# SafeNet USB HSM 6.3

Utilities Reference Guide

gemalto
security to be free

## Document Information

| | |
|---|---|
| **Product Version** | 6.3 |
| **Document Part Number** | 007-011302-015 |
| **Release Date** | 14 July 2017 |

## Revision History

| Revision | Date | Reason |
|---|---|---|
| A | 14 July 2017 | Initial release. |

## Trademarks, Copyrights, and Third-Party Software



### Acknowledgements

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

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

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

This product uses Brian Gladman's AES implementation.

Refer to the End User License Agreement for more information.

## Disclaimer

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

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

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

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

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

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

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

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

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

## Regulatory Compliance

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

### USA, FCC

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

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

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

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

- Reorient or relocate the receiving antenna
- Increase the separation between the equipment and receiver
- Connect the equipment into an outlet on a circuit different from that to which the receiver is connected
- Consult the dealer or an experienced radio/TV technician for help

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

### Canada

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

 

### Europe

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

 

# CONTENTS

 

 

 

 

# PREFACE
# About the Utilities Reference Guide

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



This preface also includes the following information about this document:



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. It is strongly recommended that you read the CRN to fully understand the capabilities, limitations, and known issues for this release. You can view or download the latest version of the CRN for this release at the following location:

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

# Gemalto Rebranding

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

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

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

# Audience

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

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

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

# Document Conventions

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

## Notes

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

**Note:** Take note. Contains important or helpful information.

## Cautions

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

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

## Warnings

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

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

## Command Syntax and Typeface Conventions

| Format | Convention |
|---|---|
| **bold** | The bold attribute is used to indicate the following:<br>• Command-line commands and options (Type dir /p.)<br>• Button names (Click Save As.)<br>• Check box and radio button names (Select the Print Duplex check box.)<br>• Dialog box titles (On the Protect Document dialog box, click Yes.)<br>• Field names (User Name: Enter the name of the user.)<br>• Menu names (On the File menu, click Save.) (Click Menu > Go To > Folders.)<br>• User input (In the Date box, type April 1.) |
| *italics* | In type, the italic attribute is used for emphasis or to indicate a related document. (See the *Installation Guide* for more information.) |
| <variable> | In command descriptions, angle brackets represent variables. You must substitute a value for command line arguments that are enclosed in angle brackets. |
| [**optional**]<br>[<optional>] | Represent optional **keywords** or <variables> in a command line description. Optionally enter the keyword or <variable> that is enclosed in square brackets, if it is necessary or desirable to complete the task. |
| {**a**\|**b**\|**c**}<br>{<a>\|<b>\|<c>} | Represent required alternate **keywords** or <variables> in a command line description. You must choose one command line argument enclosed within the braces. Choices are separated by vertical (OR) bars. |

  

| Format | Convention |
|---|---|
| [**a**\|**b**\|**c**]<br>[<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 | |
|---|---|---|
| **Phone**<br>(Subject to change. An up-to-date list is maintained on the Technical Support Customer Portal) | Global | +1 410-931-7520 |
| | Australia | 1800.020.183 |
| | India | 000.800.100.4290 |
| | Netherlands | 0800.022.2996 |
| | New Zealand | 0800.440.359 |
| | Portugal | 800.863.499 |
| | Singapore | 800.1302.029 |
| | Spain | 900.938.717 |
| | Sweden | 020.791.028 |
| | Switzerland | 0800.564.849 |
| | United Kingdom | 0800.056.3158 |
| | United States | (800) 545-6608 |
| **Web** | https://safenet.gemalto.com | |
| **Technical Support Customer Portal** | https://supportportal.gemalto.com<br>Existing customers with a Technical Support Customer Portal account can log in to manage incidents, get the latest software upgrades, and access the Knowledge Base. To create a new account, click the **Register** link at the top of the page. You will need your Customer Identifier number. | |

  

# 1
# 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

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 SafeNet 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 |
|---|---|
| **-handle**=<handle#> | 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. |
| **-inputfile** | This parameter defines the name of the file that contains the PKCS #10 certificate request. |
| **-startDate** | This parameter defines the validity start of the certificate, in the format YYYYMMDD. |
| **-endDate** | This parameter defines the validity end of the certificate, in the format YYYYMMDD. |
| **-serialNumber** | This parameter defines the serial number of the certificate, in big-endian hexadecimal form. |

## Optional Parameters

| Parameter | Description |
|---|---|
| **-keyusage** | 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. |
| **-md5WithRsa** | This is an optional parameter that defines the signature algorithm for the certificate to be pkcs-1-MD5withRSAEncryption. The default is to use sha1WithRsa. |
| **-sha1WithRsa** | This is an optional parameter that defines the signature algorithm for the certificate to be pkcs-1-SHA1withRSAEncryption. The default is to use sha1WithRsa. |
| **-label** | 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. |
| **-sha224withrsa** | 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 |
|---|---|
| **-sha256withrsa** | This is an optional parameter that defines the signature algorithm for the certificate to be pkcs-1-SHA256withRSAEncryption. The default is to use sha1WithRsa. |
| **-sha384withrsa** | This is an optional parameter that defines the signature algorithm for the certificate to be pkcs-1-SHA384withRSAEncryption. The default is to use sha1WithRsa. |
| **-sha512withrsa** | This is an optional parameter that defines the signature algorithm for the certificate to be pkcs-1-SHA512withRSAEncryption. The default is to use sha1WithRsa. |
| **-sha1withdsa** | This is an optional parameter that defines the signature algorithm for the certificate to be pkcs-1-SHA1withDSAEncryption. The default is to use sha1WithRsa. |
| **-sha1withecdsa** | This is an optional parameter that defines the signature algorithm for the certificate to be pkcs-1-SHA1withECDSAEncryption. The default is to use sha1WithRsa. |
| **- sha224withecdsa** | This is an optional parameter that defines the signature algorithm for the certificate to be pkcs-1-SHA224withECDSAEncryption. The default is to use sha1WithRsa. |
| **-sha256withecdsa** | This is an optional parameter that defines the signature algorithm for the certificate to be pkcs-1-SHA256withECDSAEncryption. The default is to use sha1WithRsa. |
| **-sha384withecdsa** | This is an optional parameter that defines the signature algorithm for the certificate to be pkcs-1-SHA384withECDSAEncryption. The default is to use sha1WithRsa. |
| **- sha512withecdsa** | This is an optional parameter that defines the signature algorithm for the certificate to be pkcs-1-SHA512withECDSAEncryption. The default is to use sha1WithRsa. |
| **-id** | 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. |
| **-certificatepolicy** | This optional parameter defines the certificate policy to be used. |
| **-keyids** | 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 |
|---|---|
| **-handle**=<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 |
|---|---|
| **-force** | 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 |
|---|---|
| **-handle**=<handle#> | 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). |
| **-outputfile** | This mandatory parameter defines the name of the file that receives the certificate. |

## Optional Parameters

| Parameter | Description |
|---|---|
| **-binary** | 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 |
|---|---|
| **-modulusBits** | 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.<br>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. |
| **-publicExponent** | 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 |
|---|---|
| **-binary** | This is an optional parameter that defines the exported certificate format to be raw binary instead of the default PEM (base64) encoding. |
| -inputFile | This optional parameter defines the name of a file from which to obtain additional parameter settings, one per line, of the form name=value. |
| **-keyType** | 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. |
| **-label** | 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. |
| **-labelPublic** | This optional parameter defines a label to apply to the public key from the newly generated key pair. |

| Parameter | Description |
|---|---|
| **-labelPrivate** | This optional parameter defines a label to apply to the private key from the newly generated key pair. |
| **-modifiable** | 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). |
| **-id** | This optional parameter defines the Id field for the newly generated keys. It must be set to a big-endian hexadecimal integer value. |
| **-startDate** | This optional parameter defines the startDate field for the newly generated keys. The format for the value is YYYYMMDD. |
| **-endDate** | This optional parameter defines the endDate field for the newly generated keys. The format for the value is YYYYMMDD. |
| **-subject** | 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. |
| **-encrypt** | 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. |
| **-decrypt** | 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. |
| **-sign** | 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. |
| **-verify** | 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. |
| **-wrap** | 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 |
|---|---|
| **-unwrap** | 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. |
| **-extractable** | 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. |
| **-curvetype** | 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). |
| **-prime** | This optional parameter defines a prime length for DSA key generation. |
| **-subprime** | This optional parameter defines a subprime bits length for DSA key generation. |
| **-base** | 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 0xfcec6182eb2<...too long to reproduce...>fe00d0204c3
   -subprime 0xd3807350xd3807<...long string...>cedc61
    -base 0x42e37<...too long to reproduce...>22c3b1205e
```

## 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: 0xfcec6182eb2<...too long to reproduce...>fe00d0204c3
```

  

```
Enter a 160 bit subprime value: 0xd3807<...long string...>cedc61
Enter a 1024-bit base value: 0x42e37<...too long to reproduce...>22c3b1205e
```

# 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 |
|---|---|
| **-handle**=<handle#> | The handle to the object. The parameter "-handle" is followed by an equal sign "=", followed by the handle of the object (no spaces).<br>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 |
|---|---|
| **-attributes** | 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. |
| **-outputFile** | 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 |
|---|---|
| **-handle**=<handle#> | 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). |
| **-outputfile** | This mandatory parameter defines the name of the file that receives the PKC. |
| **-pkctype** | This mandatory parameter defines the PKC type (1 - TC-TrustCenter, 2 - Chrysalis-ITS). |
| **-verify** | 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 |
|---|---|
| **-inputFile** | This parameter defines the name of the file containing the certificate to import. |

## Optional Parameters

| Parameter | Description |
|---|---|
| **-label** | 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 |
|---|---|
| **-in (Filename)** | This parameter defines the full path to the file containing the key to import. |
| **-keyarg (DSA\|RSA\|ECDSA)** | Specifies the key's algorithm. |

## Optional Parameters

| Parameter | Description |
|---|---|
| **-PKCS8** | Indicates that the key to import is formatted according to the PKCS#8 standard. |
| **-PKCS12** | 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.<br>It is assumed that you properly export a PKCS #12 key from Windows keystore (or other source, as appropriate). |
| **-wrapkey (handle)** | 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. |
| **-setkeyattr** | 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 |
|---|---|
| **-display** | 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. |
| **-class** | This option defines the class of object to list. It can be set to any of data, certificate, public, private and secret. |
| **-keyType** | 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. |
| **-certificateType** | This option specifies the type of certificate to list. It can only be set to x.509 if used. |
| **-label** | This option specifies the label that objects must match in order to be listed. |
| **-application** | This option specifies the application attribute that objects must match in order to be listed. |
| **-value** | This option specifies the value that objects must match in order to be listed. |
| **-issuer** | This option specifies the issuer that objects must match in order to be listed. |
| **-serialNumber** | This option specifies the serial number that objects must match in order to be listed. |
| **-subject** | This option specifies the subject that objects must match in order to be listed. |
| **-id** | This option specifies the id that objects must match in order to be listed. |
| **-token** | 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 |
|---|---|
| **-private** | Set to True or False (or 1 or 0). |
| **-sensitive** | Set to True or False (or 1 or 0). |
| **-alwaysSensitive** | Set to True or False (or 1 or 0). |
| **-extractable** | Set to True or False (or 1 or 0). |
| **-neverExtractable** | Set to True or False (or 1 or 0). |
| **-local** | Set to True or False (or 1 or 0). |
| **-encrypt** | Set to True or False (or 1 or 0). |
| **-decrypt** | Set to True or False (or 1 or 0). |
| **-sign** | Set to True or False (or 1 or 0). |
| **-verify** | Set to True or False (or 1 or 0). |
| **-wrap** | Set to True or False (or 1 or 0). |
| **-unwrap** | Set to True or False (or 1 or 0). |
| **-derive** | Set to True or False (or 1 or 0). |
| **startDate** | This option specifies the start date that objects must match in order to be listed. |
| **endDate** | This option specifies the end date that objects must match in order to be listed. |
| **modulusBits** | This option specifies the modulus size that RSA keys must match in order to be listed. |
| **publicExponent** | 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. |
| **- modifiable** | 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 |
|---|---|
| **-publichandle=** <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. |
| **-privatehandle** =<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. |
| **-outputFile** | This mandatory parameter defines the file that receives the certificate request. |

## Optional Parameters

| Parameter | Description |
|---|---|
| **-binary** | This is an optional parameter that defines the certificate request format to be raw binary instead of the default PEM (base64) encoding. |
| **-sha1WithRsa** | 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. |
| **- sha224withrsa** | 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. |
| **- sha256withrsa** | 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. |
| **- sha384withrsa** | 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. |
| **- sha512withrsa** | 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. |
| **- sha1withdsa** | 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. |
| **- sha1withecdsa** | 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. |
| **- sha224withecdsa** | 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. |
| **- sha256withecdsa** | 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. |
| **- sha384withecdsa** | 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. |
| **- sha512withecdsa** | 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. |
| **-C** | 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. |
| **-S** | 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. |
| **-L** | 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. |
| **-O** | 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. |
| **-OU** | 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 |
| **-CN** | 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 |
|---|---|
| **-publichandle=** <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. |
| **-privatehandle** =<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. |
| **-startDate** | This parameter defines the validity start of the certificate, in the format YYYYMMDD. |
| **-endDate** | This parameter defines the validity end of the certificate, in the format YYYYMMDD. |
| **-serialNumber** | This parameter defines the serial number of the certificate, in big-endian hexadecimal form. |

## Optional Parameters

| Parameter | Description |
|---|---|
| **-keyusage** | 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. |
| **-label** | 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. |
| **-id** | 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. |
| **-md5WithRsa** | This is an optional parameter that defines the signature algorithm for the certificate request to be pkcs-1-MD5withRSAEncryption.<br>**Default:** SHA256withRSAEncryption |
| **-sha1WithRsa** | This is an optional parameter that defines the signature algorithm for the certificate request to be pkcs-1-SHA1withRSAEncryption.<br>**Default:** SHA256withRSAEncryption |
| **- sha224withrsa** | This is an optional parameter that defines the signature algorithm for the certificate request to be pkcs-1-sha224withRSAEncryption.<br>**Default:** SHA256withRSAEncryption |
| **- sha256withrsa** | This is an optional parameter that defines the signature algorithm for the certificate request to be pkcs-1-sha256withRSAEncryption.<br>**Default:** SHA256withRSAEncryption |
| **- sha384withrsa** | This is an optional parameter that defines the signature algorithm for the certificate request to be pkcs-1-sha384withRSAEncryption.<br>**Default:** SHA256withRSAEncryption |
| **- sha512withrsa** | This is an optional parameter that defines the signature algorithm for the certificate request to be pkcs-1-sha512withRSAEncryption.<br>**Default:** SHA256withRSAEncryption |
| **- sha1withdsa** | This is an optional parameter that defines the signature algorithm for the certificate request to be pkcs-1-sha1withDSAEncryption.<br>**Default:** SHA256withRSAEncryption |
| **- sha1withecdsa** | This is an optional parameter that defines the signature algorithm for the certificate request to be pkcs-1-sha1withECDSAEncryption.<br>**Default:** SHA256withRSAEncryption |
| **- sha224withecdsa** | This is an optional parameter that defines the signature algorithm for the certificate request to be pkcs-1-sha224withECDSAEncryption.<br>**Default:** SHA256withRSAEncryption |

| Parameter | Description |
|---|---|
| **- sha256withecdsa** | This is an optional parameter that defines the signature algorithm for the certificate request to be pkcs-1-sha256withECDSAEncryption.<br>**Default:** SHA256withRSAEncryption |
| **- sha384withecdsa** | This is an optional parameter that defines the signature algorithm for the certificate request to be pkcs-1-sha384withECDSAEncryption.<br>**Default:** SHA256withRSAEncryption |
| **- sha512withecdsa** | This is an optional parameter that defines the signature algorithm for the certificate request to be pkcs-1-sha512withECDSAEncryption.<br>**Default:** SHA256withRSAEncryption |
| **-C** | 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. |
| **-S** | 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. |
| **-L** | 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. |
| **-O** | 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. |
| **-OU** | 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. |
| **-CN** | 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 Cer-
tificate"
-startDate=20120101 -endDate=20151231 -serialNum=0133337f
```

  

# cmu setattribute

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 setAttribute** <parameters>

## Required Parameters

| Parameter | Description |
|---|---|
| **-handle**=<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 |
|---|---|
| **-inputFile** | 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. |
| **-label** | This optional parameter defines a new value for the label of an object on the HSM. |
| **-application** | This optional parameter defines a new value for the application attribute of a data object on the HSM. |
| **-value** | 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. |
| **-issuer** | 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. |
| **-serialNumber** | 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. |
| **-subject** | 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 |
|---|---|
| **-id** | 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. |
| **-extractable** | 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). |
| **-encrypt** | 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. |
| **-decrypt** | 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. |
| **-sign** | 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. |
| **-verify** | 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. |
| **-wrap** | 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. |
| **-unwrap** | 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. |
| **-derive** | 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. |
| **-startDate** | This optional parameter defines a new startDate field for a key on the HSM. The format for the value is YYYYMMDD. |
| **-endDate** | This optional parameter defines a new endDate field for a key on the HSM. The format for the value is YYYYMMDD. |
| **-sensitive** | 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 verifyhsm

Verify a Public Key Confirmation from a Luna HSM.

This command allows you to verify that the client is connected to a genuine Luna HSM, by creating and verifying a confirmation on a temporary key created in the HSM. It also includes a proof of possession that asks the HSM to sign a user-entered string as proof the associated private key is present within the target HSM.

## Syntax

**cmu verifyhsm -challenge "**<string>**"** [**-rootcert** <filename>]

## Required Parameters

| Parameter | Description |
|---|---|
| **-challenge** | This parameter defines a user-entered string for the HSM to sign. |
| **-rootcert** | This parameter defines the name of the .pem file that contains the root certificate. |

## Optional Parameters

None.

## Example

```
./cmu verifyhsm -challenge "1234567890" -rootcert rootcert.pem
Select token
 [0] Token Label: sa200-1
 [1] Token Label: sa200-2
 Enter choice: 0
Please enter password for token in slot 0 : *******
Reading rootcert from file "rootcert.pem"... ok.
Generating temporary RSA keypair in HSM... ok.
Extracting PKC bundle from HSM... ok.
Verifying PKC certificate... ok.
Verifying DAC certificate... ok.
Verifying HOC certificate... ok.
Verifying MIC certificate... ok.
Verifying MIC against rootcert... ok.
Signing and verifying challenge... ok.
Verifying HSM serial number... ok.
Overall status: Success.
```

# cmu verifypkc

Verify a Public Key Confirmation from the HSM.

## Syntax

**cmu verifypkc** <parameters>

## Required Parameters

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

## Optional Parameters

None.

## Example

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

# 2
# CKdemo

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 HSM client and can be used with any SafeNet HSM.

This chapter contains the following sections:



## Accessing the ckdemo Utility

The ckdemo utility is included with the SafeNet 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 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 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" below .

# 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 SafeNet PED for Trusted Path appliances. If the SafeNet PED is not connected and ready when a command is issued, the command eventually times out. If the SafeNet 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 SafeNet 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 the next page
- "The CA Menu Functions" on page 45
- "The CLUSTER EXECUTION Menu Functions" on page 46
- "The HIGH AVAILABILITY RECOVERY Menu Functions" on page 47
- "The KEY Menu Functions" on page 47
- "The OBJECT MANAGEMENT Menu Functions" on page 48
- "The OFFBOARD KEY STORAGE Menu Functions" on page 50
- "The OTHERS Menu Functions" on page 50
- "The PED INFO menu functions" on page 52


  

# The AUDIT/LOG Menu Functions

The AUDIT/LOG menu provides the following functions:

## (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

## The CA Menu Functions

The CA menu provides the following functions:

### (70) Set Domain

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

### (79) Modify MofN

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

### (86) Dup. MofN Keys

(Not for SafeNet 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 HSM, to allow migration of token objects from legacy HSMs.

## The CLUSTER EXECUTION Menu Functions

The CLUSTER EXECUTION menu provides the following functions:

### (111) Get Cluster State

## The HIGH AVAILABILITY RECOVERY Menu Functions

The HIGH AVAILABILITY RECOVERY menu provides the following functions:

### (50) HA Init

(Not for SafeNet 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 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 generatanthen 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 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 OFFBOARD KEY STORAGE Menu Functions

The OFFBOARD KEY STORAGE menu provides the following functions:

## (101) Extract Masked Object

Extracts a key off the SafeNet 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 Network HSM. You are prompted for the name of the file, which must have been extracted from a SafeNet Network HSM using the same masking key (i.e., the same SafeNet 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)

## (105) SIMExtract

## (106) SIMInsert

## (107) SimMultiSign

## (118) Extract Object

## (119) Insert Object

# 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

# 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 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 SCRIPT EXECUTION Menu Functions

The SCRIPT EXECUTION menu provides the following functions:

### (108) Execute Script

### (109) Execute Asynchronous Script

### (110) Execute Single Part Script

## 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 SRK Menu Functions

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 SafeNet 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 SafeNet PED and

the purple PED Key with the correct SRV split on it - that purple key then becomes invalid).

# 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. See

## ( 2) Close Session

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 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 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 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 (option #8 - Mechanism List presents a list of them) 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 Network HSM) Copy a clonable secret-splitting vector from one token to another.

# 3
# CKlog

SafeNet Software Development Kit can record all interactions between an application and our PKCS#11-compliant library, allowing a developer to debug an application by viewing what the library receives.

The tool is the Cryptoki Logging Facility or cklog. In function, cklog is a library that displaces our PKCS#11 library. When it receives a call it does not service the request but, instead, logs the call to a file and passes the request to the originally intended library.

For cklog to function properly, perform these two steps:

1. Direct the application to use the cklog library instead of the regular Chrystoki library.
2. Instruct the cklog library where to access the regular library.

Achieve the first step by modifying the configuration files to instruct CkBridge to load the Cklog library. This redirection is described in the next sub-section. The second step involves different blocks in the configuration file.

Here are descriptions of entries that might be applicable:

- LibNT - references to a Cryptoki library for Windows 2008 and Windows 2012.
- LibUNIX - references to a Cryptoki library for UNIX (meaning Solaris, Linus and AIX).
- LibHPUX - references to a Cryptoki library specific to HP-UX.
- Enabled - 0 or 1. Allows turning the logging facility off or on.
- File - references the file to which the requests should be logged.
- Error - references a file where the logging facility can record fatal errors.
- NewFormat - 0 or 1 disables/enables a more compact output format, which is the format preferred by SafeNet Customer Support.

## Windows Example

The following example shows a typical initialization file under Windows where cklog is in use:

```
[Chrystoki2]
LibNT=c:\Program Files\SafeNet\LunaClient\cklog201.dll
[CkLog2]
LibNT=c:\Program Files\SafeNet\LunaClient\cryptoki.dll
Enabled=1
File=c:\Program Files\SafeNet\LunaClient\cklog2.txt
Error=c:\Program Files\SafeNet\LunaClient\error2.txt
NewFormat=1
LoggingMask=ALL_FUNC
```

## UNIX Example

The following example shows a typical configuration file under UNIX where cklog is in use:

```
Chrystoki2 = {
```

  

```
LibUNIX=/usr/lib/libcklog2.so;
}
CkLog2 = {
LibUNIX=/usr/lib/libCryptoki2.so;
Enabled=1;
File=/tmp/cklog.txt;
Error=/tmp/error.txt;
NewFormat=1;
LoggingMask=ALL_FUNC;
}
```

## Selective Logging

When logging is turned on, all functions are logged, by default. If you wish to restrict logging to particular functions of interest only, you can edit the "LoggingMask=" parameter in the crystoki.ini [Windows] or Chrystoki.conf [UNIX] file to include flags for the desired logging.

### LoggingMask= Flags

Here is the list of possible flags for cklog:

| Flag | Description |
|---|---|
| GEN_FUNCS | General Functions |
| SLOT_TOKEN_FUNC | Slot/Token related functions |
| SESSION_FUNC | Session related functions |
| OBJ_MNGMNT_FUNC | Object Management functions |
| ENC_DEC_FUNC | Encrypt/Decrypt related functions |
| DIGEST_FUNC | Digest Related functions |
| SIGN_VERIFY_FUNC | Signing/Verifying related functions |
| KEY_MNGMNT_FUNC | Key Management related functions |
| MISC_FUNC | Misc functions |
| CHRYSALIS_FUNC | SafeNet Extensions functions |
| ALL_FUNC | All functions logged; |

You can mix and match any or all of the flags, using the "|" operator. For example, the following:

LoggingMask=GEN_FUNC | SLOT_TOKEN_FUNC | ENC_DEC_FUNC | SIGN_VERIFY_FUNC;

would be valid.

**Note:** You can use the flags in any order. Using the ALL_FUNC flag overrides any other flag. If you have the "LoggingMask=" parameter, with NO flags set, then nothing is logged. If logging capability is enabled (cklog), but there is no "LoggingMask=" line, then default behavior prevails and everything is logged.

  

# 4
# Lunadiag

Lunadiag is a diagnostic tool for SafeNet card products. In general, you may never need to use it, other than to confirm a successful SafeNet installation. If you experience problems with a SafeNet product and need to contact Customer Support, you may be asked to perform additional tests with Lunadiag, as part of the troubleshooting process. In that circumstance, the support representative will instruct you. Several menu items are self-explanatory. The more obscure items are of interest only to Technical Support in very specific circumstances.

However, if you are an application developer, you may wish to use Lunadiag during your software-development. You have the option to run Lunadiag from the command line of a console window. From the command line, the syntax for Lunadiag is:

lunadiag [-s=num] [-o=num] [-c=num] <[options]>

Where

`-s=num` Number of slots to test at once.

(Range: 1.. N; default: 1 where N is the number of slots available to the client)

`-o=num` Offset into slots to begin testing

(Range: 0.. N-1; default: 0)

`-c=num` Command to run (Range: 1..16)

for example, `lunadiag -s=1 -o=1 -c=11`

The spaces are required. The following additional options can be executed, and exit immediately without user prompt.

`-CHRYSTOKI` Perform the Chrystoki Library configuration test.

`-DUALPORT` Dump dualport.

`-FIPS` Test for FIPS setting for one token.

Exit code 1 implies FIPS enabled.

## Using Lunadiag

Run lunadiag with no arguments, to get a list of slots that it can see.

```
C:\Program Files\SafeNet\LunaClient>lunadiag
lunadiag  version 8.0  Date: Feb 13 2015 Time: 14:21:44
Detecting Luna devices ...
Detection complete.

Slots available:
        Slot #0 - Present     - LunaNet Slot
        Slot #1 - Present     - LunaNet Slot
        Slot #2 - Present     - LunaNet Slot
        Slot #3 - Present     - Viper PCI Card
        Slot #4 - Not present - Luna UHD Tunnel Slot
        Slot #5 - Present     - Luna UHD Slot
```

  

```
        Slot #6 - Not present - Luna UHD Slot
        Slot #7 - Not present - Luna UHD Slot
Enter slot to test:
```

In the slot list, above, slots 0, 1, and 2 are listed as "LunaNet Slot", and correspond to SafeNet Network HSM application partitions that are registered with this client/host.

Slot 3, "Viper PCI Card", is a locally contained SafeNet PCIe HSM physical slot. While LunaCM shows a separate HSM administrative slot and application partition slot (if HSM firmware is version 6.22.0 or newer), lunadiag shows a single physical slot.

Similarly, Slot 5, "Present - Luna UHD Slot", is a SafeNet USB HSM physical slot.

Slot 4 "Not present - Luna UHD Tunnel Slot", is reserved for a USB HSM Device (UHD) like a SafeNet Backup HSM that could be directly connected to the SafeNet PCIe HSM card.

The slots listed as "Not Present - Luna UHD Slot" are placeholders for other possible devices that could be USB-connected, but currently are not.

Lunadiag displays a menu of commands, once you have selected a slot to work on.

```
C:\Program Files\SafeNet\LunaClient>lunadiag
lunadiag  version 8.0  Date: Feb 13 2015 Time: 14:21:44
Detecting Luna devices ...
Detection complete.

Slots available:
        Slot #0 - Present     - LunaNet Slot
        Slot #1 - Present     - LunaNet Slot
        Slot #2 - Present     - LunaNet Slot
        Slot #3 - Present     - Viper PCI Card
        Slot #4 - Not present - Luna UHD Tunnel Slot
        Slot #5 - Present     - Luna UHD Slot
        Slot #6 - Not present - Luna UHD Slot
        Slot #7 - Not present - Luna UHD Slot
Enter slot to test:
```

In order to see the lunadiag menu of commands, first select a slot on which to act:

```
Enter slot to test: 0

----------------------------------------
lunadiag  version 8.0  Date: Feb 13 2015 Time: 14:21:44

                Main Menu

           1   Select slot to test
           2   Driver Test
           3   Communication Test
           4   Read Firmware Level
           5   Read Protocol Level
           6   Read Capabilities
           7   Read Token Policies
           8   Read TSV
           9   Read Dualport
          10   Read Dualport Command
          11   Token Info Test
          12   Mechanism Info Test
```

  

```
          16   Read Debug/Trace Information

           0   Exit
--------------------------------------
```

Command 9 is a complete dual-port dump of a SafeNet PCIe HSM, which includes any debug/trace information at the end. This command does not work for SafeNet USB HSM because that HSM is not built around dual-port architecture.

Command 10 attempts to present information from the current command.

Command 16 provides just the debug/trace information for either a SafeNet USB HSM or a SafeNet PCIe HSM. For SafeNet PCIe HSM, this is a much more compact output than is available from command 9. For SafeNet USB HSM, this is all the information available, since there is no dual-port to expose.

The "missing" commands, 13, 14, and 15 appear only in special circumstances. The example that might have some general relevance is where Microsoft IIS is in use, and settings "AppIdMajor=1" and "AppIdMinor=42" are present in the Crystoki.ini file; this causes menu item 15 to appear. Generally, if a menu number does not appear, you do not need it. If in doubt, contact Technical Support.

## Verify Successful Installation

If you can run tests

```
 2  Driver Test
 3  Communication Test
```

and

```
 4  Read Firmware Level
```

successfully (if they do not return error messages) then the installation was successful.

If there is a problem, check the connections to your HSM.

If there is still a problem, remove and re-install the SafeNet HSM Client software.

If problems persist, contact SafeNet/Gemalto Technical Support.

# 5
# Multitoken

Multitoken is a simple demonstration tool that allows you to perform basic cryptographic functions on a SafeNet HSM. The 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.

**CAUTION:** To achieve maximum performance with SafeNet Network HSM 5.x and 6.x, client applications must spawn 30+ threads. The 10 threads indicated for legacy SafeNet Network HSM 4.x is not sufficient to stress the current product.

## 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>] [**-noverifyr**] [**-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 |
|---|---|---|
| **-alarm** | **-al** | Sound periodic alarm (every <secs> seconds) if error occurs. |
| **-blob** | **-b** | Number of data blobs to be signed during each multisign operation. |
| **-curv** | **-crv** | ID number of ECC curve. If user-defined (99), then must specify -parmfile. |
| **-force** | **-f** | Avoid prompts for responses. |
| **-ped** | **-ped** | 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. |
| **-help** | **-h** | Display help information only. |
| **-key** | **-k** | Size of key: asymmetric in bits (default = 1024 for RSA, 2048 for DSA). symmetric in bytes (i.e. 16, 24, 32 for AES/ARIA). |
| **-keychoice** | **-kc** | Select key type to derive/generate - specify choice list index. |
| **-kdfchoice** | **-kdf** | Select key derivation function - specify choice list index. |
| **-kdfscnt** | **-kds** | Select key derivation session counter type - specify choice list index. |
| **-usage** | **-u** | Number of times a key is allowed to be used. |

  

| Parameter | Shortcut | Description |
|---|---|---|
| **-logfile** | **-l** | File for results logging. |
| **-mode** | **-m** | Operating mode. See mode values available below. |
| **-multipartsig** | **-msig** | Use multipart signatures. |
| **-nodec** | **-nod** | Decryption operation will not be performed. Only symmetric and asymmetric encryption will be performed and measured. |
| **-nodestroy** | **-n** | Leaves created objects on the HSM after test completes. |
| **-noenc** | **-noe** | Perform only one encryption operation. Only symmetric and asymmetric decryption will be performed and measured. |
| **-nosign** | **-nos** | Perform only one sign operation. Only verify will be performed and measured. |
| **-noverify** | **-nov** | Verify operation will not be performed. Only sign will be performed and measured. |
| **-noverifyr** | **-nvr** | Do not verify decryption results. |
| **-packet** | **-p** | Size of packet used in operation. |
| **-parmfile** | **-prm** | File for EC curve parameters or OAEP source data (0 = none for OAEP). |
| **-password** | **-pwd** | Specify password to use for token. |
| **-prftype** | **-prf** | Specify the type of PRF to use for PRF based key derivation. |
| **-sharefile** | **-shf** | Shared data file used for operation. |
| **-slots** | **-s** | List of of slots to use (slot numbers separated by commas). |
| **-subprime** | **-sub** | Size of the sub-prime in bits. |
| **-symm** | **-sym** | Select symmetric key mechanism for symderive/pbegen or key choice for symgen (can also use -kc). |
| **-timed** | **-t** | Fixed amount of time to run (seconds). |
| **-nslots** | **-ns** | 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<br>This will create 5 threads on slot 1 and 10 threads on slot 2. |
| **-verbose** | **-v** | Show all thread performances. Default is only first and last threads. |
| **-template** | **-tp** | 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 |
|---|---|
| aescmac | AES CMAC sign |
| aesenc | AES ECB encrypt |
| aesenccbc | AES CBC encrypt |
| aesenccfb128 | AES CFB128 encrypt |
| aesencctr | AES CTR encrypt |
| aesencfb8 | AES CFB8 encrypt |
| aesencgcm | AES GCM encrypt |
| aesenckw | AES KW encrypt |
| aesenckwp | AES KWP encrypt |
| aesencofb | AES OFB encrypt |
| aesgmac | AES GMAC sign |
| aesmac | AES MAC sign |
| aeswrapkw | AES KW wrap |
| aeswrapkwp | AES KWP wrap |
| aesxts | AES XTS encrypt |
| ariacmac | ARIA CMAC sign |
| ariaenc | ARIA ECB encrypt |
| ariaenccbc | ARIA CBC encrypt |
| ariaenccfb8 | ARIA CFB8 encrypt |
| ariaenccfb128 | ARIA CFB128 encrypt |
| ariaencctr | ARIA CTR encrypt |
| ariaencofb | ARIA OFB encrypt |
| ariamac | ARIA MAC sign |
| des3enccfb8 | DES3 CFB8 encrypt |
| des3enccfb64 | DES3 CFB64 encrypt |
| des3encctr | DES3 CTR encrypt |
| des3encofb | DES3 OFB encrypt |

  

| Mode | Description |
|---|---|
| descmac | DES3 CMAC sign |
| desenc | DES3 ECB encrypt |
| desenccbc | DES3 CBC encrypt |
| desmac | DES3 MAC sign |
| desx919mac | DES3 X919 MAC sign |
| dhparamsgen | DH Domain Parameter Generation |
| dsakeygen | DSA Key Generation |
| dsaparamsgen | DSA Domain Parameter Generation |
| dsasigver | DSA bare sign |
| dukptderive | DUKPT key derivation |
| ecdhcderive | ECDH Cofactor derive key |
| ecdhderive | ECDH derive key |
| ecdhderivewrapnew | ECDH derive and wrap new |
| ecdhderivewrapold | ECDH derive and wrap old |
| ecdsagbcssha256sigver | SHA256 ECDSA-GBCS sign |
| ecdsakeygen | ECDSA Key Generation |
| ecdsakeygenwextrabits | ECDSA Key Gen with Extra Bits |
| ecdsasha1sigver | SHA1 ECDSA sign |
| ecdsasha224sigver | SHA224 ECDSA sign |
| ecdsasha256sigver | SHA256 ECDSA sign |
| ecdsasha384sigver | SHA384 ECDSA sign |
| ecdsasha512sigver | SHA512 ECDSA sign |
| ecdsasigver | ECDSA sign |
| ecedwardskeygen | EC Edwards Key Generation |
| 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 |

| Mode | Description |
|---|---|
| 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 |
| eciesdes3hmacsha224 | ECIES DES3 enc/dec with HMAC SHA224 |
| eciesdes3hmacsha224shared | ECIES DES3 enc/dec with HMAC SHA224 and shared data |
| eciesshimaes128hmacsha256 | ECIES AES-128 with HMAC SHA256 decrypt |
| eciesshimaes128hmacsha256shared | ECIES AES-128 with HMAC SHA256 and shared data decrypt |
| eciesshimaes192hmacsha384 | ECIES AES-192 with HMAC SHA384 decrypt |
| eciesshimaes192hmacsha384shared | ECIES AES-192 with HMAC SHA384 and shared data decrypt |
| eciesshimaes256hmacsha512 | ECIES AES-256 with HMAC SHA512 decrypt |
| eciesshimaes256hmacsha512shared | ECIES AES-256 with HMAC SHA512 and shared data decrypt |
| eciesshimdes3hmacsha224 | ECIES DES3 with HMAC SHA224 decrypt |
| eciesshimdes3hmacsha224shared | ECIES DES3 with HMAC SHA224 and shared data decrypt |
| eciesshimxorhmacsha1 | ECIES XOR with HMAC SHA1 decrypt |
| eciesshimxorhmacsha1shared | ECIES XOR with HMAC SHA1 and shared data decrypt |
| eciesxorhmacsha1 | ECIES XOR enc/dec with HMAC SHA1 |
| eciesxorhmacsha1shared | ECIES XOR enc/dec with HMAC SHA1 and shared data |
| ecmontkeygen | EC Montgomery Key Generation |
| eddsanaclsha1sigver | SHA1 EDDSA NaCl sign |
| eddsanaclsha224sigver | SHA224 EDDSA NaCl sign |
| eddsanaclsha256sigver | SHA256 EDDSA NaCl sign |
| eddsanaclsha384sigver | SHA384 EDDSA NaCl sign |
| eddsanaclsha512sigver | SHA512 EDDSA NaCl sign |
| eddsanaclsigver | EDDSA NaCl sign |
| eddsasha1sigver | SHA1 EDDSA sign |
| eddsasha224sigver | SHA224 EDDSA sign |
| eddsasha256sigver | SHA256 EDDSA sign |

| Mode | Description |
|---|---|
| eddsasha384sigver | SHA384 EDDSA sign |
| eddsasha512sigver | SHA512 EDDSA sign |
| eddsasigver | EDDSA sign |
| extractinsert | Extract Insert masked objects |
| findobject | Find objects |
| kcdsakeygen | KCDSA Key Generation |
| kcdsasha1sigver | SHA51 KCDSA sign |
| kcdsasha1sigvernopad | SHA1 KCDSA NO-PAD sign |
| kcdsasha224sigver | SHA224 KCDSA sign |
| kcdsasha224sigvernopad | SHA224 KCDSA NO-PAD sign |
| kcdsasha256sigver | SHA256 KCDSA sign |
| kcdsasha256sigvernopad | SHA256 KCDSA NO-PAD sign |
| kcdsasha384sigver | SHA384 KCDSA sign |
| kcdsasha384sigvernopad | SHA384 KCDSA NO-PAD sign |
| kcdsasha512sigver | SHA512 KCDSA sign |
| kcdsasha512sigvernopad | SHA512 KCDSA NO-PAD sign |
| kcdsasigver | HAS160 KCDSA 1024-bit sign |
| kcdsasigvernopad | HAS160 KCDSA NO-PAD 1024-bit sign |
| md5 | MD5 Hashing |
| multisignvalue | Multisign w/ masked key |
| ntlsEcho | Test NTLS/SSL Throughput |
| objectcreation | Create/delete object |
| openclosesession | Open/close session |
| pbegen | PBE key generation |
| randgen | Random number generation |
| rc4enc | RC4 encrypt |
| rsa1863auxprimekeygen | RSA FIPS 186-3 using Auxiliary Primes key generation |

  

| Mode | Description |
|---|---|
| rsa1863primekeygen | RSA FIPS 186-3 using Primes key generation |
| rsaenc | RSA encrypt |
| rsakeygen | RSA key generation |
| rsaoaepenc | RSA OAEP encrypt |
| rsasigver | RSA sign |
| rsax931keygen | RSA X9.31 key generation |
| rsax931sigver | X9.31 RSA sign |
| seedcmac | SEED CMAC sign |
| seedenc | SEED ECB encrypt |
| seedenccbc | SEED CBC encrypt |
| seedencctr | SEED CTR encrypt |
| seedmac | SEED MAC sign |
| sha1 | SHA-1 Hashing |
| sha1dsasigver | SHA1 DSA sign |
| sha1hmac | SHA1 HMAC sign |
| sha1rsapsssigver | SHA1 RSA PSS sign |
| sha1rsasigver | SHA1 with RSA sign |
| sha1rsax931sigver | SHA1 X9.31 RSA sign |
| sha224 | SHA-224 Hashing |
| sha224dsasigver | SHA224 DSA sign |
| sha224hmac | SHA224 HMAC sign |
| sha224rsaoaepenc | SHA224 RSA OAEP encrypt |
| sha224rsapsssigver | SHA224 RSA PSS sign |
| sha224rsasigver | SHA224 with RSA sign |
| sha224rsax931sigver | SHA224 X9.31 RSA sign |
| sha256 | SHA-256 Hashing |
| sha256dsasigver | SHA256 DSA sign |

  

| Mode | Description |
|---|---|
| sha256hmac | SHA256 HMAC sign |
| sha256rsaoaepenc | SHA256 RSA OAEP encrypt |
| sha256rsapsssigver | SHA256 RSA PSS sign |
| sha256rsasigver | SHA256 with RSA sign |
| sha256rsax931sigver | SHA256 X9.31 RSA sign |
| sha384 | SHA-384 Hashing |
| sha384hmac | SHA384 HMAC sign |
| sha384rsasigver | SHA384 with RSA sign |
| sha384rsasigver | SHA384 with RSA sign |
| sha384rsax931sigver | SHA384 X9.31 RSA sign |
| sha384rsax931sigver | SHA384 X9.31 RSA sign |
| sha512 | SHA-512 Hashing |
| sha512hmac | SHA512 HMAC sign |
| sha512rsaoaepenc | SHA512 RSA OAEP encrypt |
| sha512rsapsssigver | SHA512 RSA PSS sign |
| sha512rsasigver | SHA512 with RSA sign |
| sha512rsax931sigver | SHA512 X9.31 RSA sign |
| sim3extractinsert | SIM3 Extract Insert masked objects |
| simextractinsert | SIMExtract Insert masked objects |
| simmultisign | SIMMultisign w/ masked key |
| sm3 | SM3 Hashing |
| sm3hmac | SM3 HMAC sign |
| symderive | Symmetric key derivation |
| symgen | Symmetric key generation |
| wrapunwrap | Wrap/unwrap operations |
| x942dhderive | X9.42 DH Derive |
| x942dhhybridderive | X9.42 DH Hybrid Derive |

| Mode | Description |
|---|---|
| x942dhkeygen | X9.42 DH Key Pair Generation |
| x942dhparamsgen | X9.42 DH Domain Parameter 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 followiong 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 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*.

# 6
# Pedserver and Pedclient

This chapter describes how to use the pedserver and pedcient utilities to manage your remote PED devices. It contains the following topics:



## Overview

You can use the pedserver and pedclient utilities to manage your remote PED devices.

### The pedserver Utility

The pedserver utility has one function. It resides on a computer with an attached SafeNet PED [Remote], and it serves PED operations to an instance of pedClient that operates on behalf of an HSM. The HSM could be local to the computer that has pedServer running, or it could be on another HSM host computer at some distant location.

See " The pedserver Command" on page 78.

### The pedclient Utility

The pedserver utility performs the following functions:

- It mediates between the HSM where it is installed and the SafeNet PED [Remote] where pedServer is installed, to provide PED services to the requesting HSM(s).
- It resides on a computer with RBS and an attached SafeNet Remote Backup HSM, and it connects with another instance of pedClient on a distant host of an HSM, to provide the link component for Remote Backup Service.

Thus, in the case where (say) an administrative workstation or laptop has both a Remote PED and a Remote Backup HSM attached, pedClient would perform double duty. It would link with a locally-running instance of pedServer, to convey HSM requests from the locally-connected Backup HSM to the locally-connected PED, and return the PED responses, As well, it would link a locally-running instance of RBS and a distant pedClient instance to mediate Remote Backup function for that distant HSM's partitions.

See " The pedclient Command" on the next page.

  

# The pedclient Command

## General Syntax

This is the syntax of the pedClient command, which includes starting and stopping of the service, and an assortment of configuration options. Specify "pedClient" at the command line, plus one of the modes, plus any option applicable to that mode.

```
[root@lunaclient101360 bin]# ./pedClient
Ped Client Version 2.0.0 (20000)

        Error: You must specify a mode.

Usage: pedClient [mode] [options...]

   Explanation of the modes:

     To query if a Ped Client is currently running, and to get details about
     the Ped Client, use this command:
        pedClient -m show [ options... ]

     To shut down an existing Ped Client, use this command:
        pedClient -m stop [ options... ]

     To start the Ped Client, use this command:
        pedClient -m start [ options... ]

     To start the Ped Client for Windows service, use this command:
        pedClient -m start -winservice [ options... ]

     To create a PED ID mapping, use this command:
        pedClient -m setid [ options... ]

     To test a PED ID mapping, use this command:
        pedClient -m testid [ options... ]

     To delete a PED ID mapping, use this command:
        pedClient -m deleteid [ options... ]

     To assign a PED ID mapping to an HSM, use this command:
        pedClient -m assignid [ options... ]

     To release a PED ID mapping from an HSM, use this command:
        pedClient -m releaseid [ options... ]

     To show the existing configuration file settings, use this command:
        pedClient -m config -show

     To restore the internal default configuration file settings, use this command:
        pedClient -m config -create

     To modify the existing configuration file settings, use this command:
        pedClient -m config -set [ options... ]

     To view a more detailed description of the Ped Client, use this command:
        pedClient -m desc
```

  

```
Explanation of the options:

Any options that are not specified on the command line will be read from
the config file.  If the config file cannot be found, internal default settings
will be used.  Invalid options do not generate an error and are ignored.

  -mode <mode>               -> Specifies the mode that the Ped Server will be
                                executed in.  The supported modes are "start",
                                "stop", "show", "setid", "testid",
                                "deleteid", "assignid", "releaseid" and
                                "config".
  -id                        -> Specifies the PED ID (larger then 0, less then 65535).
                                Applicable to the "setid", "testid", "deleteid",
                                "assignid" and"releaseid" modes.
  -id_ip                     -> Specifies the IP or hostname for the PED Server to
                                be linked to the specified PED ID.
                                Applicable to the "setid"  mode.
  -id_port                   -> Specifies the port for the PED Server to
                                be linked to the specified PED ID.
                                Applicable to the "setid"  mode.
  -id_serialnumber           -> Specifies the serial number of the HSM to be linked
                                to the specified PED ID.
                                Applicable to the "assignid" mode.
  -eadmin <0 or 1>           -> Specifies if the administration port is on
                                "localhost" or listening on the external host
                                name.
                                Applicable to "start", "stop", "show" and
                                "config set" modes.
  -admin <admin port number> -> Specifies the administration port number.
                                Applicable to "show" and "config set" modes.
  -set                       -> When used with "-config", specifies that the
                                configuration file should be updated with values
                                of the other supplied options.
                                Applicable to "config" mode.
  -show                      -> When used with "-config", specifies that the
                                contents of the configuration file should be displayed.
                                Applicable to "config" mode.
  -idletimeout<int>          -> Specifies the idle connection timeout in seconds.
                                Applicable to "start", "assignid" and "config set"
                                modes.
  -ignoreidletimeout         -> Specifies that the idle connection timeout should not
                                apply to the connection established for the specified
                                PED ID to HSM assignement.
                                Applicable to "assignid"  and "config set" modes.
  -socketreadtimeout <int>   -> Specifies the socket read timeout in seconds.
                                Applicable to "start", "stop", "show" and
                                "config set" modes.
  -socketwritetimeout <int>  -> Specifies the socket write timeout in seconds.
                                Applicable to "start", "stop", "show" and
                                "config set" modes.
  -shutdowntimeout <int>     -> Specifies the shutdown timeout in seconds for
                                internal services.
                                Applicable to "start", "stop" and
                                "config set" modes.
  -pstartuptimeout <int>     -> Specifies the startup timeout for the detached
```

  

```
                                process.
                                Applicable to "start", "stop" and
                                "config set" modes.
     -pshutdowntimeout <int>    -> Specifies the shutdown timeout for the detached
                                process.
                                Applicable to "start", "stop" and
                                "config set" modes.
     -loginfo <0 or 1>          -> Specifies if the logger should log "info" messages.
                                Applicable to all modes.
     -logwarning <0 or 1>       -> Specifies if the logger should log "warning" messages.
                                Applicable to all modes.
     -logerror <0 or 1>         -> Specifies if the logger should log "error" messages.
                                Applicable to all modes.
     -logtrace <0 or 1>         -> Specifies if the logger should log "trace" messages.
                                Applicable to all modes.
     -logfilename <filename>    -> Specifies the log file name.
                                Applicable to all modes.
     -maxlogfilesize <size>     -> Specifies the maximum log file size in KB
                                Applicable to all modes.
     -locallogger               -> Specifies that the Remote Ped logger should be used,
                                not the IS logging system.
                                Applicable to all modes.

[admin@myluna bin]#
```

pedClient must run on any host of an HSM that needs to be served by a Remote PED.

pedClient must run on any host of a Remote Backup HSM that will be serving remote primary HSMs*.

* A distant HSM that appears as a crypto slot at the host of the Backup HSM is not considered "remote" in this sense, and so the Backup HSM's host does not need RBS. This would be the case for (say) a SafeNet Network HSM partition where the Remote Backup workstation is a registered client of the partition, and therefore has a Network Trust Link (NTL) with the SafeNet Network HSM appliance. In that case, a lunacm session on the Backup workstation sees the SafeNet Network HSM's partition as just another "local" slot. A slot-to-slot backup operation launched by lunacm at the Backup workstation is a local operation, as is a restore operation. That client relationship implies that the Backup workstation's administrator is entrusted with the partition authentication (black PED Key, challenge secret, red PED Key) for the partition on that distant SafeNet Network HSM. In many cases, that is a perfectly legitimate assumption. The partition is registered with two "clients" - one is the working, or production client that uses the partition for cryptographic operations; the other is the Backup workstation that connects with the partition only when it is time to perform backup or restore activity.

If, instead, the administrator of the Remote Backup HSM was not entrusted with the authentication secrets of the distant HSM partition, then the administrator could still perform a backup, but it would proceed differently. The backup administrator could connect by SSH or RDP session to a legitimate client computer and use lunacm at that client to launch the backup. The client, already authenticated to the activated SafeNet Network HSM partition would see the partition as a local slot, but would see the backup HSM and its attached SafeNet Remote Backup HSM only through the intermediary Remote Backup Service (rbs) running on that Backup workstation and conversing with the distant client computer by means of pedClient instances at each end. This is one version of the method used when the organization (or its customer) prefers a strict separation of roles.

A variant of the RBS method might work from the other direction, with the owner of the client computer doing the work, and the owner of the administrative/backup workstation simply allowing the client to take over the admin/backup workstation for the duration of the backup-or-restore operation. In either case, RBS must reside on the computer with the SafeNet Remote Backup HSM attached, and pedClient must run on both.

The various methods have their place, depending on your organization's structure and security protocols.

 

See "Remote Application Partition Backup and Restore Using the Backup HSM" on page 1 in the *Administration Guide* for more information.

## PedClient on SafeNet Network HSM

PedClient exists on the SafeNet Network HSM appliance, but is not directly exposed. Instead, the relevant features are accessed via lunash **hsm ped** commands.

# The pedserver Command

## Syntax

This is the syntax of the pedServer command, which includes starting and stopping of the service, and an assortment of configuration options. Specify "pedserver" at the command line, plus one of the modes, plus any option applicable to that mode.

```
pedServer.exe   -mode {start | stop | connect | disconnect | show | config }
               -ip <ip address>
               [-port <port number>]
               [-force]
```

**Note:** When running **pedserver -mode start** on an IPv6 network, you must include the **-ip** <IPv6_address> option.

```
pedServer -appliance register
             -name <unique name>
             -certificate <Network HSM certificate file>
             -ip <network-hsm-ip>
             [-port <port number>]
```

**Note:** The -name parameter must be alphanumeric only: 0 through 9 or a through z or A through Z
No punctuation or special characters are permitted.

```
pedServer -appliance delete
             -name <unique name>
             [-force]

pedServer -appliance list

pedServer -regen -commonname [-force]
```

  

**Note:** When registering, the default port 9697 is assumed. However in the special case where another application already uses port 9697, port forwarding in a router could remap a different incoming port number (that you provide in the **-appliance register** command) to 9697 when forwarded to the SafeNet Network HSM.

```
C:\Program Files\SafeNet\LunaClient>pedserver -h
Ped Server Version 1.0.6 (10006)

Usage: pedServer [mode] [options...]

   Explanation of the modes:

     To query if a Ped Server is currently running, and to get details about
     the Ped Server, use this command:
        pedServer -mode show [ options... ]

     To shut down an existing Ped Server, use this command:
        pedServer -mode stop [ options... ]

     To start the Ped Server, use this command:
        pedServer -mode start [ options... ]

     To show the existing configuration file settings, use this command:
        pedServer -mode config -show

     To restore the internal default configuration file settings, use this command:
        pedServer -mode config -create [ options... ]

     To modify the existing configuration file settings, use this command:
        pedServer -mode config -set [ options... ]

     To view a more detailed description of the Ped Server, use this command:
        pedServer -mode desc

     To connect to a Luna SA server or a PedClient (making a connection from pedServer to
     Luna SA/PedClient), use this command:
        pedServer -mode connect -name <label>

     To disconnect from Luna SA server or a PedClient and start in service mode, use this com-
mand:
        pedServer -mode disconnect

     To register a Luna SA certificate or a PedClient , use this command:
        pedServer -appliance register -name <label> -ip <connection IP address> -certificate
<certif
icate file> [-port <port number>]

     To delete a registered Luna SA server or a PedClient, use this command:
        pedServer -appliance delete -name <label> [-force]

     To list all registered Luna SA servers and PedClients, use this command:
        pedServer -appliance list

     To regenerate the client certificate, use this command:
        pedServer -regen -commonname <common name> [-force]
```

  

```
   Explanation of the options:

   Any options that are not specified on the command line will be read from
   the config file.  If the config file cannot be found, internal default settings
   will be used.  Invalid options do not generate an error and are ignored.

     -mode <mode>               -> Specifies the mode that the Ped Server will be
                                   executed in.  The supported modes are "start",
                                   "stop", "show", "config", "connect" and "disconnect".
     -configfile <filename>     -> Specifies the config file to use.  Applicable to all
                                   modes.
     -appliance                 -> Certificate management of Luna SA servers and PedClients. The
fol
lowing
                                   actions are "register", "delete" and "list".
     -regen                     -> Regenerate the client certificate. The client certificate
                                   path is specified in Chrystoki configuration file.
     -eserverport <0 or 1>      -> Specifies if the server port is on "localhost"
                                   or listening on the external host name.  Applicable
                                   to "start" and "config set" modes.
     -port <server port>        -> Specifies the server port number.  Applicable to
                                   "start", "show" and "config set" modes.
     -ip <server IP>            -> Specifies the server listening IP address if the mode is set
                                   to "start" and "config set" modes.
     -eadmin <0 or 1>           -> Specifies if the administration port is on
                                   "localhost" or listening on the external host
                                   name.  Applicable to "start" and "config set" modes.
     -admin <admin port number> -> Specifies the administration port number.
                                   Applicable to "start", "stop", and "show" modes.
     -force                     -> When used with "-start", specifies that any existing
                                   Ped Server currently running should be shutdown and a new
                                   Ped Server started.  Applicable to "start" mode.
     -set                       -> When used with "-config", specifies that the
                                   configuration file should be updated with values
                                   of the other supplied options.  Applicable to "config"
     -show                      -> When used with "-config", specifies that the
                                   contents of the configuration file should be displayed.
                                   Applicable to "config" mode.
     -idletimeout<int>          -> Specifies the idle connection timeout in seconds.
                                   Applicable to "start" and "config set" modes.
     -socketreadtimeout <int>   -> Specifies the socket read timeout in seconds.
                                   Applicable to "start", "stop", "show" and
                                   "config set" modes.
     -socketwritetimeout <int>  -> Specifies the socket write timeout in seconds.
                                   Applicable to "start", "stop", "show" and
                                   "config set" modes.
     -internalshutdowntimeout <int> -> Specifies the shutdown timeout in seconds for
                                   internal services.
                                   Applicable to "start", "stop" and "config set"
                                   modes.
     -bgprocessstartuptimeout <int> -> Specifies the startup timeout for the detached
                                   process.
                                   Applicable to "start", "stop" and "config set"
                                   modes.
     -bgprocessshutdowntimeout <int> -> Specifies the shutdown timeout for the detached
```

 

```
                                process.
                                Applicable to "start", "stop" and "config set"
                                modes.
   -loginfo <0 or 1>          -> Specifies if the logger should log "info" messages.
                                Applicable to all modes.
   -logwarning <0 or 1>       -> Specifies if the logger should log "warning" messages.
                                Applicable to all modes.
   -logerror <0 or 1>         -> Specifies if the logger should log "error" messages.
                                Applicable to all modes.
   -logtrace <0 or 1>         -> Specifies if the logger should log "trace" messages.
                                Applicable to all modes.
   -logfilename <filename>    -> Specifies the log file name.
                                Applicable to all modes.
   -maxlogfilesize <size>     -> Specifies the maximum log file size in KB
                                Applicable to all modes.
   -pinginterval <int>        -> Specifies the interval in seconds for ping commands.
                                Applicable to "start" and "config set" modes.
   -pongtimeout <int>         -> Specifies timeout in seconds for the ping response.
                                Applicable to "start" and "config set" modes.


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

## Sample Outputs

Commands you are likely to use most often are PedServer mode start, to launch the service, when working in Client/Server mode, and PedServer mode show, to display its current status.

```
C:\Program Files\Safenet\LunaClient>PedServer.exe mode start
Ped Server Version 1.0.5 (10005)
Failed to load configuration file.  Using default settings.
Ped Server launched in startup mode.
Starting background process
Background process started
Ped Server Process created, exiting this process.
C:\Program Files\Safenet\LunaClient>

C:\Program Files\Safenet\LunaClient>PedServer.exe mode show
Ped Server Version 1.0.5 (10005)
Failed to load configuration file.  Using default settings.
Ped Server launched in status mode.
   Server Information:
      Hostname:                          OTT1-202311
      IP:                                192.20.10.190
      Firmware Version:                  2.5.0-1
      PedII Protocol Version:            1.0.1-0
      Software Version:                  1.0.5 (10005)
      Ped2 Connection Status:            Connected
      Ped2 RPK Count                     1
      Ped2 RPK Serial Numbers            (5b420100834a2301)
   Client Information:                   Not Available
   Operating Information:
      Server Port:                       1503
      External Server Interface:         Yes
      Admin Port:                        1502
      External Admin Interface:          No
```

 

```
      Server Up Time:                    8 (secs)
      Server Idle Time:                  8 (secs) (100%)
      Idle Timeout Value:                1800 (secs)
      Current Connection Time:           0 (secs)
      Current Connection Idle Time:      0 (secs)
      Current Connection Total Idle Time: 0 (secs) (100%)
      Total Connection Time:             0 (secs)
      Total Connection Idle Time:        0 (secs) (100%)
Show command passed.

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

It might be necessary to regenerate the PedServer certificate:

```
C:\Program Files\SafeNet\LunaClient>PedServer.exe -regen -commonname 24.240_server -force

Ped Server Version 1.0.6 (10006)
Private Key created and written to: C:\Program Files\SafeNet\LunaClient\cert\client\24.240_serv-
er.pemKey
Certificate created and written to: C:\Program Files\SafeNet\LunaClient\cert\client\24.240_
server.pem

Successfully regenerated the client certificate.
```

As well, you might have need to delete an appliance from the registered list

```
C:\Program Files\SafeNet\LunaClient>PedServer.exe -appliance delete -name SA62 -force

Ped Server Version 1.0.6 (10006)

Successfully deleted the registered appliance: SA62
```

PedServer is required to run on any computer that has a SafeNet Remote PED attached, and is providing PED services.

PedServer always works with an instance of PedClient.

PedClient could be running on a distant HSM host computer, or it could be running on the same computer that has the Remote PED attached and PedServer running. This would normally be the case where a SafeNet Remote Backup HSM or other HSM is also attached or embedded. In other words, the one computer could be carrying on both halves of the PedClient/PedServer conversation over two ports in its own memory.

PedServer can also run in peer-to-peer mode, where the server initiates the connection to the Client. This is needed when the Client (usually SafeNet Network HSM) is behind a firewall that forbids outgoing initiation of connections.

See "Remote Application Partition Backup and Restore Using the Backup HSM" on page 1 in the *Administration Guide* for more information.

 

# 7
# 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

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

RBS requires pedClient to be running both on the RBS computer and on the host of the SafeNet 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.

**Note:** This feature is not currently supported for use with IPv6 networks.

See "Pedserver and Pedclient" on page 74 for more information.

  

# rbs

Access the RBS commands.

## Syntax

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

| Parameter | Shortcut | Description |
|---|---|---|
| **-config** | **-c** | Runs RBS to select devices to support for Remote Backup. See "rbs config" on the next page. |
| **-daemon** | **-d** | Runs RBS in daemon (background) mode. See "rbs daemon" on page 86. |
| **-genkey** | **-g** | Runs RBS to generate private key/certificate for Remote Backup. See "rbs genkey" on page 87. |
| **-help** | **-h** | Displays help information for the rbs command. |
| **-nopassword** | **-n** | Require no password for encoded keys. See "rbs nopassword" on page 88. |

## 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 RBSin 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]#
```

# 8
# SAlogin

Cryptographic applications that are not specifically adapted to use an HSM Server can nevertheless be run using SafeNet Enterprise 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 HSM appliance with openssl, which can be used with HSMs, but which has no inherent ability to provide credentials to the HSM.

## 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
```

**Note:** The applications in the integrations documents have been explicitly integrated by SafeNet, to work with your SafeNet 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.

## 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.

# 9
# SCP and PSCP

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 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>

> **Note:** When using **scp** or **pscp** over an IPv6 network, enclose addresses in square brackets.

## 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 passw** [] login with specified password

**-unsafe** [] allow server-side wildcards (DANGEROUS)

  

# Examples

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

## Transferring a file from a SafeNet HSM client to a SafeNet 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 Network HSM to a SafeNet 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".

# 10
# Ureset

You can use the **ureset** command to reset a SafeNet USB HSM that has become unresponsive, without having to reboot the host computer.

## Syntax

**ureset** [<device>]

| Parameter | Description |
|---|---|
| <device> | Specifies the handle of the device that you want to reset. This parameter is not required if you have only one SafeNet USB HSM connected to the host computer, and is optional if you have multiple SafeNet USB HSMs connected.<br>If you have multiple SafeNet USB HSMs connected, you can use this parameter to specify which HSM to reset, as follows:<br>• **/dev/lunauhd0** - reset the first SafeNet USB HSM listed by the **slot list** command.<br>• **/dev/lunauhd1** - reset the second SafeNet USB HSM listed by the **slot list** command.<br>• **/dev/lunauhd2** - reset the third SafeNet USB HSM listed by the **slot list** command.<br>If you have multiple SafeNet USB HSMs installed, and do not specify this parameter, then the first SafeNet USB HSM listed by the **slot list** command is reset. |

## Example

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

        Available HSMs:

        Slot Id ->              0
        Label ->                myG5par
        Serial Number ->        16302360890475
        Model ->                G5Base
        Firmware Version ->     6.22.0
        Configuration ->        Luna User Partition With SO (PED) Signing With Cloning Mode
        Slot Description ->     User Token Slot

        Slot Id ->              1
        Label ->                SafeG5
        Serial Number ->        7001812
        Model ->                G5Base
        Firmware Version ->     6.22.0
        Configuration ->        Luna HSM Admin Partition (PED) Signing With Cloning Mode
        Slot Description ->     Admin Token Slot
        HSM Configuration ->    Luna HSM Admin Partition (PED)
        HSM Status ->           OK
```

  

```
        Current Slot Id: 0

lunacm:>exit

C:\Program Files\SafeNet\LunaClient>ureset /dev/lunauhd1

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

# 11
# Vreset

You can use the **vreset** command to reset a SafeNet PCIe HSM that has become unresponsive, without having to reboot the host computer.

## Syntax

**vreset** [<device>]

| Parameter | Description |
|---|---|
| <device> | Specifies the handle of the device that you want to reset. This parameter is not required if you have only one SafeNet PCIe HSM installed in the host computer, and is optional if you have multiple SafeNet PCIe HSMs installed.<br>If you have multiple SafeNet PCIe HSMs installed, you can use this parameter to specify which HSM to reset, as follows:<br>• **/dev/viper0** - reset the first SafeNet PCIe HSM listed by the **slot list** command.<br>• **/dev/viper1** - reset the second SafeNet PCIe HSM listed by the **slot list** command.<br>• **/dev/viper2** - reset the third SafeNet PCIe HSM listed by the **slot list** command.<br>If you have multiple SafeNet PCIe HSMs installed, and do not specify this parameter, then the first SafeNet PCIe HSM listed by the **slot list** command is reset. |

## Example

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

        Available HSMs:

        Slot Id ->              2
        Tunnel Slot Id ->       4
        Label ->                mypciepsopar
        Serial Number ->        349297122736
        Model ->                K6 Base
        Firmware Version ->     6.22.0
        Configuration ->        Luna User Partition With SO (PED) Signing With Cloning Mode
        Slot Description ->     User Token Slot

        Slot Id ->              3
        Tunnel Slot Id ->       4
        Label ->                mypcie6
        Serial Number ->        150022
        Model ->                K6 Base
        Firmware Version ->     6.22.0
        Configuration ->        Luna HSM Admin Partition (PED) Signing With Cloning Mode
        Slot Description ->     Admin Token Slot
```

  

```
        HSM Configuration ->    Luna HSM Admin Partition (PED)
        HSM Status ->           OK

        Current Slot Id: 0

lunacm:>exit

C:\Program Files\SafeNet\LunaClient>vreset /dev/viper01

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

  

# 12
# VTL

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

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 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.

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

**Note:** Just as you need to be root or Administrator (or a user with equivalent privileges) when installing, you need to be root or 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 Network HSM Servers (you need to have already imported the server certificate from each SafeNet Network HSM that you wish to add). See "vtl addServer " on page 100. |
| deleteServer | Use this command to remove a server/host from the client's list of trusted SafeNet Network HSM Servers. See "vtl deleteServer " on page 117. |
| replaceServer | Use this command to replace a named server/host from the client's list of trusted SafeNet 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 " vtl replaceserver" on page 149. |
| listServers | Use this command to display a list of the SafeNet Network HSM Servers trusted by this client. See "vtl listServers " on page 145. |
| 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 "vtl createCert" on page 115. |
| listSlots | List all PKCS#11 cryptographic device slots that can be seen at this time. See "vtl listSlots " on page 146. |
| verify | Use this command to verify the SafeNet Network HSM Servers slots or partitions that are visible. See "vtl verify " on page 151. |
| haAdmin | Use these commands to create and manage HA groups of several SafeNet Network HSM appliances, providing load-sharing and redundancy for the cryptographic operations required by this client. See "vtl haAdmin" on page 121 |

  

| Subcommand | Description |
|---|---|
| fingerprint | Use this command to display the fingerprint of a specified certificate. See "vtl fingerprint " on page 120. |
| examineCert | Use this command to display the details of a specified certificate. See "vtl examineCert " on page 118. |
| backup | Administer backup/backup HSM features - used for Remote Backup (where the backup HSM is connected to a computer remote from your SafeNet Network HSM. If you connect the SafeNet Backup HSM directly to your SafeNet Network HSM, then use the SafeNet Shell command line and "token backup" commands, instead of VTL. See "vtl backup" on page 101. |
| supportInfo | Use this command to create a support information file, when one is requested by SafeNet Customer Support. See "vtl supportInfo " on page 150. |
| logging | Configure logging for Windows computers. See "vtl logging configure " on page 147. |

# vtl addServer

## Name

vtl addServer

## Syntax

vtl addServer -n <server hostname> -c <servers cert filename> [-htl]

## 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.

**-htl** [optional] Require HTL (host trust link) an additional layer of trust, that separates the link authentication from specific hardware, allowing trust of virtual clients, such as in cloud environments - also can be used to enhance trust for non-virtual/non-cloud clients.

## Example

$ ./vtl add -n 192.20.9.161 -c server161.pem -htl
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 Network HSM partitions. See "partition backup" on page 1 in the *Lunacm Command Reference Guide*.

## Name

vtl backup - backup SafeNet 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 Network HSM partition contents to a specified slot or labeled partition on a backup HSM.

See also:

- "vtl backup append" on page 103
- "vtl backup restore " on page 106
- "vtl backup delete " on page 105
- "vtl backup token" on page 107

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 Network HSM partition to a SafeNet Backup HSM that is connected **locally** to the SafeNet 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 Network HSM partition contents to a specified slot or labeled partition on a backup HSM.

See also:

- "vtl backup" on page 101
- "vtl backup restore " on page 106
- "vtl backup delete " on page 105
- "vtl backup token" on page 107

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 Network HSM partition to a SafeNet Backup HSM that is connected **locally** to the SafeNet 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 Network HSM partition contents from a specified slot or labeled partition on a backup HSM.

See also:

- "vtl backup" on page 101
- "vtl backup restore " on the next page
- "vtl backup append" on page 103
- "vtl backup token" on page 107

## 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 Network HSM partition contents from a specified slot or labeled partition on a backup HSM. See also:

"vtl backup" on page 101

"vtl backup append" on page 103

"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 "vtl backup token factoryreset " on the next page. |
| init | Initialize backup token. See "vtl backup token init " on page 109. |
| resize | Resize backup token container. See "vtl backup token resize " on page 110. |
| show licenses | Show all licenses for backup token. see "vtl backup token show licenses " on page 112. |
| show | Show all SafeNet UHD slots for backup token. See "vtl backup token show " on page 111 |
| update | Update backup token. See "vtl backup token update " on page 113. |

  

# 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 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 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 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 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 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.)

**-s <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

Exponent: 65537 (0x10001)
```

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 "hagroup creategroup" on page 1 |
| deleteGroup | Disabled. See lunacm command "hagroup deletegroup" on page 1. |
| addMember | Disabled. See lunacm command "hagroup addmember" on page 1. |
| standbyMembers | Disabled. See lunacm command "hagroup addstandby" on page 1. |
| removeMember | Disabled. See lunacm command "hagroup removemember" on page 1. |
| synchronize | Disabled. See lunacm command "hagroup synchronize" on page 1. |
| recover | Disabled. See lunacm command "hagroup recover" on page 1. |
| autoRecovery | Disabled. See lunacm command "hagroup retry" on page 1. |
| HALog | Disabled. See lunacm command "hagroup halog" on page 1. |
| HAOnly | Disabled. See lunacm command "hagroup haonly" on page 1. |
| show | Disabled. See lunacm command "hagroup listgroups" on page 1. |

  

# vtl haAdmin addMember

**Note:** This command is disabled. See lunacm 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 deletegroup" 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 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 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 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 "vtl haAdmin HAOnly enable " on page 131. |
| -disable | Show all slots when retrieving slot lists. See "vtl haAdmin HAOnly disable " 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 it's 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 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 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 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 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_proxy_commands

Delete this text and replace it with your own content.

# vtl haAdmin proxy disable

## Name

( FOR FUTURE USE)

vtl haAdmin -proxy - disable - stop using Proxy Token Agent for HA.

## Syntax

vtl haAdmin -proxy -disable

## Description

Configures the client to discontinue using the Proxy Token Agent for HA .

You have full control over whether to configure your system to use the traditional HA solution or to use the new Proxy Token Agent HA design. Note, however, that you can only use one HA solution: you cannot configure some HA groups to use the traditional method and some to use the new method. All HA groups need to use one or the other method.

## OPTIONS >

**None**

## Example

vtl haAdmin proxy -disable

proxy disabled.

.

# vtl haAdmin proxy enable

## Name

( FOR FUTURE USE)

vtl haAdmin -proxy - enable - use Proxy Token Agent for HA.

## Syntax

vtl haAdmin -proxy -enable

## Description

Configures the client to use the Proxy Token Agent for HA .

You have full control over whether to configure your system to use the traditional HA solution or to use the new Proxy Token Agent HA design. Note, however, that you can only use one HA solution: you cannot configure some HA groups to use the traditional method and some to use the new method. All HA groups need to use one or the other method.

## OPTIONS >

**None**

## Example

vtl haAdmin proxy -enable

proxy enabled.

.

# 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 "vtl haAdmin standbyMembers -set " on page 143. |
| -remove | Use a list of serial numbers to remove partitions as standby members from a named HA group. See "vtl haAdmin standbyMembers -remove " 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 lunacm 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 examine 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: lunasa1
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 Network HSM
- slot 1 is the application partition on a SafeNet PCIe HSM with firmware 6.22.0 or newer
- slot 2 is the HSM administrative partition of the same SafeNet PCIe HSM
- slot 4 is a SafeNet 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 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 Network HSM appliance.
4. Manually delete the client certificate file.
5. Run ckemo or another application against a partition on that SafeNet 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> -n <new server hostname> -c <certificate filespec> [-htl]

## 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>** [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>** [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 filespec>** [mandatory] The name (including the path to its location on your computer) of the server's certificate file.

**-htl** [optional] Require HTL (host trust link) an additional layer of trust, that separates the link authentication from specific hardware, allowing trust of virtual clients, such as in cloud environments - also can be used to enhance trust for non-virtual/non-cloud clients.

## Example

```
bash # ./vtl replaceServer -o yourluna -n myluna -c server.pem -htl
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 Network HSM> vtl supportInfo
Creating client-side support information file now...
'vtl supportInfo' completed.
File "c_supportInfo.txt" created.
C:\Program Files\SafeNet Network HSM>
```

# vtl verify

## Name

vtl verify

## Syntax

vtl verify

## Description

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

## Options

None.

## Example

```
            bash-2.03# ./vtl verify

The following SafeNet Network HSM Slots/Partitions were found:
Slot    Serial #        Label
====    ========        =====
 1      65091001        MyPartition
 2      65097001        YourPartition
 3      65093001        HisPartition
bash-2.03#
```