multiple Objects
  (32) Extract Public Key

SECURITY:
  (40) Encrypt file (41) Decrypt file (42) Sign
  (43) Verify       (44) Hash file   (45) Simple Generate Key
                   (46) Digest Key

HIGH AVAILABILITY RECOVERY:
  (50) HA Init      (51) HA Login      (52) HA Status

KEY:
  (60) Wrap key     (61) Unwrap key    (62) Generate random number
  (63) Derive Key   (64) PBE Key Gen   (65) Create known keys
  (66) Seed RNG     (67) EC User Defined Curves

CA:
  (70) Set Domain   (71) Clone Key     (72) Set MofN
  (73) Generate MofN (74) Activate MofN (75) Generate Token Keys
  (76) Get Token Cert Info (77) Sign Token Cert
  (78) Generate CertCo Cert (79) Modify MofN
  (86) Dup. MofN Keys (87) Deactivate MofN
  (88) Get Token Certificates (112) Set Legacy Cloning Domain

OTHERS:
  (90) Self Test
  (94) Open Access (95) Close Access
  (97) Set App ID (98) Options (100) LKM Commands

OFFBOARD KEY STORAGE:
  (101) Extract Masked Object (102) Insert Masked Object
  (103) Multisign With Value (104) Clone Object
  (105) SIMExtract (106) SIMInsert
  (107) SimMultiSign (118) Extract Object

```



```

(119) Insert Object

SCRIPT EXECUTION:
(108) Execute Script          (109) Execute Asynchronous Script
                              (110) Execute Single Part Script

CLUSTER EXECUTION:
(111) Get Cluster State
(113) Lock Clustered Slot    (114) Unlock Clustered Slot

PED INFO:
(120) Set Ped Info          (121) Get Ped Info (122) Init RPV
(123) Delete RPV

AUDIT/LOG:
(130) Get Config           (131) Set Config   (132) Verify logs
(133) Get Time             (134) Set Time     (135) Import Secret
(136) Export Secret        (137) Init Audit   (138) Get Status
(139) Log External

SRK:
(200) SRK Get State        (201) SRK Restore  (202) SRK Resplit
(203) SRK Zeroize          (204) SRK Enable/Disable

POLICY:
(53) Show Partition Policies (54) Set Partition Policies
(55) Show HSM Policies (56) Set HSM Policies (57) Set Destructive HSM Policies

(TITLE) menu titles, (99 or FULL) Full Help, (NONE) No help, (0 or EXIT) Quit

Enter your choice :
```

## Executing a Menu Function

To execute one of the functions listed in the menu, type the number of the function and press **Enter**. In general, if parameters or options are required, you are prompted to provide the additional information. Because most of the commands represent separate functions on an HSM, you may need to use more than one command to accomplish a task. For example, many of the commands require that you open a session on a token slot or HSM partition. Other commands require that you first login to the HSM or partition.

Functions that involve authentication or initialization of the HSM invoke the Luna PED for Trusted Path appliances. If the Luna PED is not connected and ready when a command is issued, the command eventually times out. If the Luna PED is connected and ready, it displays a prompt requesting the appropriate action. If you do not provide the requested PED key or keypad press, the Luna PED eventually times out and returns an error to the calling application (in this case, ckdemo).

The individual ckdemo functions are described in detail in the following sections:

- ["The AUDIT/LOG Menu Functions" on page 54](#)
- ["The CA Menu Functions" on page 49](#)
- ["The CLUSTER EXECUTION Menu Functions" on page 53](#)
- ["The HIGH AVAILABILITY RECOVERY Menu Functions" on page 47](#)
- ["The KEY Menu Functions" on page 48](#)
- ["The OBJECT MANAGEMENT Menu Functions" on page 45](#)
- ["The OFFBOARD KEY STORAGE Menu Functions" on page 52](#)
- ["The OTHERS Menu Functions" on page 50](#)
- ["The PED INFO Menu Functions" on page 53](#)

- ["The SCRIPT EXECUTION Menu Functions" on page 53](#)
- ["The SECURITY Menu Functions" on page 46](#)
- ["The SRK Menu Functions" on page 55](#)
- ["The TOKEN Menu Functions" below](#)

## The TOKEN Menu Functions

---

The TOKEN menu provides the following functions:

### ( 1) Open Session

Before you can manipulate objects or perform cryptographic operations on a token, you must have an open session on that token. This command prompts you for the number of the slot on which to open the new session. By default, an exclusive, Read/Write session is opened. If you would like to open a read only or non-exclusive session, you must use the **(98) Options** function and specify that you want to be prompted for session types.

### ( 2) Close Session

Once you are finished using a session, the session should be closed. The Close Session option allows you to close a single session, or to close all the sessions on a specific token.

### ( 3) Login

Once a session is opened, you usually log on to the token. You have a choice between logging on as a User (where you do most of your work with the token) or as Security Officer "SO" (where you can set up the user PIN and do any token administration operations).

### ( 4) Logout

When you are finished with the token, you should first log out, then close the session.

### ( 5) Change PIN

(Not for SafeNet Luna Network HSM) This option lets you change the logon password (the PIN) of the currently logged in user. You must supply both the old PIN and the new PIN to complete the operation.

### ( 6) Init Token

(Not for SafeNet Luna Network HSM) This option allows you to reset a token to its initial state. You are prompted for the following:

- The slot containing the token to be initialized
- The token label (which is simply a text string that you can use for Token Identification)
- A new password for the Security Officer

Token initialization performs the following actions:

- Wipes out any token objects (Keys, certificates, etc)

- Clears the user PIN (so that it must be reset by the Security Officer)
- Sets the SO PIN to the value that you have specified

## ( 7) Init PIN

(Not for SafeNet Luna Network HSM) This command is used to create a user (and thus overwrites an existing user) and is run when you are logged in as the Security Officer.

## ( 8) Mechanism List

This option gives a list of all the encryption/authentication/hashing/key-generation mechanisms supported by the token. If you want to know if the token supports a specific type of encryption, you can check for it in the mechanism list.

## ( 9) Mechanism Info

This option allows you to query a specific mechanism to find such information as supported key sizes. You are asked for the Mechanism type, which is a numeric value representing the mechanism (these numeric values are given when you request a mechanism list).

## (10) Get Info

This option returns basic information on the Dynamic Library that is being used to talk to the token. None of this information is token specific, and it can be viewed even if there is no token present.

## 11) Slot Info

This option gives specific information on a card slot. The slot description and slot ID are given, as well as some flags to represent if a token is present.

## 12) Token Info

This option gives information on a token in a specific slot, including the following:

- Token Label
- Token Manufacturer
- Token Model
- Token Flags
- Session Count
- Min and Max PIN Lengths
- Private memory size/free
- Public memory size/free

## 13) Session Info

This option gives information on an open session. You must have at least one session opened to query session information. For a particular session you can find the session handle, the slot ID, the session state, and any associated

session flags.

## 14) Get Slot List

This option returns a list of card slots available on the system. You are given the option to view all slots, or just the slots which contain tokens.

## 15) Wait for Slot Event

Runs CK\_WaitforSlotEvent (from PKCS#11 Extensions).

## 18) Factory Reset

This option resets the HSM to its factory settings.

## 19) Clone MofN

(Not for SafeNet Luna Network HSM) Copy a clonable secret-splitting vector from one token to another.

## 33) Token Insert

(For SafeNet Luna USB HSM) This option signals the HSM or local workstation that a token will be inserted. Insert the token to begin performing operations with it.

## 34) Token Delete

(For SafeNet USB HSM) This option deletes the token in a specific slot.

## 36) Show Roles

This option lists the roles currently configured on the token in a specific slot.

## 37) Show Role Configuration Policies

This option lists the role configuration policies currently in effect for the named role on the token in a specific slot.

## 38) Show Role State

This option shows the state of the named role. Information given includes:

- Primary authentication type
- Secondary authentication type
- Failed login attempts before lockout
- Init status

## 39) Get OUID

This option retrieves the OUID (Object Unique Identifier) of a token in a specific slot.

## 58) HSM Zeroize

This option zeroizes the HSM, removing all partitions and keys. HSM zeroization does not destroy the RPV or Auditor role.

## 59) Token Zeroize

This option zeroizes the token in a specific slot, removing all keys and objects.

# The OBJECT MANAGEMENT Menu Functions

The OBJECT MANAGEMENT menu provides the following functions:

## (20) Create Object

This option allows you to create objects on the token. You can use this option to create data or certificate objects on the token. You are presented with a default template for your new object that you can change or choose to accept as default.



**Note:** Key generation is not done with this option, instead you should use option #45 - Generate Key

---

## (21) Copy Object

This option allows you to make a copy of a token object and allows you to add/remove/change attributes of the object as you copy it.

## (22) Destroy Object

This option allows you to permanently delete a token object from the token.

## (23) Object Size

This option asks you for an object handle and returns the total size of the object (how much memory it is occupying on the token).

## (24) Get Attribute

This option asks you for an object handle and returns the attributes of that object.

## (25) Set Attribute

This option allows you to change the value of an attribute on an object that already exists on the token.

## (26) Find Objects

This option searches the token for objects that are available to you as the User or the SO (depending on which identity you used to log in). You specify a type (such as Data Objects, various Key objects, Certificate Objects, etc.). Option

(#6) shows all the objects on the token.

## (27) Display Object

This option shows all the attributes and associated values for an object on the token (if that object is available to you).



**Note:** If a key is sensitive, it contains an attribute called CKA\_VALUE but this attribute is not displayed because the token does not allow this information to be exported.

---

## (30) Modify Usage Count

This option allows you to increment the current value, or specify a new value, for an object's usage counter. You are prompted for the object handle and whether you want to increment or reset the usage counter for the specified object.

## (31) Destroy Multiple Objects

This option allows you to permanently delete multiple token objects from the selected token.

## (32) Extract Public Key

This option allows you to specify a public key to extract from the HSM. The key is saved as **publickey.bin** in the current directory, overwriting any existing **publickey.bin** file.



**Note:** The Extractable attribute must be set to 1 (On) in order for a public key to be extracted from the HSM.

---

# The SECURITY Menu Functions

---

The SECURITY menu provides the following functions:

## (40) Encrypt File

This option allows you to encrypt a file. You are asked which encryption mechanism you wish to use, then the filename of the file to be encrypted, and finally the key handle of the key to be used in the encryption operation.

## (41) Decrypt File

This option allows you to decrypt an encrypted file. You are asked for the encryption mechanism to use to decrypt the file, name of the file to be decrypted, and the handle of the key to be used for the decryption.

## (42) Sign

This option signs a string of data using a token signing mechanism. You are prompted for the signing mechanism that you wish to use, the data to be signed, and the key handle of the signing key (private key when using a Private/Public key pair).



**Note:** This option takes in a string of data to be signed from the keyboard, rather than a filename of a file containing the data (like encryption does). The signature is saved to a file called SIGN.BIN

### (43) Verify

This option verifies a signature against a string of data. You are prompted for the mechanism to be used for verification, the data to be verified and the key handle of the verification key. The signature is read from the file SIGN.BIN that is generated during the sign operation.

### (44) Hash

File This option prompts for the hashing mechanism to be used, and the name of the file to be hashed. The hash value is saved to a file called DIGEST.HSH at the end of the operation.

### (45) Simple Generate Key

This option performs key generation on the token. You are presented with a menu of possible key types. Depending on the key type being generated, you are asked a list of question about the attributes of the key(s). If the option to use array attributes is enabled through the OTHERS menu, you are presented with the option to use and edit a CKA\_UNWRAP\_TEMPLATE or CKA\_DERIVE\_TEMPLATE. These templates affect the Unwrap Key and Derive Key functions.

### (46) Digest Key

This option prompts for a digest mechanism and a key handle. The key value is digested using the selected mechanism.

## The HIGH AVAILABILITY RECOVERY Menu Functions

The HIGH AVAILABILITY RECOVERY menu provides the following functions:

### (50) HA Init

(Not for SafeNet Luna Network HSM) Requires that an RSA keypair have been previously created, and the private key cloned to User space of the affected tokens. This option requires the handle to the session (of the User that owns the key pair) and the handle to the login private key itself.

### (51) HA Login

(Not for SafeNet Luna Network HSM) This option initiates several functions, including creation of a TWC (Token Wrapping Certificate) blob and HA Login Challenge (secondary token in the current HA domain) and Acceptance (primary token), as described in the document Extensions to PKCS#11, Cryptographic Token Interface Standard.

### (52) HA Status

Display the current status for a specified HA slot.

---

## The KEY Menu Functions

---

The KEY menu provides the following functions:

### 60) Wrap Key

This option allows you to encrypt a key. You must provide the encryption mechanism type, the handle of the wrapping key (used to encrypt the key), and the handle of the key to be wrapped (the one that is going to be encrypted). Currently, the wrapping of private asymmetric keys is not supported.

### 61) Unwrap Key

This option allows you to import a wrapped (encrypted) key into the token. You are asked for the mechanism to be used for the unwrapping operation as well as what type of key is being unwrapped. Depending on the type of key being unwrapped, you are asked for some information about the key. Then you must provide a key handle of the token key to be used in the unwrapping (decryption) operation, and finally, give the name of the file containing the wrapped key. If the unwrapping key has an associated CKA\_UNWRAP\_TEMPLATE attribute, this affects the results of the operation. Note that if you are generating a key in CKDemo, the option to attach an unwrap template is disabled by default. You can enable this option in the OTHERS menu.

### 62) Generate Random Number

This option generates a specified amount of random data. You are asked how many bytes of random data to generate, then are presented with the random value.

### 63) Derive Key

This option allows you to use a key derivation mechanism to derive a key on the token. There are several key derivation mechanisms to choose from, and you are presented with a menu of the choices. Depending on the key derivation mechanism, you are asked for some information about the key. If the base key used for generation includes a CKA\_DERIVE\_TEMPLATE attribute, the information you provide is added with the attributes in the derive template. If your information contradicts the attributes in the derive template, the derive operation fails. Note that if you are generating a key in CKDemo, the option to attach a derive template is disabled by default. You can enable this option in the OTHERS menu.

### 64) PBE Key Generation

This option allows you to perform a "Password Based Encryption" key generation. This option is useful because it allows you to put the same key on multiple tokens without ever knowing the key value itself.

### 65) Create Known Keys

This option attempts to load a known key onto the token. However, due to policy setting on most tokens, this option is not allowed. As an alternative, it is possible to encrypt a known key and then unwrap it onto the token. See the Unwrap Key sample code on the SDK distribution CD.

### 66) Seed RNG

Provide a seed value to the HSM's Random Number Generator.



## 67) EC User Defined Curves

Set the desired attributes and point to a file containing Elliptical Curve parameters for generating EC keys.

## The CA Menu Functions

---

The CA menu provides the following functions:

### (70) Set Domain

(Not for SafeNet Luna Network HSM) This option prompts for a text string and sets the token cloning domain name to that value. To clone a key between two SafeNet CA3 tokens, both tokens must share the same red PED key.

### (71) Clone Key

(Not for SafeNet Luna Network HSM) This option allows you to clone a key from one SafeNet RA token to another (or one SafeNet CA3 token to another). Both tokens must have the same cloning domain name (or red PED key). Both tokens must have an open and logged on session active.

### (72) Set MofN

(Not for SafeNet Luna Network HSM) If you have a SafeNet CA3 token (which supports MofN authentication), this option allows you to turn on the MofN token feature. This option alone does nothing to the token, but instead sets a flag specifying that the next token to be initialized should have its MofN feature turned on (assuming, of course, that the token supports it).

### (73) Generate MofN

(Not for SafeNet Luna Network HSM) This option allows you to generate MofN authentication splits, or secret shares. You can generate up to 16 shares (N), and you can specify how many of these shares are needed (M) in order to activate the token (up to 16).

### (74) Activate MofN

(Not for SafeNet Luna Network HSM) This option allows you to authenticate yourself to the token using MofN secret shares generated by option #73 (Generate MofN). You must activate MofN on a token on which MofN has been generated, or you are unable to perform any cryptographic operations with the token.

### (75) Generate Token Keys

(Not for SafeNet Luna Network HSM) Some tokens have the ability to support customer loaded certificates used for key cloning. If your token supports this feature, and you wish to use your own key cloning certificates (rather than the default certificates provided by SafeNet), the first step is to Generate token keys.



**Note:** If you do this, you are not able to clone to any other SafeNet CA tokens except those containing your own certificate.

---

## (76) Get Token Cert

(Not for SafeNet Luna Network HSM) This option is the next step in loading your own key cloning certificate onto the token. This action is done after #75 (Generate Token Keys).

## (77) Sign Token Cert

(Not for SafeNet Luna Network HSM) This option is the final step to load a customer key cloning certificate to the token. This step is done after Steps 75 and 76.

## (78) Generate CertCo Cert

(Not for SafeNet Luna Network HSM) Generate a special-purpose certificate for CertCo application.

## (79) Modify MofN

(Not for SafeNet Luna Network HSM) Modifies the secret splitting vector on a token.

## (86) Dup. MofN Keys

(Not for SafeNet Luna Network HSM) Create duplicates (copies) of all MofN secret splits.

## (87) Deactivate MofN

Decache the MofN data.

## (88) Get Token Certificates

Extract one of the following certificates from the HSM. You must supply the type and filename of the certificate you want to extract:

- Root certificate
- Hardware origin certificate
- ECC hardware origin certificate
- TWC (token wrapping certificate) version 1, 2, or 3.
- TCTrust device authentication certificate
- CITS device authentication certificate

## (112) Set Legacy Cloning Domain

This option sets the legacy Cloning Domain, from a legacy token, into association with the modern cloning domain attached to a current-model SafeNet Luna HSM, to allow migration of token objects from legacy HSMs.

## The OTHERS Menu Functions

---

The OTHERS menu provides the following functions:

## 90) Self Test

Not currently supported.

## 94) Open Access

Creates a token access ID that is independent of any sessions so that the login state can be maintained even when your application exits. Used to allow the same application to return repeatedly for access without requiring a separate login each time. Remains active until Closed (command 95, below) or until the token is removed.

## 95) Close Access

Kills the ID generated by command 94, above.

## 97) Set App ID

You are prompted to type in an explicit application ID (in two parts, Major and Minor), rather than having it generated by Chrystoki. Doing so effectively causes all processes (using that Major/Minor application ID) on the machine to be recognized as the same application. Refer to the PKCS#11 Extensions document.

## 98) Options

This item allows you to change some default options of the CKDemo program. You can turn off help (which prevents the entire menu from being displayed after each command), or select the type of session you wish the Open Session command to use. Use option #0 to exit this menu and return to the CKDemo main menu.

Use option 16 if HSM firmware is newer than version 6.22.0 and you wish to use CKR\_TEMPLATE\_INCONSISTENT.

Option	Description (Default)	Alternate
1 - Open Session Type	Always R/W and Serial	User selectable
2 - Display Help	Always	On demand
3 - PIN path	User supplied ASCII password	Selectable
4 - Echo input	Disabled	On all commands and data
5 - Sleep for n seconds	Sleep for n seconds after writing special instructions to stderr	Enter a number of seconds to sleep. Then enter the desired instructions. Finish entering instructions with a period ( . ) alone on a line
6 - KCV Default	User supplied KCV Domain	Selectable
7 - MofN path	User supplied MofN path	Selectable
8 - Show Response Code	SHOW_RESPONSE_BEFORE_MENU	SHOW_RESPONSE_BEFORE_AND_AFTER_MENU

Option	Description (Default)	Alternate
9 - Input data for sign/derive	Input from keyboard	Input from file
10 - Object Usage Counters	Disabled	Selectable
11 - GCM IV Source	External	Internal
12 - ECIES Parameters	Use default (XOR with HMAC_SHA1)	Selectable
13 - X9.31 Signatures	Allow X9.31 generated keys only	Allow non-X9.31 generated keys
14 - Multipart enc/dec/sig/ver	Use single part operations	Use multi-part operations
15 - Use Old Enc/Dec Menu	Use old Encrypt/Decrypt menu	Use new Encrypt/Decrypt menu
16 - Role Support	Enhanced roles - use with roles as they are implemented with PSO-capable firmware (f/w 6.22.0 and newer)	Use HSM with legacy SafeNet roles (as found with f/w previous to v6.22.0)
17 - OAEP Hash Params	Use default (SHA1 Digest and MGF1)	Selectable
18 - Array Template Attributes	Do not use array template attributes	Use array template attributes
0 - Finished		

## 100) LKM Commands

This option is to be deprecated in a future release, and is not usable.

## The OFFBOARD KEY STORAGE Menu Functions

The OFFBOARD KEY STORAGE menu provides the following functions:

### (101) Extract Masked Object

Extracts a key off the SafeNet Luna Network HSM in a masked format, into a file "masked.key". You can rename the resulting file if you are testing with multiple extractions.

### (102) Insert Masked Object

Inserts an extracted, masked blob (file) back onto the SafeNet Luna Network HSM. You are prompted for the name of the file, which must have been extracted from a SafeNet Luna Network HSM using the same masking key (i.e., the same SafeNet Luna Network HSM or a clone of it).

### **(103) Multisign With Value**

Performs the multisign function, after prompting you for the mechanism to use, the number of datablobs to be signed (limited to 5 for this demonstration command), and the data or filenames to be signed.

### **(104) Clone Object**

(Reserved for SafeNet use) Copies an object from the SafeNet Luna Network HSM to another HSM.

### **(105) SIMExtract**

This function takes a list of object handles, extracts them using the given authorization data for protection, and returns the extracted set of objects as a single data blob. The objects can be left on the partition or destroyed, depending on the value of the delete-after-extract flag.

### **(106) SIMInsert**

This function inserts the objects contained in a previously extracted blob into the HSM, and returns the list of handles assigned to the objects.

### **(107) SimMultiSign**

This function uses the key material in a previously extracted key blob to sign pieces of data in the input data table, returning the signatures through the signature table. The key blob must contain a single key, otherwise an error is returned.

### **(118) Extract Object**

Extracts a key off the SafeNet Luna Network HSM into a file.

### **(119) Insert Object**

Inserts an extracted blob (file) back onto the SafeNet Luna Network HSM.

## **The SCRIPT EXECUTION Menu Functions**

---

The SCRIPT EXECUTION menu and its functions are to be deprecated in a future release, and are not usable.

## **The CLUSTER EXECUTION Menu Functions**

---

The CLUSTER EXECUTION menu and its functions are to be deprecated in a future release, and are not usable.

## **The PED INFO Menu Functions**

---

The PED INFO menu provides the following functions:

## 120) Set PED Info

Specify the PED (local or remote) that is associated with the HSM in a specific slot.

## 121) Get PED Info

Display information describing the PED that is associated with the HSM in a specific slot.

## 122) Init RPV

Create a Remote PED Vector, and imprint it onto an orange Remote PED Key (RPK), to allow PED functions with a remotely located SafeNet Luna HSM (which must also have the same RPV).

## 123) Delete RPV

Remove the Remote PED Vector from the current HSM. Disallows Remote PED operation for this HSM until (if) a new RPV is created or an existing RPV is acquired from an imprinted RPK.

# The AUDIT/LOG Menu Functions

---

The AUDIT/LOG menu provides the following functions:

## (130) Get Config

Shows the current configuration for audit logging.

## (131) Set Config

Set the audit logging configuration parameters. This command allows you to configure the following:

- Which events are captured in the log
- The log rotation interval

## (132) Verify Logs

This command displays details for the indicated file, or verifies the audit log records in the specified range from the named file.

## (133) Get Time

This command displays the current HSM time.

## (134) Set Time

This command synchronizes the HSM time to the host time. This is especially useful when the host computer is synchronized by NTP, or by local drift correction. This ensures that the log times of HSM events coincide with file creation and update events in the host file system.

### (135) Import Secret

This command imports an audit log secret that was previous exported.

### (136) Export Secret

This commands exports the audit logging secret to the user local directory for import to another HSM.

### (137) Init Audit

This command initializes the Audit role on the HSM. An audit domain and role password (for password-authenticated HSMs) or white Audit PED key (for PED-authenticated HSMs) are attached. This command destroys any previously existing Audit role on the HSM.

### (138) Get Status

This command displays the audit logging information for the indicated HSM.

### (139) Log External

This SafeNet extension to PKCS#11 allows a user application to insert text into the log record stream. This command logs a string of the user's choice to the audit log file.

## The SRK Menu Functions



**Note:** These functions are not applicable to SafeNet Luna HSMs with firmware 7.x or newer.

The SRK menu provides the following functions:

### (200) SRK Get State

Shows the current state of the Master Tamper Key.

### (201) SRK Restore

Gets the external split (SRK) of the Secure Recovery Vector from a connected Luna PED, combines it with the internally-stored split, to regenerate the SRV, and re-validates the MTK

### (202) SRK Resplit

Performs a new split of the Secure Recovery Vector and places the external portion of the split onto a PED key (purple-labeled key called the Secure Recovery Key or SRK).

### (203) SRK Zeroize

Zeroize the SRK. This action simulates a hardware tamper.

## (204) SRK Enable/Disable

Enable splitting of the Secure Recovery Vector into an internal (to the HSM) portion and an external portion (stored on a purple PED key). Or, disables that function by bringing the external split back into the HSM (requires Luna PED and the purple PED key with the correct SRV split on it - that purple key then becomes invalid).

## The POLICY Menu Functions

---

The Policy menu provides the following functions:

### (53) Show Partition Policies

This command displays the partition-level capability and policy settings for the partition and User.

### (54) Set Partition Policies

This command sets a user policy on the partition.

### (55) Show HSM Policies

This command displays the HSM-level capability and policy settings for the HSM.



**Note:** The output of this command differs considerably, depending on the firmware version of the HSM in the current slot.

---

### (56) Set HSM Policies

This command sets HSM-level policies that are non-destructive.

### (57) Set Destructive HSM Policies

This command sets HSM-level policies that are destructive. That is, setting these policies forces the HSM to be wiped (reinitialized), destroying all contents.



This chapter describes how to access and use Multitoken, a simple demonstration tool that allows you to perform basic cryptographic functions on a SafeNet Luna HSM. It contains the following topics:

- ["Accessing Multitoken" below](#)
- ["Using Multitoken" below](#)

## Accessing Multitoken

The **multitoken** utility is a command line utility included with the SafeNet Luna Client software.

### To access the multitoken utility

1. Open a console window.
2. Go to the SafeNet Luna Client installation folder/directory:

<b>Windows</b>	C:\Program Files\SafeNet\LunaClient
<b>Linux/Unix</b>	/usr/safenet/lunaclient/bin

3. Launch the multitoken utility:

```
./multitoken
```

## Using Multitoken

The multitoken utility allows you to specify an operation, and one or more “slots” or HSM Partitions on which to perform that operation. The multitoken utility runs the operations and returns a summary, or progress report, of the results.

### Syntax

```
multitoken -mode <mode> -slots <slot list> [-nodestroy] [-key <key size>] [-curve <curve num>] [-blob <blob count>] [-packet <packet size>] [-logfile <logfile name>] [-force] [-help] [-symm] [-password <password>] [-timed <fixed time>] [-nodec] [-parmfile <param file>] [-noverify] [-multipartsignatures] [-subprime <subprime size>] [-noverify] [-nslots] [-keychoice <key index>] [-kdfchoice <kdf index>] [-kdfscnt [counter index>] [-sharefile <data file>] [-noenc] [-nosign] [-verbose] [-alarm <secs>] [-template]
```

Parameter	Shortcut	Description
<b>-alarm</b>	<b>-al</b>	Sound periodic alarm (every <secs> seconds) if error occurs.

Parameter	Shortcut	Description
<b>-blob</b>	<b>-b</b>	Number of data blobs to be signed during each multisign operation.
<b>-curv</b>	<b>-crv</b>	ID number of ECC curve. If user-defined (99), then must specify <b>-parmfile</b> .
<b>-force</b>	<b>-f</b>	Avoid prompts for responses.
<b>-ped</b>	<b>-ped</b>	Specify ped id (-ped 0 for local, -ped 1 for remote). This applies only to the first HSM slot to be specified using the '-s' option.
<b>-help</b>	<b>-h</b>	Display help information only.
<b>-key</b>	<b>-k</b>	Size of key: asymmetric in bits (default = 1024 for RSA, 2048 for DSA). symmetric in bytes (i.e. 16, 24, 32 for AES/ARIA).
<b>-keychoice</b>	<b>-kc</b>	Select key type to derive/generate - specify choice list index.
<b>-kdfchoice</b>	<b>-kdf</b>	Select key derivation function - specify choice list index.
<b>-kdfscnt</b>	<b>-kds</b>	Select key derivation session counter type - specify choice list index.
<b>-usage</b>	<b>-u</b>	Number of times a key is allowed to be used.
<b>-logfile</b>	<b>-l</b>	File for results logging.
<b>-mode</b>	<b>-m</b>	Operating mode. See mode values available below.
<b>-multipartsig</b>	<b>-msig</b>	Use multipart signatures.
<b>-nodec</b>	<b>-nod</b>	Decryption operation will not be performed. Only symmetric and asymmetric encryption will be performed and measured.
<b>-nodelstroy</b>	<b>-n</b>	Leaves created objects on the HSM after test completes.
<b>-noenc</b>	<b>-noe</b>	Perform only one encryption operation. Only symmetric and asymmetric decryption will be performed and measured.
<b>-nosign</b>	<b>-nos</b>	Perform only one sign operation. Only verify will be performed and measured.
<b>-noverify</b>	<b>-nov</b>	Verify operation will not be performed. Only sign will be performed and measured.
<b>-noverifyr</b>	<b>-nvr</b>	Do not verify decryption results.
<b>-packet</b>	<b>-p</b>	Size of packet used in operation.
<b>-parmfile</b>	<b>-prm</b>	File for EC curve parameters or OAEP source data (0 = none for OAEP).
<b>-password</b>	<b>-pwd</b>	Specify password to use for token.
<b>-prftype</b>	<b>-prf</b>	Specify the type of PRF to use for PRF based key derivation.
<b>-sharefile</b>	<b>-shf</b>	Shared data file used for operation.
<b>-slots</b>	<b>-s</b>	List of of slots to use (slot numbers separated by commas).

Parameter	Shortcut	Description
<b>-subprime</b>	<b>-sub</b>	Size of the subprime in bits.
<b>-symm</b>	<b>-sym</b>	Select symmetric key mechanism for symderive/pbegen or key choice for symgen (can also use -kc).
<b>-timed</b>	<b>-t</b>	Fixed amount of time to run (seconds).
<b>-nslots</b>	<b>-ns</b>	Slots and threads to be specified as slot number times (x or X) number of threads, then comma for next pair. Ex. -ns 1x5,2X10 This will create 5 threads on slot 1 and 10 threads on slot 2.
<b>-verbose</b>	<b>-v</b>	Show all thread performances. Default is only first and last threads.
<b>-template</b>	<b>-tp</b>	Attaches a generic unwrap template or derive template for the wrapunwrap or symderive mode respectively.

## Operating Modes

The following table lists the available operating modes for the multitoken utility. The operating mode is specified using the **-mode** parameter.

Mode	Description
rsakeygen	RSA key generation
rsax931keygen	RSA X9.31 key generation
rsasigver	RSA sign
sha1rsasigver	SHA1 with RSA sign
sha224rsasigver	SHA224 with RSA sign
sha256rsasigver	SHA256 with RSA sign
sha384rsasigver	SHA384 with RSA sign
sha512rsasigver	SHA512 with RSA sign
rsax931sigver	X9.31 RSA sign
sha1rsax931sigver	SHA1 X9.31 RSA sign
sha224rsax931sigver	SHA224 X9.31 RSA sign
sha256rsax931sigver	SHA256 X9.31 RSA sign
sha384rsax931sigver	SHA384 X9.31 RSA sign
sha512rsax931sigver	SHA512 X9.31 RSA sign
sha1rsapsssigver	SHA1 RSA PSS sign

Mode	Description
rsaenc	RSA encrypt
rsaoaepenc	RSA OAEP encrypt
dsakeygen	DSA Key Generation
dsasigver	DSA bare sign
sha1dsasigver	SHA1 DSA sign
sha224dsasigver	SHA224 DSA sign
sha256dsasigver	SHA256 DSA sign
ecdsakeygen	ECDSA Key Generation
ecdsasigver	ECDSA sign
ecdsasha1sigver	SHA1 ECDSA sign
ecdsasha224sigver	SHA224 ECDSA sign
ecdsasha256sigver	SHA256 ECDSA sign
ecdsasha384sigver	SHA384 ECDSA sign
ecdsasha512sigver	SHA512 ECDSA sign
kcdsakeygen	KCDSA Key Generation
kcdsasigver	HAS160 KCDSA 1024-bit sign
kcdsasha1sigver	SHA51 KCDSA sign
kcdsasha224sigver	SHA224 KCDSA sign
kcdsasha256sigver	SHA256 KCDSA sign
kcdsasha384sigver	SHA384 KCDSA sign
kcdsasha512sigver	SHA512 KCDSA sign
pbegen	PBE key generation
symgen	Symmetric key generation
symderive	Symmetric key derivation
rc4enc	RC4 encrypt
des3enc	DES3 ECB encrypt
des3enccbc	DES3 CBC encrypt

Mode	Description
des3encfb8	DES3 CFB8 encrypt
des3encfb64	DES3 CFB64 encrypt
des3encofb	DES3 OFB encrypt
desmac	DES3 MAC sign
descmac	DES3 CMAC sign
aesenc	AES ECB encrypt
aesenccbc	AES CBC encrypt
aesencfb8	AES CFB8 encrypt
aesencfb128	AES CFB128 encrypt
aesencofb	AES OFB encrypt
aesengcm	AES GCM encrypt
aesmac	AES MAC sign
aescmac	AES CMAC sign
ariaenc	ARIA ECB encrypt
ariaenccbc	ARIA CBC encrypt
ariaencfb8	ARIA CFB8 encrypt
ariaencfb128	ARIA CFB128 encrypt
ariacencofb	ARIA OFB sign
ariamac	ARIA MAC sign
ariacmac	ARIA CMAC sign
seedenc	SEED ECB encrypt
seedmac	SEED MAC sign
seedcmac	SEED CMAC sign
extractinsert	Extract Insert masked objects
multisignvalue	Multisign w/ masked key
simextractinsert	SIMExtract Insert masked objects
simmultisign	SIMMultisign w/ masked key

Mode	Description
sim3extractinsert	SIM3 Extract Insert masked objects
md5	MD5 Hashing
sha1	SHA-1 Hashing
sha224	SHA-224 Hashing
sha256	SHA-256 Hashing
sha384	SHA-384 Hashing
sha512	SHA-512 Hashing
sha1hmac	SHA1 HMAC sign
sha224hmac	SHA224 HMAC sign
sha256hmac	SHA256 HMAC sign
sha384hmac	SHA384 HMAC sign
sha512hmac	SHA512 HMAC sign
ecdhderive	ECDH derive key
ecdhcderive	ECDH Cofactor derive key
eciesxorhmacsha1	ECIES XOR enc/dec with HMAC SHA1
eciesxorhmacsha1shared	ECIES XOR enc/dec with HMAC SHA1 and shared data
eciesdes3hmacsha224	ECIES DES3 enc/dec with HMAC SHA224
eciesdes3hmacsha224shared	ECIES DES3 enc/dec with HMAC SHA224 and shared data
eciesaes128hmacsha256	ECIES AES-128 enc/dec with HMAC SHA256
eciesaes128hmacsha256shared	ECIES AES-128 enc/dec with HMAC SHA256 and shared data
eciesaes192hmacsha384	ECIES AES-192 enc/dec with HMAC SHA384
eciesaes192hmacsha384shared	ECIES AES-192 enc/dec with HMAC SHA384 and shared data
eciesaes256hmacsha512	ECIES AES-256 enc/dec with HMAC SHA512
eciesaes256hmacsha512shared	ECIES AES-256 enc/dec with HMAC SHA512 and shared data
wrapunwrap	Wrap/unwrap operations
randgen	Random number generation

## Notes

1. If you are performing RSA operations, you have the option of specifying a key size (512, 1024, 2048, 4096, 8192). If

no key size is specified, the default key size of 1024 will be used. For example:

```
multitoken -mode rsasigver -key 512 -slots 1
```

2. If you are performing wrapunwrap operation, it will perform the following operations:
  - Generate RSA key pair and a symmetric DES key.
  - Wrap DES key with RSA public key.
  - Unwrap wrapped key above with RSA private key.
  - Verify the unwrapped key.
3. If you are performing a Multisign operation, you have the option of specifying a key size (512, 1024, 2048, 4096, 8192). If no key size is specified, the default key size of 1024 will be used. You must also specify a blob count, indicating the number of data blobs to be signed during each multisign operation. For example:

```
multitoken -mode multisignvalue -key 512 -blob 10 -s 1,1,2,2,2
multitoken -mode multisignvalue -blob 10 -s 1,1,2,2,2,2
```

4. A thread will be spawned to perform tests on each slot specified. A slot can be specified multiple times, in which case multiple threads will be created for the slot.
5. Options for the following modes can be used with the default 1024 bit key size only:
  - sha256rsasign - SHA256 with RSA
  - sha384rsasign - SHA384 with RSA
  - sha512rsasign - SHA512 with RSA

If you specify a keysize on the command line (any of 1024, 2048 or 4096), the result is the 1024 bit benchmark speed, and a file called "1024" or "2048" or "4096" is created - that is the keysize parameter is parsed as a filename to which results are saved.

## Named and User-defined Curves

The SafeNet Luna HSMs employ named and user-defined curves. Multitoken supports this option, as illustrated in the following example:

```
C:\Program Files\SafeNet\LunaClient>multitoken -mode ecdsasigver -s 1,1,1,1,1,1,1,1
```

Prime field curves:

```
[0]secp112r1
[1]secp112r2
[2]secp128r1
[3]secp128r2
[4]secp160k1
[5]secp160r1
[6]secp160r2
[7]secp192k1
[8]secp224k1
[9]secp224r1
[10]secp256k1
[11]secp384r1
[12]secp521r1
[13]X9_62_prime192v1
[14]X9_62_prime192v2
[15]X9_62_prime192v3
```

```
[16]X9_62_prime239v1
[17]X9_62_prime239v2
[18]X9_62_prime239v3
[19]X9_62_prime256v1
```

Characteristic two field curves:

```
[20]sect113r1
[21]sect113r2
[22]sect131r1
[23]sect131r2
[24]sect163k1
[25]sect163r1
[26]sect163r2
[27]sect193r1
[28]sect193r2
[29]sect233k1
[30]sect233r1
[31]sect239k1
[32]sect283k1
[33]sect283r1
[34]sect409k1
[35]sect409r1
[36]sect571k1
[37]sect571r1
[38]X9_62_c2pnb163v1
[39]X9_62_c2pnb163v2
[40]X9_62_c2pnb163v3
[41]X9_62_c2pnb176v1
[42]X9_62_c2tnb191v1
[43]X9_62_c2tnb191v2
[44]X9_62_c2tnb191v3
[45]X9_62_c2pnb208w1
[46]X9_62_c2tnb239v1
[47]X9_62_c2tnb239v2
[48]X9_62_c2tnb239v3
[49]X9_62_c2pnb272w1
[50]X9_62_c2pnb304w1
[51]X9_62_c2tnb359v1
[52]X9_62_c2pnb368w1
[53]X9_62_c2tnb431r1
[54]Brainpool_P160r1
[55]Brainpool_P160t1
[56]Brainpool_P192r1
[57]Brainpool_P192t1
[58]Brainpool_P224r1
[59]Brainpool_P224t1
[60]Brainpool_P256r1
[61]Brainpool_P256t1
[62]Brainpool_P320r1
[63]Brainpool_P320t1
[64]Brainpool_P384r1
[65]Brainpool_P384t1
[66]Brainpool_P512r1
[67]Brainpool_P512t1
```



Please pick a curve (0-67) or enter (99) for a user defined curve:99

Please enter the filename for the EC parameters:

Here, you would provide the filepath to the file specifying the Elliptical Curve parameters. The format and content of the parameter file follow industry standards, and are discussed in more detail in "[Named Curves and User-Defined Parameters](#)" on page 1 in the [SDK Reference Guide](#).

# Remote Backup Service (RBS)

This chapter describes how to use the RBS utility to remotely back up your HSMs. It contains the following topics:

- ["RBS Overview" below](#)
- ["rbs" on the next page](#)
- ["rbs config" on page 68](#)
- ["rbs daemon" on page 69](#)
- ["rbs genkey" on page 70](#)
- ["rbs nopassword" on page 71](#)

## RBS Overview

---

RBS implements the Remote Backup Service. RBS is run on a workstation with a SafeNet Luna Backup HSM connected.

RBS requires pedClient to be running both on the RBS computer and on the host of the SafeNet Luna HSM primary (the HSM being backed-up from, or being restored-to). PedClient enables the communication link over which RBS works.

PedClient is also used in conjunction with pedServer to enable Remote PED, and in the case where both the Backup HSM and the Remote PED are connected to the same administrative workstation, you might legitimately have all three of RBS, pedServer, and pedClient running on the one system.

See for more information.

## rbs

Access the RBS commands.

### Syntax

**rbs** [-daemon] [-genkey] [-nopassword] [-config] [-help]

Parameter	Shortcut	Description
<b>-config</b>	<b>-c</b>	Runs RBS to select devices to support for Remote Backup. See <a href="#">"rbs config" on the next page</a> .
<b>-daemon</b>	<b>-d</b>	Runs RBS in daemon (background) mode. See <a href="#">"rbs daemon" on page 69</a> .
<b>-genkey</b>	<b>-g</b>	Runs RBS to generate private key/certificate for Remote Backup. See <a href="#">"rbs genkey" on page 70</a> .
<b>-help</b>	<b>-h</b>	Displays help information for the rbs command.
<b>-nopassword</b>	<b>-n</b>	Require no password for encoded keys. See <a href="#">"rbs nopassword" on page 71</a> .

### Example

```
[admin@myluna bin] # ./rbs --h
```

Supported Options:

```
--help          help
--daemon        run as daemon           [optional] - default NOT daemon
--genkey        generate private key/certificate [optional]
--nopassword    no password for encoded keys  [optional] - password required
--config        select devices to support   [optional]
```

```
[admin@myluna bin] #
```

## rbs config

---

Runs rbs to select devices to support for Remote Backup.

### Syntax

**rbs --config**

### Options

None. "config" is an option of the rbs command.

### Example

```
[admin@myluna bin] # ./rbs --config
```

```
[admin@myluna bin] #
```

## rbs daemon

---

Runs RBS in daemon (background) mode. RBS is required for Remote Backup.

### Syntax

**rbs --daemon**

### Options

None. "daemon" is an option of the rbs command. Default for rbs is non-daemon mode.

### Example

```
[admin@myluna bin] # ../rbs/bin/rbs --daemon
Enter password : *****
[admin@myluna bin]#
```

## rbs genkey

---

Runs RBS to generate private key/certificate for Remote Backup.

### Syntax

**rbs --genkey**

### Options

None. "genkey" is an option of the rbs command. .

### Example

```
[admin@myluna bin] # ./rbs --genkey
Enter password : *****
Verify password: *****
[admin@myluna bin] #
```

## rbs nopassword

---

Require no password for encoded keys. Default is password required.

### Syntax

**rbs --nopassword**

### Options

None. "nopassword" is an option of the rbs command. .

### Example

```
[admin@myluna bin] # ./rbs --nopassword
```

```
[admin@myluna bin] #
```

This chapter describes how to use the **salogin** utility to allow applications that do not provide a native login interface to an HSM to login to a SafeNet Luna Network HSM. It contains the following topics:

- ["Using the salogin Utility" below](#)

## Using the salogin Utility

Cryptographic applications that are not specifically adapted to use an HSM Server can nevertheless be run using SafeNet Luna HSMs, with the aid of the **salogin** utility. This section provides the settings required for some widely-used applications.

An example of a situation where you might use **salogin** is where you wish to use a SafeNet Luna HSM appliance with openssl, which can be used with HSMs, but which has no inherent ability to provide credentials to the HSM.



**Note:** The applications in the integrations documents have been explicitly integrated by SafeNet, to work with your SafeNet Luna HSM product. Contact your SafeNet representative. If you are a developer, you might prefer to create or modify your own application to include support for the HSM or appliance. Refer to the Software Development Kit and the Extensions sections of this document set.



**Note:** The salogin utility does not work with STC-enabled slots. If you require salogin with your applications, you must use NTLS client links.

## The salogin Command

The **salogin** client-side utility is provided to assist clients that do not include the requisite HSM login and logout capability within the client application. Run the utility from a shell or command prompt, or include it in scripts.

The **salogin** utility has a single command, with several arguments, as follows:

```
>salogin -h
```

Luna Login Utility 1.0 Arguments:

o		open application access
c		close application access
i	hi:lo	application id; high and low component
s	slot	token slot id number (default = 1)
u		specifies that login should be performed as the Crypto-User if no



		user type is supplied, the Crypto-Officer will be used
p	pswd	challenge password - if not included, login will not be performed
r	server IP	remote ped server ip
v		verbose
h		this help

## Examples

```
salogin -o -s 1 -i 1:1
# open a persistent application connection
# on slot 1 with app id 1:1

salogin -o -s 1 -i 1:1 -p HT7bHTHPRp/4/Cdb
# open a persistent application connection
# and login with Luna HSM challenge

salogin -c -s 1 -i 1:1
# close persistent application connection 1:1
# on slot 1
```

## Attempting to use salogin on an STC-enabled slot

```
lunacm:>slot list
```

```
Slot Id ->          0
Label ->           stc_ppso
Serial Number ->   1213429268189
Model ->           LunaSA
Firmware Version -> 7.0.1
Configuration ->   Luna User Partition With SO (PW) Signing With Cloning Mode
Slot Description -> Net Token Slot
```

```
Current Slot Id: 0
```

```
Command Result : No Error
```

```
lunacm:>stc status
```

```
Enabled:           Yes
Status:            Connected
Channel ID:        3
Cipher Name:       AES 256 Bit with Cipher Block Chaining
HMAC Name:         HMAC with SHA 512 Bit
```

```
Command Result : No Error
```

```
lunacm:>stc identityshow
```

```
Client Identity Name:      mySTCclientID
Public Key SHA1 Hash:     58feec48e485762c39a8c32f94cf535bf545699e
List of Registered Partitions:
```

Partition Identity Label	Partition Serial Number	Partition Public Key SHA1 Hash
par1	1213429268189	d4d4d65d281fd17580c56ddf09439c79c466a09a

Command Result : No Error

```
lunacm:>clientconfig listservers
```

Server ID	Server	Channel	HTL Required
0	192.20.11.184	STC	no

Command Result : No Error

```
lunacm:>exit
```

```
# ./salogin -o -s 0 -i 1:1 -p userpin
CA_OpenApplicationID: failed to open application id. err 0x80000030
token not present or app id already open?
```

## Other options

For java applications you could consider the KeyStore interface. It is internally consistent with the service provider interface defined by SUN/Oracle and does not require any proprietary code or applications.

If you are using an integration that does not refer to a KeyStore then the `salogin` method might be required. You are then limited to working with 1 partition. The type of HSM doesn't matter, as long as it is SafeNet and visible by the client at the time that the library is initialized.

This chapter describes how to use the **scp** (Linux/UNIX) or **pscp** (Windows) utilities to transfer files between a SafeNet Luna HSM client computer and a SafeNet Luna Network HSM appliance. It contains the following sections:

- "Using the scp and pscp Utilities" below

## Using the scp and pscp Utilities

Use the **scp** (Linux/Unix) or **pscp** (Windows) command to securely move updates and certificates and other files from a source computer onto the SafeNet appliance, or to move appliance certificates or log files out to a client computer.

All packages from SafeNet are signed and encrypted and come with an authorization code (authcode) that must be provided to decrypt and use the package.



**Note:** In Windows, use PSCP.exe (provided with SafeNet Luna HSM Client software) to transfer certificates, software updates, capability updates, logs, etc. The command syntax is similar to scp. PuTTY and PSCP have their own help.

## Syntax

### Client to appliance

```
scp [options] [<user>@]<host>:<source> <target>
```

### Appliance to client

```
scp [options] <source> [<source>...] [<user>@]<host>:<target>
```

### List files on the appliance

```
scp [options] -ls <user>@<host>:<file_path>
```

## Options

- p [] preserve file attributes.
- q [] quiet, don't show statistics
- r [] copy directories recursively
- S [<path-to-ssh>] specify the location of SSH
- v [] show verbose messages
- P port [] connect to specified port

- pw **passwd** [] login with specified password
- unsafe [] allow server-side wildcards (DANGEROUS)

## Examples

The following examples illustrate how to transfer files from a SafeNet Luna HSM client to a SafeNet Luna Network HSM, and from a SafeNet Luna Network HSM to a SafeNet Luna HSM client.

### Transferring a file from a SafeNet Luna HSM client to a SafeNet Luna Network HSM

```
/usr/safenet/lunaclient/>scp test-file.txt admin@myluna:
admin@myluna's password: *****
test-file.txt          |                0 kB |   0.1 kB/s | ETA: 00:00:00 | 100%

/usr/safenet/lunaclient/>
```

The colon is required. Type nothing after the colon when moving files onto the SafeNet appliance. All files that are scp'd to the SafeNet appliance go to a predetermined directory, which you cannot change (for security reasons). While it is possible to change the filename during scp (by typing a new filename after the colon in the scp command), this is not recommended since most operations expect certain filenames and can fail if those are not found.

```
/usr/safenet/lunaclient/>scp test-file.txt admin@myluna:different-file.txt
admin@myluna's password: *****
test-file.txt          |                0 kB |   0.1 kB/s | ETA: 00:00:00 | 100%

/usr/safenet/lunaclient/>
```

If the arriving file carries an unexpected name, it might not be handled correctly by subsequent commands

If you have SSH located in a non-standard (UNIX) location, launch the scp command with the "-S" option (that's an uppercase "s"), followed by the path to SSH, before supplying the paths to the source and target files, like:

```
scp -S /usr/bin/ssh <source file> <dest file>
```

### Transferring a file from a SafeNet Luna Network HSM to a SafeNet Luna HSM client

```
bash-2.05# scp admin@myLuna3:server.pem .
admin@myLuna3's password: *****
server.pem          100%
|*****| 928
00:00
```

Note the dot (.) at the end of the command, denoting "place the resulting file in the current directory".

This chapter describes how to use the VTL utility to manage the relationship between your Client computer and one or more SafeNet appliances.



**Note:** VTL is a legacy utility that is included for backwards compatibility reasons only. We strongly recommend that you discontinue use of VTL and use LunaCM or LunaSH instead.

---

This chapter contains the following topics:

- ["VTL Overview" on the next page](#)
- ["vtl addServer " on page 80](#)
- ["vtl backup" on page 81](#)
- ["vtl cklogsupport" on page 94](#)
- ["vtl createCert" on page 95](#)
- ["vtl deleteServer " on page 97](#)
- ["vtl examineCert " on page 98](#)
- ["vtl fingerprint " on page 100](#)
- ["vtl haAdmin" on page 101](#)
- ["vtl listServers " on page 122](#)
- ["vtl listSlots " on page 123](#)
- ["vtl logging configure " on page 124](#)
- ["vtl logging show " on page 125](#)
- [" vtl replaceserver" on page 126](#)
- ["vtl supportInfo " on page 127](#)
- ["vtl verify " on page 128](#)

## VTL Overview

VTL stands for "Virtual Token Library", and is a command-line utility that is loaded onto each of your Client computers when you install the SafeNet Software.

Open a command prompt window or console, cd to the directory where you installed your SafeNet software, and run the **vtl** command (with the -h option, to see the available sub-commands).

These are the commands that you use to manage the relationship between your Client computer and one or more SafeNet appliances. You must have Administrator privileges on your own computer (the computer that you are using as a client to the SafeNet Luna Network HSM). If you do not also have authority on the SafeNet device(s), then you need the co-operation of the person who holds that authority.

```
admin@mycomputer:~>vtl
usage: (select command -h for additional information)
```



**Note:** You need to be Administrator (or equivalent) when running vtl commands that need to access /etc and /user (and the equivalents in Windows).

## Subcommands

Subcommand	Description
addServer	Use this command to add a server to the client's list of trusted SafeNet Luna Network HSM Servers (you need to have already imported the server certificate from each SafeNet Luna Network HSM that you wish to add). See " <a href="#">vtl addServer</a> " on page 80.
deleteServer	Use this command to remove a server/host from the client's list of trusted SafeNet Luna Network HSM Servers. See " <a href="#">vtl deleteServer</a> " on page 97.
replaceServer	Use this command to replace a named server/host from the client's list of trusted SafeNet Luna Network HSM Servers with a new named server/host - requires the original server's name [-o], the replacing server's name [-n], and the path to the new server's certificate file [-c]. See " <a href="#">vtl replaceserver</a> " on page 126.
listServers	Use this command to display a list of the SafeNet Luna Network HSM Servers trusted by this client. See " <a href="#">vtl listServers</a> " on page 122.
createCert	Use this command to create (or re-create) the client's certificate and private key that are used for NTLS (Network Trust Link Service). See " <a href="#">vtl createCert</a> " on page 95.
listSlots	List all PKCS#11 cryptographic device slots that can be seen at this time. See " <a href="#">vtl listSlots</a> " on page 123.
verify	Use this command to verify the SafeNet Luna Network HSM Servers slots or partitions that are visible. See " <a href="#">vtl verify</a> " on page 128.
haAdmin	Use these commands to create and manage HA groups of several SafeNet Luna Network HSM appliances, providing load-sharing and redundancy for the cryptographic operations required by this client. See " <a href="#">vtl haAdmin</a> " on page 101
fingerprint	Use this command to display the fingerprint of a specified certificate. See " <a href="#">vtl fingerprint</a> " on

Subcommand	Description
	<a href="#">page 100</a> .
examineCert	Use this command to display the details of a specified certificate. See " <a href="#">vtl examineCert</a> " on <a href="#">page 98</a> .
backup	Administer backup/backup HSM features - used for Remote Backup (where the backup HSM is connected to a computer remote from your SafeNet Luna Network HSM. If you connect the SafeNet Luna Backup HSM directly to your SafeNet Luna Network HSM, then use the SafeNet Shell command line and "token backup" commands, instead of VTL. See " <a href="#">vtl backup</a> " on <a href="#">page 81</a> .
supportInfo	Use this command to create a support information file, when one is requested by SafeNet Customer Support. See " <a href="#">vtl supportInfo</a> " on <a href="#">page 127</a> .
logging	Configure logging for Windows computers. See " <a href="#">vtl logging configure</a> " on <a href="#">page 124</a> .

---

## vtl addServer

---

### Name

vtl addServer

### Syntax

```
vtl addServer -n <server_IP_or_hostname> -c <server_cert_filename>
```

### Description

Adds the specified server to the client's list of trusted SafeNet Servers. You may wish to check the fingerprint of the server certificate with the vtl fingerprint command before adding it. The server certificate is one that you have imported from the SafeNet appliance to your Client computer, using scp.

You must be Administrator on your Client computer, or logged in as a user with Administrator privileges.

### Options

**-n <server hostname>** [mandatory] The hostname (or IP address) of the server to add. Use the IP address if the server's certificate uses its ip address instead of its hostname. If you are uncertain what format the server's certificate uses, contact your SafeNet appliance administrator, or look for the "CN=" field when using the vtl examineCert command.

**-c <certificate file>** [mandatory] The name (including the path to its location on your computer) of the server's certificate file. Use the 'scp' utility to collect the server's certificate from the SafeNet appliance, or use the certificate provided by your SafeNet appliance administrator. You may wish to confirm the authenticity of the certificate by using the vtl fingerprint command.

### Example

```
$ ./vtl add -n 192.20.9.161 -c server161.pem  
New server 192.20.9.161 successfully added to server list..
```



## vtl backup



**Note:** This command will be deprecated in a future release. It is strongly recommended that you use the `lunacm` utility to backup your SafeNet Luna Network HSM partitions. See "[partition backup](#)" on page 1 in the *Lunacm Command Reference Guide*.

### Name

`vtl backup` - backup SafeNet Luna Network HSM partition to slot

### Syntax

```
vtl backup -source <slot# or label> -p <source password> -target <slot# or label> -partition <backupHSM
partitionname> -r <backupHSM SO password> -u <backup partition user password>
```

### Description

This command is used **remotely** to back up SafeNet Luna Network HSM partition contents to a specified slot or labeled partition on a backup HSM.

See also:

- "[vtl backup append](#)" on page 83
- "[vtl backup restore](#)" on page 86
- "[vtl backup delete](#)" on page 85
- "[vtl backup token](#)" on page 87

If you use this command without any sub-commands, a partition is created on the backup HSM and the objects on the source HSM slot are cloned to the target partition. Or, if the target partition is found, the objects on the source HSM slot are cloned to the target slot, overwriting any objects already there.

If you wish to preserve objects already on the target partition, then use `vtl backup append` to add the objects from the source to those already on the target.

If you wish to back up your SafeNet Luna Network HSM partition to a SafeNet Luna Backup HSM that is connected **locally** to the SafeNet Luna Network HSM appliance, then use the [token backup](#) commands instead.

### Options

**-source <slot # or label>** [mandatory] The slot number or the label of the source HSM. Do NOT use a numeral as the first character in a slot label. The command looks for a slot number first, and any numeral it sees is interpreted as a slot number.

**-p <source password>** source user password, if needed

**-target <slot # or label>** [mandatory] The slot number or the label of the target HSM. Do NOT use a numeral as the first character in a slot label. The command looks for a slot number first, and any numeral it sees is interpreted as a slot number.

**-partition <backup partition name>** [mandatory] The name of the target partition on the Backup HSM.

**-r <backup HSM SO password>** The SO password of the Backup HSM, needed for password-authenticated HSMs only - ignored for PED-authenticated HSMs.

**-u <backup HSM User password>** The User password of the Backup HSM partition, needed for password-authenticated HSMs only - ignored for PED-authenticated HSMs.

## Example

```
bash # ./vtl backup -source 5 -target 1 -partition bck1
Backup partition 'bck1' does not exist on G5backup2.
*** Are you sure you want to create the new backup? [yes/no]: yes
Luna PED operation required to login to p1 - use User or Partition Owner (black) PED key
Please enter the secret challenge: userpin
Luna PED operation required to login to G5backup2 - use Security Officer (blue) PED key

Luna PED operation required to login to G5backup2 - use User or Partition Owner (black) PED key
Luna PED operation required to set legacy cloning domain - use Domain (red) PED key.

Luna PED operation required to login to G5backup2 - use User or Partition Owner (black) PED key
21 objects found on p1/ (slot #5).
Cloning object 0 - success
Cloning object 1 - success
Cloning object 2 - success
Cloning object 3 - success
Cloning object 4 - success
Cloning object 5 - success
Cloning object 6 - success
Cloning object 7 - success
Cloning object 8 - success
Cloning object 9 - success
Cloning object 10 - success
Cloning object 11 - success
Cloning object 12 - success
Cloning object 13 - success
Cloning object 14 - success
Cloning object 15 - success
Cloning object 16 - success
Cloning object 17 - success
Cloning object 18 - success
Cloning object 19 - success
Cloning object 20 - success
21 objects successfully backed up.
bash #.
```

## vtl backup append

### Name

vtl backup append

### Syntax

```
vtl backup append -source <slot# or label> -p <source password> -target <slot# or label> -partition <backupHSM
partitionname> -r <backupHSM SO password> -u <backup partition user password>
```

### Description

This command is used **remotely** to back up SafeNet Luna Network HSM partition contents to a specified slot or labeled partition on a backup HSM.

See also:

- ["vtl backup" on page 81](#)
- ["vtl backup restore " on page 86](#)
- ["vtl backup delete " on page 85](#)
- ["vtl backup token" on page 87](#)

If you wish to preserve objects already on the target partition, then use vtl backup append to add the objects from the source to those already on the target.

If you wish your cloned objects to overwrite any objects already on the target slot, then use vtl backup (without the append sub-command).

If you wish to back up your SafeNet Luna Network HSM partition to a SafeNet Luna Backup HSM that is connected **locally** to the SafeNet Luna Network HSM appliance, then use the [token backup](#) commands in LunaSH instead.

### Options

**-source <slot # or label>** [mandatory] The slot number or the label of the source HSM. Do NOT use a numeral as the first character in a slot label. The command looks for a slot number first, and any numeral it sees is interpreted as a slot number.

**-p <source password>** source user password, if needed

**-target <slot # or label>** [mandatory] The slot number or the label of the target HSM. Do NOT use a numeral as the first character in a slot label. The command looks for a slot number first, and any numeral it sees is interpreted as a slot number.

**-partition <backup partition name>** [mandatory] The name of the target partition on the Backup HSM.

**-r <backup HSM SO password>** The SO password of the Backup HSM, needed for password-authenticated HSMs only - ignored for PED-authenticated HSMs.

**-u <backup HSM User password>** The User password of the Backup HSM partition, needed for password-authenticated HSMs only - ignored for PED-authenticated HSMs.

## Example

```
bash # ./vtl backup append
  -source 5 -target 1 -partition bck1
Container 'bck1' already exist on G5backup2..
Luna PED operation required to login to p1 - use User or Partition Owner (black) PED key
Please enter the secret challenge: userpin
Luna PED operation required to login to G5backup2 - use Security Officer (blue) PED key

Luna PED operation required to login to G5backup2 - use User or Partition Owner (black) PED key
Luna PED operation required to set legacy cloning domain - use Domain (red) PED key.

Luna PED operation required to login to G5backup2 - use User or Partition Owner (black) PED key
25 objects found on p1/ (slot #5).
21 objects found on G5backup2 (slot #1).
Cloning object 21 - success
Cloning object 22 - success
Cloning object 23 - success
Cloning object 24 - success
4 objects successfully backed up.
bash #.
```

---

## vtl backup delete

---

### Name

vtl backup delete

### Syntax

vtl backup delete -target <slot# or label> -partition <backupHSM partitionname>

### Description

This command is used remotely to delete backed-up SafeNet Luna Network HSM partition contents from a specified slot or labeled partition on a backup HSM.

See also:

- ["vtl backup" on page 81](#)
- ["vtl backup restore " on the next page](#)
- ["vtl backup append" on page 83](#)
- ["vtl backup token" on page 87](#)

### Options

**-target <slot # or label>** [mandatory] The slot number or the label of the target HSM. Do NOT use a numeral as the first character in a slot label. The command looks for a slot number first, and any numeral it sees is interpreted as a slot number.

**-partition <backup partition name>** [mandatory] The name of the target partition on the Backup HSM.

### Example

```
\LunaSA>vtl backup delete -t mylunabackup -p mylunapar1
*** Are you sure you want to delete mylunapar1 partition? [yes/no]: yes
Luna PED operation required to login to mylunabackup - use Security Officer (blue) PED key
Partition 'mylunapar1' deleted.
\LunaSA>
```

## vtl backup restore

### Name

vtl backup restore

### Syntax

```
vtl backup restore -source <slot# or label> -partition <backupHSM partitionname> -r <backupHSM SO password> -u
<backup partition user password> -target <slot# or label> -p <source password>
```

### Description

This command is used remotely to restore SafeNet Luna Network HSM partition contents from a specified slot or labeled partition on a backup HSM. See also:

["vtl backup" on page 81](#)

["vtl backup append" on page 83](#)

["vtl backup delete " on the previous page](#)

["vtl backup token" on the next page](#)

### Options

**-source <slot # or label>** [mandatory] The slot number or the label of the source HSM. Do NOT use a numeral as the first character in a slot label. The command looks for a slot number first, and any numeral it sees is interpreted as a slot number.

**-partition <backup partition name>** [mandatory] The name of the target partition on the Backup HSM.

**-r <backup HSM SO password>** The SO password of the Backup HSM, needed for password-authenticated HSMs only - ignored for PED-authenticated HSMs.

**-u <backup HSM User password>** The User password of the Backup HSM partition, needed for password-authenticated HSMs only - ignored for PED-authenticated HSMs.

**-target <slot # or label>** [mandatory] The slot number or the label of the target HSM. Do NOT use a numeral as the first character in a slot label. The command looks for a slot number first, and any numeral it sees is interpreted as a slot number.

**-p <source password>** source user password, if needed

### Example

```
vtl backup restore -source 2 -partition mylunapar1 -target 1
Luna PED operation required to login to G5 - use User or Partition Owner (black) PED key
Luna PED operation required to login to mylunapar1 - use User or Partition Owner (black) PED key
Please enter the secret challenge: *****
2 objects found on source (slot #2)
Cloning object 0 - success
Cloning object 1 - success
2 objects restored.
```

## vtl backup token

### Name

vtl backup token

### Subcommands and Usage

Subcommand	Description
factoryreset	Factory reset for backup token. See <a href="#">"vtl backup token factoryreset "</a> on the next page.
init	Initialize backup token. See <a href="#">"vtl backup token init "</a> on page 89.
resize	Resize backup token container. See <a href="#">"vtl backup token resize "</a> on page 90.
show licenses	Show all licenses for backup token. see <a href="#">"vtl backup token show licenses "</a> on page 92.
show	Show all SafeNet UHD slots for backup token. See <a href="#">"vtl backup token show "</a> on page 91
update	Update backup token. See <a href="#">"vtl backup token update "</a> on page 93.

## vtl backup token factoryreset

### Name

vtl backup token factoryreset - factory reset a backup HSM.

### Syntax

vtl backup token factoryreset -target <slot# or label>

### Description

This command factory-resets a backup HSM connected to the SafeNet Luna Network HSM appliance.

### Options

Option	Description
-target <token>	(mandatory) label or slot number for the target token/HSM

### Example

If the Backup HSM has not been initialized since the last factory reset:

```
bash # ./vtl backup token factoryreset -target 1
*** Are you sure you wish to reset this HSM to factory default settings? [yes/no]: yes
Token is already zeroized
Error (RC_GENERAL_ERROR)
bash #
```

If the Backup HSM has been initialized since the last factory reset:

```
bash # ./vtl backup token factoryreset -target 1
*** Are you sure you wish to reset this HSM to factory default settings? [yes/no]: yes
'factory reset' successful.
bash #
```



## vtl backup token init

### Name

vtl backup token init - initialize a backup HSM.

### Syntax

```
vtl backup token init -target <slot# or label> [ -label <label> ]
```

### Description

This command initializes a backup HSM connected to the SafeNet Luna Network HSM appliance.

### Options

Option	Description
-target <token>	(mandatory) label or slot number for the target token/HSM
-label <label>	(optional new label name for target token/HSM

### Example

```
bash # ./vtl backup token init -target 1
*** Are you sure you wish to initialize this HSM? [yes/no]: yes
This HSM can be initialized to use either PED or password authentication.
*** Use PED authentication? [yes/no]: yes
Please enter the new label: G5backup2
Luna PED operation required to initialize backup token - use Security Officer (blue) PED key.
Luna PED operation required to login to no label - use Security Officer (blue) PED key
Luna PED operation required to set legacy cloning domain - use Domain (red) PED key.
'init' successful.
```

## vtl backup token resize

### Name

vtl backup token resize - resize backup token container.

### Syntax

```
vtl backup token resize -target <slot# or label> -container <container> [ -size <size> ]
```

### Description

This command resizes a backup HSM partition (connected to the SafeNet Luna Network HSM appliance).

### Options

Option	Description
-target <token>	(mandatory) label or slot number for the target token/HSM
-container <container>	(mandatory) target token container (partition) name
-size <size>	(optional) target token container size in bytes

### Example

```
bash # ./vtl backup token resize
      -target 1 -container backuppar
*** Are you sure you wish to resize this HSM? [yes/no]: yes
.
'resize' successful.
```

## vtl backup token show

### Name

vtl backup token show - show backup HSM slots or slot info.

### Syntax

vtl backup token show -target <slot# or label>

### Description

This command shows a summary of slots associated with a backup HSM connected to the SafeNet Luna Network HSM appliance, or shows slot info for a named slot.

### Options

Option	Description
-target <token>	(optional) label or slot number for the target token/HSM

### Example

```
bash # ./vtl backup token show
The following Luna UHD slots were found:
Slot #   Label       Serial #       Description     Status
slot #1  no label     7000179       Luna UHD slot  Present
slot #2  -            -              -              Not present
slot #3  -            -              -              Not present
bash #
```

## vtl backup token show licenses

### Name

vtl backup token show licenses - show licenses for a backup HSM .

### Syntax

vtl backup token show licences -target <slot# or label>

### Description

This command shows a summary of licenses associated with a backup HSM connected to the SafeNet Luna Network HSM appliance.

### Options

Option	Description
-target <token>	(optional) label or slot number for the target token/HSM

### Example

```
bash # ./vtl backup token show licenses
```

```
HSM CAPABILITY LICENSES
License ID      Description
621010355-000  621-010355-000 G5 Backup Device Base
621000005-001  621-000005-001 Backup Device Partitions 20
621000006-001  621-000006-001 Backup Device Storage 15.5 MB
621000007-001  621-000007-001 Backup Device Store MTK Split Externally
621000008-001  621-000008-001 Backup Device Remote Ped Enable
```

```
bash #
```

## vtl backup token update

### Name

vtl backup token update - update firmware or capability of a backup HSM.

### Syntax

```
vtl backup token update firmware -target <slot# or label>
```

```
vtl backup token update capability -target <slot# or label>
```

### Description

This command updates the firmware or the capabilities of a backup HSM. The firmware update file (fuf) or capability update file (cuf) must be ready to install.

### Options

Option	Description
-target <token>	(mandatory) label or slot number for the target token/HSM

### Example

```
bash # ./vtl backup token update firmware -target 1
Please enter firmware update file (fuf) name: lunasa_6.2.1_firmware_update.fuf
This command updates the token firmware. This process cannot be reversed.
Are you sure you want to update firmware? [yes/no]: yes
update firmware' successful.
```

---

## vtl cklogsupport

---

### Name

vtl cklogsupport

### Syntax

vtl cklogsupport {enable | disable}

### Description

Enable or disable CKLOG support. CKLOG is a facility which can record all interactions between an application and our PKCS#11-compliant library. It allows a developer to debug an application by viewing what the library receives. See "[Libraries and Applications](#)" on page 1 in the *SDK Reference Guide* for more information.

### Options

**enable** Enable CKLOG.

**disable** Disable CKLOG.

### Example

```
$ ./vtl cklogsupport enable
Chrysoki2 LibUNIX = /usr/lib/libCryptoki2.so
Cklog not enabled
Enabling cklog
```

## vtl createCert

### Name

vtl createCert

### Syntax

**vtl createCert -n** <common name/server hostname> [-**c** <country code>] [-**s** <state>] [-**l** <locality>] [-**o** <organization name>] [-**u** <organization unit name>] [-**e** <e-mail address>] [-**P** <private key out filename>] [-**C** <certificate out filename>] [-**d** <certificate validity period>] [-**v**]

### Description

Creates the client's certificate and private key that are used by NTLS. Re-creates the key and certificate if they already exist.



**CAUTION:** If the key and certificate are re-created, the client will need to be removed and re-registered from each of the SafeNet servers with which it was registered.



**Note:** The server hostname (-n) is the only mandatory field for certificate creation. This is because all other fields of the certificate are used simply for display and visual confirmation purposes. The NTLA never displays certificate data fields to the user, so the content in these fields is irrelevant.

### Options

- n** <**server hostname**> [mandatory] The hostname (or IP address) of the server to add.
- c** <**country**> [optional] The country in which the client computer resides. (Data not used.)
- s** <**state**> [optional] The state in which the client computer resides. (Data not used.)
- l** <**locality**> [optional] The city/locality in which the client computer resides. (Data not used.)
- o** <**organization**> [optional] The organization to which the client computer belongs. i.e. SafeNet-inc (Data not used.)
- u** <**organizational unit**> [optional] The unit within the organization to which the client belongs. i.e. Engineering, or IT (Data not used.)
- e** <**e-mail**> [optional] An E-mail address for the certificate. (Data not used.)
- P** <**private key outfile name**> [optional - default filename is <hostname/ip>Key.pem] A filename for the private key to be created. Only use this switch if you have a need to override the default value.
- C** <**certificate outfile name**> [optional - default filename is <hostname/ip>.pem] A filename for the certificate to be created. Use this switch only if you have a need to override the default value.
- d** <**certificate validity period**> [optional - default is 3650, or 10 years] Specifies the validity period for the client certificate, in days.
- v** [optional] Verbose mode. Output extra information while creating the certificate and private key.

## Example

### Windows

```
vtl createCert -n test
Private Key created and written to: E:\temp\clientCerts\testKey.pem
Certificate created and written to: E:\temp\clientCerts\test.pem

vtl createCert -n test -v
Using configuration from C:\Program Files\SafeNet\LunaClient\openssl.cnf
It needs to be at least 1024
Writing new private key to stdout E:\temp\clientCerts\testKey.pem'
CA [CA]:CA
Ontario [Ontario]:Ontario
Ottawa [Ottawa]:Ottawa
My company [My company]:My company
[]:
test [test]:test
[]:
Private Key created and written to: E:\temp\clientCerts\testKey.pem
Certificate created and written to: E:\temp\clientCerts\test.pem
```

### UNIX

```
vtl createCert -n test
Private Key created and written to: /usr/safenet/lunaclient/cert/client/testKey.pem
Certificate created and written to: /usr/safenet/lunaclient/cert/client/test.pem
```



---

## vtl deleteServer

---

### Name

vtl deleteServer

### Syntax

```
vtl deleteServer -n <serverhostname>
```

### Description

Removes the given host from the list of trusted SafeNet Servers. View a list of all trusted servers with the command `vtl listServers`.

### Options

**-n <server hostname>** [mandatory] The hostname (or IP address) of the SafeNet server to add. Use the IP address if the server's certificate uses its ip address instead of its hostname. Use the `vtl listServers` command for a list of trusted servers.

### Example

```
vtl delete -n LunaSA1  
Server lunasa1 successfully removed from server list.
```

## vtl examineCert

### Name

vtl examineCert - Certificate details for this client

### Syntax

```
vtl examineCert [-f <filespec-of-serverCert.pem>] [-h]
```

### Description

Displays the details of the specified certificate. If the command is issued with no additional parameters, it returns the client certificate. If the -f option is used, then a filespec is required, and the command returns the details of the indicated certificate.

### Options

**-f** [optional] Requires the filespec to the certificate file (usually a servercert). The server cert files are located in the cert/server (or cert\server) directory, and are of the form "nameCert.pem", where "name" is the name that you gave when you ran the vtl addServer command.

**-h** [optional] The "help" text.

<nothing/default> display's the client's certificate

### Example

#### Windows

```
C:\Program Files\SafeNet\LunaClient>vtl examineCert -f cert\server\bigCert.pem
Certificate:
Data:
Version: 3 (0x2)
Serial Number: 0 (0x0)
Signature Algorithm: sha256WithRSAEncryption
Issuer: C=CA, ST=Ontario, L=Ottawa, O=Chrysalis-ITS, CN=168.0.1.0
Validity
  Not Before: Nov 10 14:10:36 2011 GMT
  Not After : Nov 11 14:10:36 2021 GMT
Subject: C=CA, ST=Ontario, L=Ottawa, O=Chrysalis-ITS, CN=168.0.1.0
Subject Public Key Info:
  Public Key Algorithm: rsaEncryption
  RSA Public Key: (2048 bit)
  Modulus (2048 bit):
    00:a9:c3:db:59:33:b8:65:20:c9:13:f7:a7:e5:59:
  7b:12:a4:31:d3:62:36:9a:62:68:6e:1d:d7:c7:f0:
  8c:fd:06:43:f8:42:f7:8c:de:74:d1:38:a3:8f:37:
  94:c4:82:cc:67:d8:51:14:cd:e4:b7:dd:f8:ff:09:
  c8:03:f9:62:c5:ad:fc:4d:2e:fe:67:dd:6b:e7:de:
  bd:9e:bd:92:14:63:a6:99:2a:78:e7:72:6d:ba:79:
  3d:55:a8:a4:5d:85:11:36:9f:3d:4c:9a:e6:e8:bf:
  b4:5b:45:83:46:c4:2c:d9:22:fa:50:5a:28:ba:6e:
  2f:cb:2f:54:47:8d:3b:fd:73:bc:5a:ce:cd:bb:4e:
  ec:b5:1c:87:b6:b1:cd:53:77:f0:f2:36:e9:b2:3d:
```

```
2e:61:6f:f2:73:c6:ad:c4:d4:fe:20:3b:de:e8:a9:
a4:cd:93:17:0a:65:a5:58:ef:e3:11:d5:f0:ac:92:
af:33:dc:1c:c0:8f:04:fc:13:53:65:7f:52:34:07:
71:7a:9b:e5:d8:1e:e0:bd:ca:13:0f:f9:00:33:e5:
2a:0c:79:78:42:ff:4c:1a:d6:83:2c:ae:bf:2d:1d:
93:ac:f5:6b:60:97:ab:fb:1a:d5:86:2c:2f:3c:f6:
7e:37:8d:77:0a:7a:dd:7c:38:61:26:9a:c9:c0:0d:
b3:57
```

Exponent: 65537 (0x10001)

Signature Algorithm: sha256WithRSAEncryption

```
15:49:31:22:c4:1a:80:9f:2d:de:4b:df:63:b8:b0:16:b0:af:
7a:f4:8f:62:0b:ad:fa:21:b5:95:6e:fc:a6:09:b9:f9:5f:ea:
8e:c8:a7:d5:90:0b:12:ff:a6:34:b5:9a:02:7f:81:66:38:21:
c7:92:21:a2:d4:0f:e9:44:84:2a:f5:ea:d2:00:4b:f1:0f:d5:
55:5b:15:3e:b4:b5:b6:d4:32:7d:fe:8c:ef:80:ef:f8:dd:73:
e6:1e:a2:41:4c:8c:1d:c7:fa:2a:a9:25:ef:aa:29:8e:40:8e:
da:2a:3d:af:67:a7:7e:da:a9:76:6d:c6:10:e7:3a:5d:45:ac:
a0:f3:35:30:44:76:7c:b0:ce:61:19:0b:74:b1:3f:51:08:f9:
12:47:75:7c:33:0c:ee:02:d7:bb:48:10:6d:40:5b:fe:26:f2:
8f:28:0f:d9:2d:25:d9:af:49:44:b3:25:c6:cf:97:21:f0:3a:
0d:0e:41:30:34:56:e8:8d:6b:d6:36:fb:a9:79:e6:bc:dd:6b:
61:cf:98:01:c0:70:b2:81:41:1c:79:6e:58:47:e9:22:83:98:
9f:9f:62:87:e3:74:df:87:fe:0b:78:55:0f:1e:6e:56:21:b6:
0e:29:64:cb:75:de:90:82:bd:24:64:ef:db:8c:9b:5b:b4:7e:
86:61:89:64
```

The only difference for a UNIX client would be the path in the filespec.

---

## vtl fingerprint

---

### Name

vtl fingerprint

### Syntax

```
vtl fingerprint [-f <filespec-of-serverCert.pem>] [-h]
```

### Description

Displays the fingerprint of the specified certificate. If the command is issued with no additional parameters, it returns the client fingerprint. If the -f option is used, then a filespec is required, and the command returns the fingerprint of the indicated certificate.

### Options

**-f** [optional] Requires the filespec to the certificate file (usually a servercert). The server cert files are located in the server/cert (or server\cert) directory, and are of the form "nameCert.pem", where "name" is the name that you gave when you ran the vtl addServer command.

**-h** [optional] The "help" text.

### Example

```
vtl fingerprint  
Certificate fingerprint: 91:01:EC:BA:6A:31:19:69:CF:8D:1A:23:87:95:76:35.
```

## vtl haAdmin



**Note:** The HA vtl subcommands are no longer supported, and are disabled. Use the lunacm hagroup commands to create and administer an HA group of SafeNet application partitions for this Client. See ["hagroup" on page 1](#).

```
myname@mycomputer:~>vtl haAdmin
```

```
Usage: vtl haAdmin (parameters):
```

### Subcommands

Subcommand	Description
newGroup	Disabled. See lunacm command <a href="#">"hagroup creategroup" on page 1</a>
deleteGroup	Disabled. See lunacm command <a href="#">"hagroup deletegroup" on page 1</a> .
addMember	Disabled. See lunacm command <a href="#">"hagroup addmember" on page 1</a> .
standbyMembers	Disabled. See lunacm command <a href="#">"hagroup addstandby" on page 1</a> .
removeMember	Disabled. See lunacm command <a href="#">"hagroup removemember" on page 1</a> .
synchronize	Disabled. See lunacm command <a href="#">"hagroup synchronize" on page 1</a> .
recover	Disabled. See lunacm command <a href="#">"hagroup recover" on page 1</a> .
autoRecovery	Disabled. See lunacm command <a href="#">"hagroup retry" on page 1</a> .
HALog	Disabled. See lunacm command <a href="#">"hagroup halog" on page 1</a> .
HAOnly	Disabled. See lunacm command <a href="#">"hagroup haonly" on page 1</a> .
show	Disabled. See lunacm command <a href="#">"hagroup listgroups" on page 1</a> .

## vtl haAdmin addMember



**Note:** This command is disabled. See lunadm command "[hagroup addmember](#)" on page 1.

### Name

vtl haAdmin addMember - Add a member to an HA group

### Syntax

```
vtl haAdmin addMember -group <groupNum> -serialNum <SN> -password <password>
```

### Description

Add a member to a HA group that already exists on this client. If network replication is allowed on the partition, the new member will share the same HA key as the existing members in the group.

Use the command `vtl haAdmin -synchronize` to replicate any objects on other group members to the new member (only works if network replication is allowed - if network replication is not allowed, use the `lunash:> partition backup` and `partition restore` commands to manually copy objects among the HA group members.)

### Options

**-group <groupNum>** [mandatory] The HA group's number. Group numbers can be found using the `vtl haAdmin -listGroups` command.

**-serialNum <SN>** [mandatory] The serial number of the partition to add to the group. The partition's serial number can be obtained using 'partition -show' from the lunash, or by using C\_GetTokenInfo via a PKCS#11 application such as `ckdemo`. All partitions have unique serial numbers.

**-password <password>** [mandatory] The text password for the partitions. The password must be the same as other partitions in the group or an error will occur.

### Example

```
vtl haAdmin addMember -group testgroup -serial 66010002 -password testpassword
Member 66010002 successfully added to group testgroup. New group configuration is:
```

```
HA Group Label: testgroup
HA Group Number: 165010001
HA Group Slot#: 6
Synchronization: enabled
Group Members: 65010001, 66010002
Standby members: <none>
```

```
In sync: yes
```

Please use the command 'vtl haAdmin -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.)



**Note:** Notice here that you are told to synchronize - do NOT synchronize if you intend to perform a recovery. Synchronization occurs automatically when you use the `vtl haAdmin recover` command.

---

---

## vtl haAdmin autoRecovery

---



**Note:** This command is disabled. See lunacm command "[hagroup retry](#)" on page 1.

---

### Name

vtl haAdmin autoRecovery - Set the autorecovery interval and retries.

### Syntax

```
vtl haAdmin autorecovery [ -retry <retry count> | -interval <seconds> ]
```

### Description

Set the HA autoRecovery retry count value - a positive value between 0 and 500 (or -1 for infinite retries), and the polling interval for those retries.

### Options

**-retry <retry count>** [one or the other of '-retry' or '-interval' must be used] The number of times HA function will attempt to automatically recover a member that has failed to synchronize or has dropped from the HA group. Setting to a value of zero switches the feature off. Any other positive value (up to 500) switches it on. A value of -1 means infinite retry attempts can be made.

**-interval <seconds>** [one or the other of '-retry' or '-interval' must be used] The interval at which the HA function will attempt to automatically recover a member that has failed to synchronize or has dropped from the HA group. Set the polling interval between 60 seconds and 1200 seconds.

### Example

```
C:\Program Files\SafeNet\LunaClient> vtl haAdmin autorecovery -retry 9
```

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



## vtl haAdmin deleteGroup



**Note:** This command is disabled. See lunacm command "[hagroup deletigroup](#)" on page 1.

### Name

vtl haAdmin deleteGroup - delete an HA group.

### Syntax

```
vtl haAdmin deleteGroup -group <groupNumber> -password <password>
```

### Description

Delete the specified HA group. After a group is deleted, it will no longer appear in the slot list in PKCS#11 applications. During the delete, the application attempts to login to each partition and remove the HA key from it. If the NTLA is not correctly set up or if the user no longer has access to one or more of the partitions in the group, a warning message indicates that the HA key was not successfully removed.

---

**CAUTION:** Do not use this command when an HA group is shared among multiple clients, because the `-deleteGroup` command deletes the HA Key material, which is still required by the other clients. The other clients would find that their HA group had been destroyed.

If you wish to remove a client from an HA group where other clients continue to share the HA group, then edit the `Chrystoki.conf` or `crystoki.ini` file on that client and remove the "VirtualToken" section. (Never insert TAB characters into the `chrystoki.ini` (Windows) or `crystoki.conf` (UNIX) file.)



At that point, you still have an NTL connection which no longer sees the HA virtual Partition, but now sees the individual HSM Partitions on the SafeNet Luna HSM.

You **MUST NOT** use the individual Partitions (from the HA virtual Partition), or the other clients will find their HA out-of-sync.

What you can do is login to the SafeNet Luna HSM and de-register that client from those Partitions. You may then register other, non-HA partitions to that client without disturbing any remaining clients of the HA virtual partition.

---

### Options

**-group <groupNumber>** [mandatory] The HA group's designating number. Group numbers can be found using the `vtl haAdmin -listGroups` command.

**-password <password>** [mandatory] The text password for the partitions. (All share the same password.)

## Example

```
vtl haAdmin deleteGroup -group 165010001 -password testpassword
HA key removed from HA group member with serial number 65010001.
The HA group 165010001 was successfully deleted.
vtl haAdmin deleteGroup -group 165010001 -password testpassword
Warning: This host is not assigned to a SafeNet Luna Network HSM partition with
the serial number 65010001, the HA key was not removed
from this group member.
The HA group 165010001 was successfully deleted.
```

## vtl haAdmin haLog



**Note:** This command is disabled. See lunacm command "[hagroup halog](#)" on page 1.

### Name

vtl haAdmin halog - set HA logging on or off.

### Syntax

```
vtl haAdmin halog -path <log-file_path> [-maxlen <maximum-log-file-length>
```

### Description

Set the HA logger on and logging to a file of 262144 bytes or larger, or set the HA logger off by specifying a log file length of 0 (zero).

### Options

**-enable** Enable HA log.

**-disable** Disable HA log.

**-path <log-file path>** [mandatory] The path to the location where the logfile is stored. If there is a space in the path, then put quote characters around the entire path string.

**-maxlen <maximum log-file length>** [optional] The maximum allowable size the logfile can reach before it is overwritten. Possible settings are zero, which disables the HA logger, or a number equal to, or greater than, 262144 bytes, which enables the HA logger.

### Example

```
C:\Program Files\SafeNet\LunaClient>vtl haadmin halog -path "C:\Program Files\SafeNet\Lun-
aClient"
C:\Program Files\Safenet\LunaClient>vtl haadmin show
===== HA Global Configuration Settings =====
HA Auto Recovery: disabled
Maximum Auto Recovery Retry: 0
Auto Recovery Poll Interval: 60 seconds
HA Logging: disabled
Only Show HA Slots: no
C:\Program Files\SafeNet\LunaClient>vtl haadmin -halog -enable
HA Log enabled
C:\Program Files\SafeNet\LunaClient>vtl haadmin -show
===== HA Global Configuration Settings =====
HA Auto Recovery: disabled
Maximum Auto Recovery Retry: 0
Auto Recovery Poll Interval: 60 seconds
HA Logging: enabled
HA Log File: C:\Program Files\SafeNet\LunaClient\haErrorLog.txt
Maximum HA Log File Length: 262144 bytes
Only Show HA Slots: no
C:\Program Files\SafeNet\LunaClient>vtl haadmin -halog -maxlen 1000000
C:\Program Files\SafeNet\LunaClient>vtl haadmin -show
```

---

```
===== HA Global Configuration Settings =====  
HA Auto Recovery: disabled  
Maximum Auto Recovery Retry: 0  
Auto Recovery Poll Interval: 60 seconds  
HA Logging: enabled  
HA Log File: C:\Program Files\SafeNet\LunaClient\haErrorLog.txt  
Maximum HA Log File Length: 1000000 bytes  
Only Show HA Slots: no
```

---

## vtl haAdmin HAOnly Client

---



**Note:** This command is disabled. See lunacm command "[hagroup haonly](#)" on page 1.

---

The HAOnly subcommands are used for creating and administering an HA group of SafeNet appliances for this Client.

```
myname@mycomputer:~>vtl haAdmin HAOnly
```

### Subcommands

Subcommand	Description
-enable	Show only HA slots when retrieving slot lists. See " <a href="#">vtl haAdmin HAOnly enable</a> " on page 111.
-disable	Show all slots when retrieving slot lists. See " <a href="#">vtl haAdmin HAOnly disable</a> " on the next page.



**Note:** This option affects all applications using this client.

---

---

## vtl haAdmin HAOnly disable

---

### Name

vtl haAdmin HAOnly - disable - disable showing of only HA virtual slots

### Syntax

```
vtl haAdmin HAOnly -disable
```

### Description

Configures the client to show all slots, rather than hide the physical slots and show only the HA virtual slots.

### Options

None

### Example

```
vtl haAdmin HAOnly - disable
```

```
HAOnly disabled.
```

## vtl haAdmin HAOnly enable

---

### Name

vtl haAdmin HAOnly - enable - show only HA virtual slots.

### Syntax

```
vtl haAdmin HAOnly -enable
```

### Description

Configures the client to show only the HA virtual slots, and not the individual physical slots that make up the HA group.

### Options

None

### Example

```
vtl haAdmin HAOnly - enable
```

```
HAOnly enabled.
```

## vtl haAdmin HAOnly show

---

### Name

vtl haAdmin HAOnly - show - show current status of HAOnly option.

### Syntax

```
vtl haAdmin HAOnly -show
```

### Description

Show the current status of the "HAOnly" display option.

### Options

None

### Example

```
vtl haAdmin HAOnly - show
```

This client is configured to show all slots.



## vtl haAdmin newGroup



**Note:** This command is disabled. See lunacm command "[hagroup creategroup](#)" on page 1

### Name

vtl haAdmin newGroup - create a new HA group.

### Syntax

```
vtl haAdmin newGroup -serialNum <serialnumber> -label <label> -password <password>
```

### Description

Creates a new High Availability (HA) group. The user selects a label for the new group, and provides a primary partition and its password. Using these, the HA group is set up and is ready for new partitions to be added to it. Note that the user must be assigned the partition in question, and the NTLA must be correctly established.

The new HA group will be assigned an HA group number that is used for all other commands associated with this HA group.

The new HA group will appear as an additional slot in the client machine's slot list. The slot will be denoted as an HA Virtual Card Slot slot-type when using C\_GetSlotInfo call

If there are any existing objects on the partition, the user is asked if he would like to keep them, remove them, or quit to further examine them.

If this new HA group is a copy of a group on another client, the user will be warned that there is an existing HA key on the partition. If the user's intention is to have both clients able to talk to the same set of partitions in HA groups, the user must type 'copy' to keep and use the existing HA key. (If the user removes it, the partition will no longer be a working member of the other HA group(s) to which it belongs.)

---

**CAUTION:** VTL manages the HA groups that you create, and must therefore remember each group and each member (serial number) that is used. You might wish to create a group, then create additional groups based on the configuration of the first one, by "re-using" the primary member - deleting that primary member from the first group and using it to start another group [a SafeNet Luna Network HSM can be a member of just one HA group at one time].



This can work for a second and a third HA group, but cannot be done for any additional HA groups (fourth, fifth, etc.) unless you remove an existing group before you attempt to create any new group.

The maximum concurrent HA groups administered by one vtl and re-using the same primary member is three.

To administer many HA groups, all started by the same SafeNet Luna Network HSM, from one administrative workstation, run Virtual Machine environments on that workstation with a separate instance of vtl in each VM.

---

## Options

**-serialNum <serialnumber>** [mandatory] The serial number of the primary partition for the group. The partition's serial number can be obtained using 'partition -show' from the lunash, or by using C\_GetTokenInfo via a PKCS#11 application such as ckdemo. All partitions have unique serial numbers.

**-label <label>** [mandatory] Provide a label for the new HA group. This is the value that will be returned to the PKCS#11 call C\_GetTokenInfo for the HA slot.

**-password <password>** [mandatory] The text password for the primary partition. Note that for SafeNet Luna Network HSM with Trusted Path Authentication partitions, all partitions that will be added to the HA group must share this password. You may wish to use the lunash command 'partition -changePw' to set the password before completing this step.

## Example

```
vtl haAdmin -newGroup -label testgroup -serial 65010001 -password testpassword
```

```
Warning: There are 2 objects currently on the new member.
```

```
Do you wish to propagate these objects within the HA
group, or remove them?
```

```
Type 'copy' to keep and propagate the existing
objects, 'remove' to remove them before continuing,
or 'quit' to stop adding this new group member.
```

```
> copy
```

```
New group with label "testgroup" created at group number 165010001.
```

```
Group configuration is:
```

```
HA Group Label: testgroup
```

```
HA Group Number: 1150520008
```

```
HA Group Slot #: unknown
```

```
Synchronization: enabled
```

```
Group Members: 150520008
```

```
Standby members: <none>
```

```
In Sync: yes
```

## Error When Attempting More than Three Groups

The following is an example of what happens if you are re-using a primary SafeNet Luna Network HSM to attempt to create a fourth HA group, without deleting any of the earlier groups. VTL on your administrative computer must keep track of all HA groups that it is managing, and it allows a maximum of three:

```
hbash-3.2# ./vtl haadmin -new -l ha4 -s 951357004
```

```
Please enter the password for the partition:
```

```
> *****
```

```
Warning: There are 119 objects currently on the new member.
```

```
Do you wish to propagate these objects within the HA
group, or remove them?
```

```
Type 'copy' to keep and propagate the existing
objects, 'remove' to remove them before continuing,
or 'quit' to stop adding this new group member.
```

```
> copy
```

```
Can not generate a unique serial number for the HA group.
```

```
You may want to delete any un-used HA group and try again.
```

```
'vtl haAdmin -newGroup' aborted.
```

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

---

## vtl haAdmin recover group

---



**Note:** This command is disabled. See lunacm command "[hagroup recover](#)" on page 1.

---

### Name

vtl haAdmin recover -group - recover an HA group.

### Syntax

vtl haAdmin recover -group <group name>

### Description

Allow a previously removed member to rejoin an HA group, or a new appliance to replace a member that failed and was removed. This command is required if autorecover is disabled.

### Options

**-group <groupName>** [mandatory] The HA group's name. Group names can be found using the *vtl haAdmin -show* command.

### Example

```
bin]# ./vtl haadmin recover -group testgroup
Signal sent to HA group: testgroup to recover!
bin]#
```

## vtl haAdmin removeMember



**Note:** This command is disabled. See lunacm command "[hagroup removemember](#)" on page 1.

### Name

vtl haAdmin removeMember - remove a member from an HA group.

### Syntax

```
vtl haAdmin removeMember -group <groupNum> -serialNum <serialnumber> -password <password>
```

### Description

Removes the specified member from the HA group. If the member is still accessible via the NTLA, the HA key will also be removed from the member, otherwise a warning will be printed that the key was not removed.

Currently the partition must be assigned to the user for the removeMember command to work. If it is not possible to assign the partition to the user, contact Technical Support to ask them how to manually remove the member from the HA group.



**CAUTION:** You should never manually remove members unless you absolutely must, because the application has many steps that cannot be replicated manually.

### Options

**-group <groupNumber>** [mandatory] The HA group's designating number. Group numbers can be found using the *vtl haAdmin -listGroups* command.

**-serialNum <serialNumber>** [mandatory] The serial number of the partition to remove from the group. Use the *vtl haAdmin -listGroups* command to see a list of partition serial numbers that belong to the each HA group.

**-password <password>** [mandatory] The text password for the partitions.

### Example

```
C:\Program Files\SafeNet\LunaClient>vtl haadmin removeMember -group testgroup -serial 150520009
-password default
Member 150520009 successfully removed from group testgroup. New group configuration is:
HA Group Label: testgroup
HA Group Number: 1150520008
HA Group Slot #: 6
Synchronization: enabled
Group Members: 150520008
Standby members: <none>
In Sync: yes
```

## vtl haAdmin show



**Note:** This command is disabled. See lunacm command "[hagroup listgroups](#)" on page 1.

### Name

vtl haAdmin show - show status of HA members.

### Syntax

vtl haAdmin show

### Description

Shows the cryptoki connectivity status of all currently configured HA members.

### Options

**-syncStatus** (optional) show sync status for each group. Prompt for group password

**-help** (optional) display this message

### Example

```
C:\Program Files\SafeNet\LunaClient>vtl haadmin -show
===== HA Global Configuration Settings =====
HA Auto Recovery: disabled
Maximum Auto Recovery Retry: 0
Auto Recovery Poll Interval: 60 seconds
HA Logging: enabled
HA Log File: C:\Program Files\SafeNet\LunaClient\haErrorLog.txt
Maximum HA Log File Length: 1000000 bytes
Only Show HA Slots: no
===== HA Group and Member Information =====
HA Group Label: testgroup
HA Group Number: 1150520008
HA Group Slot #: 6
Synchronization: enabled
Group Members: 150520008
Standby members: <none>
Slot #      Member S/N                Member Label      Status
=====
1           150520008                 zspar             alive
```

## vtl haAdmin standbyMembers



**Note:** This command is disabled. See lunacm command "[hagroup addstandby](#)" on page 1.

### Name

vtl haAdmin standbyMembers

### Syntax

vtl haAdmin standbyMembers [ -set | -remove ]

### Subcommands

Subcommand	Description
-set	Use a list of serial numbers to add partitions to a named HA group, on standby. See " <a href="#">vtl haAdmin standbyMembers -set</a> " on page 120.
-remove	Use a list of serial numbers to remove partitions as standby members from a named HA group. See " <a href="#">vtl haAdmin standbyMembers -remove</a> " on the next page.

---

## vtl haAdmin standbyMembers -remove

---

### Name

vtl haAdmin standbyMembers -remove - remove standby HA members

### Syntax

vtl haAdmin standbyMembers -remove

### Description

Use a list of serial numbers to remove standby members from a named HA group.

### Options

**-group <groupName>** [mandatory] The name of the HA group to modify.

**-serialNum <serial number list>** [mandatory] The serial numbers of the partitions to remove from the named group.

### Example

```
vtl haAdmin standbyMembers -remove -group 165010001 -serial 66010002
```

---

## vtl haAdmin standbyMembers -set

---

### Name

vtl haAdmin standbyMembers -set - add standby HA members

### Syntax

vtl haAdmin standbyMembers -set

### Description

Adds members, from a list of partition serial numbers, to a named HA group, in standby status.

### Options

**-group <groupName>** [mandatory] The name of the HA group to modify.

**-serialNum <serial number list>** [mandatory] The serial numbers of the partitions to add to the named group.

### Example

```
vtl haAdmin standbyMembers -set -group 165010001 -serialnum 66010002
```



## vtl haAdmin synchronize



**Note:** This command is disabled. See lunadm command "[hagroup synchronize](#)" on page 1.

### Name

vtl haAdmin synchronize - synchronize contents among HA group members.

### Syntax

```
vtl haAdmin synchronize [-enable] | [-disable] -group <groupNumber> -password <password>
```

### Description

Synchronizes the contents of members of the HA group that have network replication enabled. The contents of each partition will be examined with those of all others to ensure that all objects are found on all partitions. SHA-1 digests (fingerprints) of the objects are used to identify two objects as being the same or different.

Synchronization is not usually needed since objects that are created on the HA slot are automatically replicated to all members that are available and have network replication allowed. Synchronization is required when one member was unavailable for a time when keys or other objects were created in the HA group, or when a key or object was added directly to a member of an HA group instead of to the group itself (i.e. if a key is generated on slot 1, which is partition 1, instead of slot 3 which is the HA group.)

Use the command with the "-enable" option to enable synchronization.

Use the command with neither "-enable" nor "-disable" to trigger synchronization (if it is enabled).

Use the command with the "-disable" option to disable synchronization.

### Options

**-enable** [optional] Enables the synchronization.

**-disable** [optional] Disables the synchronization.

**-group <groupNum>** [mandatory] The group number of the HA group's to synchronize. Group numbers can be found using the *vtl haAdmin -listGroups* command.

**-password <password>** [mandatory] The text password for the partitions/group members. It is prompted if not supplied at the command line. Not needed for the -enable and -disable options.

### Example

```
vtl haAdmin synchronize -group 165010001
Please enter the password for the member partitions:
> *****
No synchronization performed/needed.vtl haAdmin -synchronize -group 165010001
Synchronization completed.
```

## vtl listServers

---

### Name

vtl listServers

### Syntax

vtl listServers

### Description

Displays a list of the SafeNet Servers trusted by this client.

### Options

None

### Example

```
vtl listServers
Server: lunas1
Server: test
```

## vtl listSlots

---

### Name

vtl listSlots

### Syntax

vtl listServers

### Description

Displays a list of all slots found.

### Options

None

### Example

```
:>vtl listSlots
Number of slots: 3
The following slots were found:
```

Slot#	Description	Label	Serial#	Status
0	Net Token Slot	kbPSO	1311583664227	Present
1	User Token Slot	mypciepsopar	349297122736	Present
2	Admin Token Slot	mypcie6	150022	Present
4	Luna UHD Slot	myG5pw	7001312	Present
5	Luna UHD Slot	-	-	Not present
6	Net Admin Token Slot	myRBSG5Bk	7000329	Present

```
:>
```

---

**Note:** In the example list above,

- slot 0 represents a network-linked application partition on a SafeNet Luna Network HSM
- slot 1 is the application partition on a SafeNet Luna PCIe HSM with firmware 6.22.0 or newer
- slot 2 is the HSM administrative partition of the same SafeNet Luna PCIe HSM
- slot 4 is a SafeNet Luna USB HSM - the label suggests that it is password authenticated, but we can't tell from the information in this listing
- slot 5 is a placeholder slot for an HSM that could be attached to a USB port
- slot 6 is the HSM administrative partition of a SafeNet Luna Backup HSM that is connected to this client via Remote Backup Service



You won't necessarily see all, or even most of those in your situation, with your equipment; the list in the example merely shows how different types are presented.

---

---

## vtl logging configure

---

### Name

vtl logging configure - set path to store log files

### Syntax

vtl logging configure <logPath>

### Description

Sets the directory path where log files are to be stored.

See "[vtl logging show](#) " on the next page to display the current log path.

The client library writes log messages to SYSLOG on Linux/UNIX systems.

However, for Windows, the log messages are written to the file "LunaCryptokiLog.htm" at the location that you specify in <logPath>.

To demonstrate that the logging is working on a Windows platform, you could create an error situation as follows:

1. Enable the client side log on a Windows platform.
2. Create a client certificate.
3. Register the client with a SafeNet Luna Network HSM appliance.
4. Manually delete the client certificate file.
5. Run ckemo or another application against a partition on that SafeNet Luna Network HSM. NTL is broken for this client (due to the missing certificate), so any commands from your application should fail.
6. Check LunaCryptokiLog.htm and observe error messages written there.

### Options

None

### Example

```
C:\Program Files\SafeNet\LunaClient>vtl logging configure "C:\Program Files\SafeNet\LunaClient"  
Success setting log path to C:\Program Files\SafeNet\LunaClient
```

## vtl logging show

---

### Name

vtl logging show - Displays the directory path where log files are stored.

### Syntax

```
vtl logging show
```

### Description

Displays the directory path where log files are stored.

See "[vtl logging configure](#) " on the previous page to set the log path.

### Options

None

### Example

```
C:\Program Files\SafeNet\LunaClient>vtl logging show  
Client logging written to: C:\Program Files\SafeNet\LunaClient\LunaCryptokiLog.htm
```

---

## vtl replaceserver

---

### Name

vtl replaceServer

### Syntax

```
vtl replaceServer -o <old_server_hostname_or_IP> -n <new_server_hostname_or_IP> -c <certificate_file>
```

### Description

Replaces the specified old server in the client's list of trusted SafeNet Servers, with the specified new server .

### Options

**-o** <old\_server\_hostname\_or\_IP> [mandatory] The hostname (or IP address) of the server being replaced. Use the IP address if the server's certificate uses its ip address instead of its hostname.

**-n** <new\_server\_hostname\_or\_IP> [mandatory] The hostname (or IP address) of the server that is replacing the previous server. Use the IP address if the server's certificate uses its ip address instead of its hostname. If you are uncertain what format the server's certificate uses contact your SafeNet appliance administrator, or look for the "CN=" field when using the vtl examineCert command.

**-c** <certificate\_file> [mandatory] The name (including the path to its location on your computer) of the server's certificate file.

### Example

```
bash # ./vtl replaceServer -o yourluna -n myluna -c server.pem  
New server myluna successfully added to server list.  
Server yourluna successfully replaced with myluna.
```

## vtl supportInfo

---

### Name

vtl supportInfo

### Syntax

vtl supportInfo

### Description

Creates a client-side support information file (may be requested by Technical Support to help resolve an issue).

### Options

None.

### Example

```
C:\Program Files\SafeNet Luna Network HSM> vtl supportInfo
Creating client-side support information file now...
'vtl supportInfo' completed.
File "c_supportInfo.txt" created.
C:\Program Files\SafeNet Luna Network HSM>
```

---

## vtl verify

---

### Name

vtl verify

### Syntax

vtl verify

### Description

Verify the SafeNet Luna Network HSM slots/Partitions visible to this Client.

### Options

None.

### Example

```
bash-2.03# ./vtl verify
```

The following SafeNet Luna Network HSM Slots/Partitions were found:

Slot	Serial #	Label
====	=====	=====
1	65091001	MyPartition
2	65097001	YourPartition
3	65093001	HisPartition

```
bash-2.03#
```