

u.trust Anchor FIPS 140-3

cxitool Manual



Imprint

Copyright 2024	Utimaco IS GmbH Germanusstr. 4 D-52080 Aachen Germany
Phone	AMERICAS +1-844-UTIMACO (+1 844-884-6226) EMEA +49 800-627-3081 APAC +81 800-919-1301
Internet e-mail	https://support.hsm.utimaco.com/ support@utimaco.com

Document Version	1.0.2
Product Version	6.0.0
Date	2024-10-24
Document No.	2024-0012
Status	PUBLISHED

All rights reserved

No part of this documentation may be reproduced in any form (printing, photocopy or according to any other process) without the written approval of Utimaco IS GmbH or be processed, reproduced or distributed using electronic systems.

Utimaco IS GmbH reserves the right to modify or amend the documentation at any time without prior notice. Utimaco IS GmbH assumes no liability for typographical errors and damages incurred due to them. Any mention of the company name Utimaco in this documents refers to the Utimaco IS GmbH.

All trademarks and registered trademarks are the property of their respective owners.

Table of Contents

1	Introduction	5
1.1	About this Manual	5
1.1.1	Target Audience for this Manual	5
1.1.2	Document Conventions	5
1.1.3	Abbreviations	6
2	General	7
2.1	Features	7
2.2	System Requirements	7
2.3	Installation and Update of cxitool.....	9
2.4	Syntax of cxitool Commands.....	9
2.5	Authentication Key Specifiers	11
2.6	Password Entry	14
2.7	Cryptographic Keys and Key Groups	15
3	Commands and Parameters	21
3.1	Basic Commands.....	21
3.1.1	Help	21
3.1.2	Version	22
3.1.3	Authentication Commands	22
3.1.3.1	LogonPass.....	25
3.1.3.2	LogonSign	26
3.2	Special Parameters	28
3.2.1	Export	28
3.2.2	Usage.....	29
3.2.3	Timeout	32
3.3	CXI Firmware Configuration	33
3.3.1	GetConfig.....	33
3.3.2	SetConfig.....	35
3.3.3	ResetConfig	42
3.3.4	SecureGroupPass	43
3.4	Key Management Commands.....	43
3.4.1	ListKeys	44
3.4.2	KeyInfo	47
3.4.3	GenerateKey	55

3.4.4	DeleteKey	59
3.4.5	BackupKey.....	61
3.4.6	RestoreKey	65
3.4.7	ExportPubKeyFile	67
3.4.8	CopyKeyStore	69
3.4.9	KeyMigration	70
3.5	Certificate Management Commands	72
3.5.1	ExportP10.....	72
3.5.2	ExportCert	76
3.5.3	ImportCert	78
3.5.4	ImportP12.....	80
3.5.5	SelfSignedCert.....	82
3.6	Cryptography Commands	87
3.6.1	Encrypt	87
3.6.2	Decrypt	89
3.6.3	Hash	92
3.6.4	Sign	93
3.6.5	Verify	95
3.7	External Keystore Commands	98
3.7.1	ConfigODBC	98
3.7.2	CreateDBSchema.....	99
3.7.3	ListODBCDrivers	100
3.8	FIPS-specific Commands	100
3.8.1	SetFipsUsage	100
3.9	CC-specific Commands	104
3.9.1	SetCCUsage	104
4	Built-in Elliptic Curves.....	105
5	Contact Address for Support Queries	106

1 Introduction

Thank you for purchasing our u.trust Anchor security system. We hope you are satisfied with our product. Please do not hesitate to contact us if you have any questions or comments.

1.1 About this Manual

1.1.1 Target Audience for this Manual

This manual is primarily intended for system administrators who use cxitool, a command-line utility provided by Utimaco, to perform cryptographic tasks like generating cryptographic keys, encrypting, decrypting and signing data and verifying signatures.

1.1.2 Document Conventions

We use the following document conventions:

<i>Convention</i>	<i>Use</i>	<i>Example</i>
Bold	Items of the Graphical User Interface (GUI), e.g., menu options	Press OK
<code>Monospaced</code>	Code that is given for explanation or as an example, file paths	<code>chsm-create</code>
<i>Italic</i>	References and important terms	See <i>Sample Chapter</i> in the <i>CryptoServer - Sample Manual</i>

Table 1: Document conventions

We use special icons to highlight the most important notes and information.



Here you find important safety information that should be followed.



Here you find additional notes or supplementary information.



This message marks the result expected after the successful execution of an instruction.

1.1.3 Abbreviations

We use the following abbreviations in this manual:

<i>Abbreviation</i>	<i>Description</i>
AES	Advanced Encryption Standard
CC	Common Criteria
cHSM	Containerized Hardware Security Module
CXI	Cryptographic eXtended Interface
csadm	CryptoServer command-line administration tool
DES	Data Encryption Standard
DRNG	Deterministic random number generator
DSA	Digital Signature Algorithm
EC	Elliptic curve
ECDSA	Elliptic Curve DSA
FIPS	Federal Information Processing Standard
HMAC	Keyed-Hash Message Authentication Code
HSM	Hardware security module
MAC	Message authentication code
MBK	Master Backup Key
NTP	Network Time Protocol
P11CAT	PKCS#11 CryptoServer Administration Tool
PCIe	PCI Express Interface
PKCS	Public key cryptography standard
RSA	Rivest, Shamir, Adleman (cryptosystem)

Table 2: Abbreviations

2 General

This chapter contains information about system requirements, installation and general information about the cxitool.

2.1 Features

The cxitool offers for example the following features:

- Basics
 - Retrieving the version of cxitool and the CXI API
 - User authentication
- Key Management
 - Generating keys (AES, RSA and EC)
 - Retrieving key properties
 - Backing up and restoring keys
 - Deleting keys
- Cryptography
 - Encrypting and decrypting
 - Signing and verifying signatures (ECDSA and RSA)
 - Hashing
- Certificate handling
 - Generating self-signed certificates
 - Importing and exporting certificates
 - Importing and exporting keys

2.2 System Requirements

Hardware (PC)

- No special requirements as far as memory and CPU performance are concerned.
- Network interface card to access u.trust Anchor LAN FIPS 140-3

- A free USB port is needed to connect the PIN pad (smartcard reader with keyboard and display).

If RSA or ECDSA signature authentication with a smartcard is used, this port must either be available on the computer cxitool is running on or on another computer. If a PIN pad is used at another computer, follow the instructions in *Using a PIN Pad* in the u.trust Anchor - Containerized Hardware Security Module (cHSM) - Administration Manual. The instructions for csadm commands in that section can be applied in an analog way to cxitool commands.

Software (OS)

<i>Operating System</i>	<i>CryptoServer</i>	<i>u.trust Anchor</i>
Windows x64		
Windows 10	✓	✓
Windows 11 (Pro)	✓	✓
Windows Server 2016	✓	✓
Windows Server 2019	✓	✓
Windows Server 2022 (Standard)	✓	✓
Linux x64		
Red Hat Enterprise Linux 8	✓	✓
Red Hat Enterprise Linux 9	✓	✓
SUSE Linux Enterprise Server 12	✓	✓
SUSE Linux Enterprise Server 15	✓	✓
Ubuntu 20.04 LTS	✓	✓
Ubuntu 22.04 LTS	✓	✓



You can find the list of all supported operating systems in the document [CS_PD_SecurityServer_Supported_Platforms.pdf](#) in the u.trust Anchor product bundle in the directory `...\Documentation\Product Details`.

2.3 Installation and Update of cxitool

To install or update cxitool, copy the following file from the u.trust Anchor FIPS 140-3 bundle onto your host computer:

- Windows
 - The Windows Installer provides the installation and update of cxitool.
- Linux
 - `.../Software/Linux/Administration`
This directory is to be found on the general SecurityServer bundle.

To start cxitool, open a command line and execute the copied file.

2.4 Syntax of cxitool Commands

The syntax of a cxitool command is according to the following scheme:

```
cxitool [Dev=...] <param1>[=...] <param2>[=...] ... <command1>[=...]  
<command2>[=...] ...
```

Consider the following:

- Parameters and commands are processed from left to right.
- If a subsequent command requires to set a parameter or to run another command prior to its own execution, it will have to be entered rightmost.
- Expressions in squared brackets [] are optional.
- In the syntax description of the individual command, all parameters are shown in angle brackets < > and are explained later.
- Some parameters or commands require an assigned value ('=...'), some do not.
- Some commands use a default value if none is given ('[=...]').



Certain types of shell processes treat certain characters (commas, colons, semi-colons) differently. If the execution of a cxitool command fails with an error message from the shell about an illegal parameter format, quoting parameter values may be necessary.

The following is an example of a correct cxitool command entry in the Microsoft PowerShell:

```
cxitool [Dev=<device>] LogonSign="<user>,<keyspec>" <command>
```

Examples

```
cxitool Dev= /dev/cs2. 0.1 ...
cxitool Dev= /dev/cs2. 0.2 LogonSign=ADMIN,d:\keys\cs2\ADMIN.key ...
cxitool Dev= /dev/cs2. 0.3 LogonPass=KeyMgr,ask ...
cxitool LogonPass=KeyUsr,123456 ...
cxitool Dev=4001@192.168.1.98 LogonSign=ADMIN,:cs2:cjo:USB0 ...
cxitool Dev=4002@192.168.1.1 LogonPass=SOAll,123456 GetConfig Group=1234 GetConfig
Group=asdf GetConfig
```

Almost every command addressing the device requires the `Dev=` device parameter, which sets the address of the device. Commands running locally without using a u.trust Anchor cHSM (like module preparation commands) do not need this parameter. Possible values are:

Device address	Description
<code>/dev/cs2.n.m</code> where n = 0 and m = {1, 2, ..., 31}	n+1: No. of local u.trust Anchor PCIe card on a UNIX system. m: No. of cHSM on a local u.trust Anchor PCIe card on a UNIX system.
<code>PCI:n</code> where n = {0, 1, 2, ..., 31}	u.trust Anchor PCIe card No. n+1 on a Windows system
<code>4001@194.168.4.107</code>	Port number of a cHSM and the IP address of the u.trust Anchor LAN. Typical: 4001 = first cHSM, 4002 = second cHSM etc. In cxitool commands, always use IP addresses without leading zeros although they are shown in the LAN display, e.g., 194.168.004.107 .
<code>TCP:288@194.168.4.107</code>	IP address and port number of a LAN device In cxitool commands, always use IP addresses without leading zeros although they are shown in the LAN display, e.g., 194.168.004.107 .
<code>TCP:194.168.4.107</code>	IP address of a LAN device (default: port=288) In cxitool commands, always use IP addresses without leading zeros although they are shown in the LAN display, e.g., 194.168.004.107 .
<code>194.168.4.107</code>	IP address of a LAN device (default: protocol=TCP, port=288) In cxitool commands, always use IP addresses without leading zeros although they are shown in the LAN display, e.g., 194.168.004.107 .

Device address	Description
TCP:288@cslan01	Host name and port number of a LAN device (using DNS request to resolve host name)
TCP:cslan01	Host name of a LAN device (using DNS request to resolve host name, default: port=288)
cslan01	Host name of a LAN device (using DNS request to resolve host name, default: protocol=TCP, port=288)
TCP:3001@127.0.0.1 or TCP:3001@localhost	Local device simulator for Windows/Linux (SDK)
3001@127.0.0.1 or 3001@localhost	Local device simulator for Windows/Linux (SDK) with the default protocol TCP

Table 3: Example values for the Device parameter



If the environment variable CRYPTOSERVER is set according to the above mentioned syntax, the 'Dev=' parameter can be skipped. Using the 'Dev=' parameter overrides the CRYPTOSERVER environment variable in any case for the specific command.

2.5 Authentication Key Specifiers



Authentication keys must not be confused with cryptographic keys. Authentication keys are used to ensure that only users with the needed permissions can perform a cxitool command. Cryptographic keys are used to encrypt, decrypt, sign or verify data, see [Cryptographic Keys and Key Groups](#).

Some of the cxitool commands use a private RSA or ECDSA key to authenticate a command. The cxitool can handle these keys in three different ways:

- RSA/ECDSA key stored in a keyfile `*.key` (as plaintext)
- RSA/ECDSA key stored in an encrypted keyfile `*.key`
- RSA/ECDSA key stored on a smartcard

In the latter case, the key will not be read out of the smartcard. A PIN has to be entered via the PIN pad to enable the smartcard to use the private key.

Encrypted RSA keyfiles are protected by a 168-bit Triple-DES key that is derived from a password with the SHA-256 hashing algorithm. Encrypted ECDSA keyfiles are protected by a 256-bit AES key that is derived from a password with the SHA-256 hashing algorithm. In both cases, the password can be changed with the `csadm ChangePassword` command. With the same command a plaintext keyfile can be changed in an encrypted keyfile and vice versa (by omitting the old respectively the new password).



Apart from test environments, we strongly recommend storing private keys on smartcards. Only in this case the private key will never leave the secure token and not be used for calculations on the host.



For all commands that use RSA or ECDSA keys for command authentication, a key specifier is required in the command syntax. A key specifier is either a name of a keyfile (`*.key`) or a smartcard specifier.

Keyfile Specifiers

In case that the private part of the key is needed for the command and is given in an encrypted keyfile, the password of the keyfile has to be given in the command syntax, too. This can be done by appending the password in cleartext (Example 1) directly after the filename separated by a '#' or by using a hidden password entry (Example 2):

Example1:

```
csadm LogonSign=ADMIN,c:\my_keys\myKey.key#sIlEnCe DeleteUser=example user
```

Example 2:

```
csadm LogonSign=ADMIN,c:\my_keys\myKey.key#ask DeleteUser=example user
```

If no password is given (or the password is replaced by the string 'ask'), hidden password entry will be performed for encrypted keyfiles, which we strongly recommend.

Key Specifier Examples

Key Specifier	Description
C:\my_keys\myKey.key	Keyfile

Key Specifier	Description
C:\my_keys\myKey.key#mypassword	Keyfile including a password
C:\my_keys\myKey.key#ask	Keyfile including an interactive password retrieval

Smartcard Specifiers

A smartcard specifier always starts with a colon and consists of three strings separated by colons (for example, `:cs2:cjo:USB0`):

- The first string identifies the type of the smartcard.
- The second string identifies the type of the smartcard reader (PIN pad).
- The last string is the name of the serial device or the USB device the reader is connected to.

Currently, only the smartcards of type TC30 and JavaCard, with the identifier cs2, are supported:

The following types of smartcard readers (PIN pads) are supported:

PIN Pad Identifier	Smartcard reader type (PIN pad types)
cyb	REINER SCT cyberJack (COM) or REINER SCT cyberJack (USB)
cjo	Utimaco cyberJack one
auto	Automatic detection of the PIN pad type


Table 4: Supported PIN pads

The following port types are supported:

Port Identifier	Description
USBn where n = {0, 1, ...}	USB port No. n+1
COMn where n = {0, 1, ...}	COM port No. n+1
USBn[@<port>]@<IP address>[/<password>] [#<smartcard PIN>] where n = {0, 1, ...}	For local PIN pad and remote CHSM Tools/API only

Table 5: Supported ports

In addition to that, a smartcard PIN preceded by a # can be appended. Use this option, if you do not want to enter the PIN at the PIN pad.

 The PIN you enter here is shown in plaintext in your command line. There is no option to hide it as it can be done for example for a password in the csadm `LogonSign` or the csadm `LogonPass` command.

If the PIN pad is used without using the PIN pad daemon, the smartcard PIN may even be used when a PIN is not expected. Otherwise, an error message is shown.

Example: `csadm GetCardInfo=:cs2:cjo:USB0#123456`

Key Specifier Examples

Key Specifier	Description
C:\my_keys\myKey.key	Keyfile
C:\my_keys\myKey.key#mypassword	Keyfile including a password
C:\my_keys\myKey.key#ask	Keyfile including an interactive password retrieval
:cs2:cjo:USBn or :cs2:auto:USBn where n = {0, 1, 2, ...}	Key from a smartcard using a Utimaco cyberJack one connected to a USB port of a Linux computer

Table 6: Examples for key specifiers

If RSA or ECDSA signature authentication with a smartcard is used and if a PIN pad is used at another computer, follow the instructions in *Using a PIN Pad* in the u.trust Anchor - Containerized Hardware Security Module (cHSM) - Administration Manual.

2.6 Password Entry

To authenticate a security-relevant administration command, an operator who uses a password-based authentication mechanism (HMAC password authentication) has to enter his user name and password to the cxitool tool. At this and other opportunities, it is possible to read the password on the monitor which is connected to the computer on which cxitool is running.



To avoid the password being shown as plaintext on the monitor, cxitool offers the possibility for hidden password entry.

For a hidden password entry, specify the 'ask' string instead of the password. Then, cxitool will prompt for the password separately, before starting to process the authentication (and the rest of the command).

Example: `cxitool LogonPass=KeyMgr,ask ListKeys`
Enter Passphrase:

If now the password is entered on the keyboard, it is not shown in clear text on the monitor. The hidden password entry mechanism can also be used to protect the password of an encrypted keyfile.



The usage of hidden password entry is strongly recommended.

2.7 Cryptographic Keys and Key Groups

Cryptographic keys are keys that can be used to encrypt, decrypt, sign or verify data, etc. These keys are stored in the u.trust Anchor cHSM's internal key database, `CXIKEY.db`.

The access of a user to a restricted number of cryptographic keys is realized by using key groups. This is applied in a multi-client environment to avoid that a customer A can use the cryptographic keys of a customer B on the same hardware security module. Key groups are supported for cryptographic keys only. They are not supported for other types of keys, for example, authentication keys, Master Backup Keys (MBKs).

Cryptographic keys are identified by the combination of the following items.

- `<name>`

Key name, e.g., `AES_Key`

If a key name is specified, the minimum key name length is 1 character, the maximum is 256 characters. When using the external keystore, the use of any kind of brackets within the key name, e.g. `() [] {}`, is not supported. All printable ASCII characters except the * and ? wildcards are supported.

If a cryptographic key has been created by using the PKCS#11 CryptoServer Administration Tool (P11CAT) or the PKCS#11 command-line tool p11tool2, the then

assigned `CKA_LABEL` property does not correspond to the Name parameter used in the `cxitool`. All PKCS#11 cryptographic keys have no names in `cxitool`.

- `<group>`



Never use a blank or one or more of the following characters in a key group name: `*`, `?`, `[`, `]`, `{`, `}`, `<`, `>`, `:`, `^`, `"`, `!`, `/`, `\`, `|`. Apart from these characters, all printable ASCII characters are supported for key group names.

When a user is created, he must be assigned to a key group to be able to create or use a key of this key group. If PKCS#11 is used, the key group is a PKCS#11 slot name, e.g., `SLOT_0000` or `SLOT_0001`, etc.

- `<spec>`

Cryptographic key specifier to differentiate between keys if the key name and the key group are identical or empty. This is, for example, useful for PKCS#11 cryptographic keys, because these keys have no names in `cxitool`. It is an integer value. The minimum value is 1, and the maximum value is $2^{31}-1$. If no key specifier is set, the next free bigger value is used as the default value. This specifier must not be confused with the authentication key specifier used, for example, in `cxitool` (see [Authentication Key Specifiers](#)), `csadm` or `p11tool2`, specifying the location of a key, that is, either the path and the filename of a `*.key` file or a smartcard.

If a cryptographic key is specified by a non-empty name, a non-empty group and a specifier and a `cxitool` command shall be performed for exactly this cryptographic key, not only the specifier but also the name and the group must be stated in the command.



Key groups must not be confused with user groups.

A key group is a group of cryptographic keys. Only users who are assigned to a key group may use cryptographic keys of this key group to perform certain actions such as encrypting and decrypting or signing and verifying etc.

A user group is a permission category for performing certain actions, for example, a user with level 2 in the user group 0 (authentication status 00000002) may perform cryptographic commands (encrypt, decrypt, sign, verify etc.), and a user with level 2 in the user group 7 (authentication status 20000000) may perform user management commands such as creating

and deleting users. User groups are described in detail in *User Management, User Groups and Authentication* in the u.trust Anchor - Containerized Hardware Security Module (cHSM) - Administration Manual.



"Cryptographic keys" in CXI are called "objects" in PKCS#11. "Export" and "import" of a cryptographic key in CXI are called "wrap" and "unwrap" of a key in PKCS#11.

The following table shows how a cryptographic key can be assigned to a key group by using the different u.trust Anchortools.

Tool	Description
csadm	The <code>csadm GenKey</code> command only generates keys that are stored on a file system. These keys are no cryptographic keys, they do not belong to any key group and are not stored in the u.trust Anchor cHSM's internal key database, CXIKEY.db. However, these keys can be copied onto smartcards.
cxitool	The key group is defined by the Group parameter when performing a <code>cxitool GenerateKey</code> command
P11CAT	The key group is identical to the slot name that is built by using the selected Slot ID value in the slot list when one of the following buttons is clicked. <ul style="list-style-type: none"> ▪ Object Management > Generate Key > Generate ▪ Object Management > Generate Key Pair > Generate ▪ Object Management > Generate Key from File > Generate Key from File ▪ Object Management > Generate Key Pair from File > Generate Key Pair from File
p11tool2	The key group is defined by the slot name, e.g. , <code>SLOT_00000</code> , that is created from the <code>slot id</code> parameter (default value: 0) when performing the <code>GenerateKey</code> or <code>GenerateKeyPair</code> command. The slot name is identical to the key group.

Table 7: Key group definitions at cryptographic key generation or import

The cryptographic key access for a certain user is restricted on user creation by setting the `CXI_GROUP` attribute of a user. Consider the following rules:

- Only one assignment to `CXI_GROUP` is supported, for example, `CXI_GROUP=1234` . A concatenation of several assignments is not supported, for example, `CXI_GROUP=1234 , CXI_GROUP=5678` .
- Use the '*' and '?' wildcards to enable a user to access multiple key groups.

The following table shows examples for defining cryptographic key access restrictions.

Assignment	Description
	No assignment, that is, the <code>CXI_GROUP</code> attribute has been omitted completely. If there is any key group whose group name is not empty, the user is not permitted to access any cryptographic key that is assigned to this key group.
<code>CXI_GROUP=Test-CA01</code>	The user is allowed to access the <code>Test-CA01</code> key group.
<code>CXI_GROUP=Test-CA02</code>	The user is allowed to access the <code>Test-CA02</code> key group.
<code>CXI_GROUP=Test-CA0?</code>	The user is allowed to access every key group named <code>Test-CA0x</code> , for example, <code>CXI_GROUP=Test-CA01</code> and <code>CXI_GROUP=Test-CA02</code> .
<code>CXI_GROUP=Test-CA*</code>	The user is allowed to access every key group beginning with <code>Test-CA</code> .
<code>CXI_GROUP=*-CA*</code>	The user is allowed to access every key group containing the <code>-CA</code> pattern in the middle.
<code>CXI_GROUP=*</code>	The user is allowed to access all objects regardless of their group properties.

Table 8: Examples for defining cryptographic key access restrictions

- If the user is a PKCS#11 user, the assignment must be built according to the `CXI_GROUP=SLOT_XXXX` pattern with `XXXX` indicating the slot ID.

Example of a user with access to PKCS#11 slot 1: `CXI_GROUP=SLOT_0001`

Users are generated by performing one of the steps described in the next table.

Tool	Description
csadm	<p>The <code>CXI_GROUP=</code> assignment can be put into curly braces <code>{}</code> and appended to the permission mask/authentication status of a <code>csadm AddUser</code> command.</p> <p>Example 1:</p> <pre>csadm Dev= /dev/cs2.0.1 LogonSign=ADMIN,:cs2:cjo:USB0 AddUser=CNGUsr,002{CXI_GROUP=HSM-1},hmacpwd,ask</pre> <p>Example 2:</p> <pre>csadm Dev=/dev/cs2.0.1 LogonSign=ADMIN,:cs2:cjo:USB0 AddUser=KeyUsr,00000002{CXI_GROUP=*},hmacpwd,ask</pre> <p>Example 3 (PKCS#11 key manager in PKCS#11 slot 4 with HMAC password authentication):</p> <pre>csadm LogonSign=ADMIN,:cs2:cjo:USB0 AddUser=KM_0004,020{CXI_GROUP=SLOT_0004},hmacpwd,ask</pre>
cxitool	-

Tool	Description
P11CAT	<p>Only for generating PKCS#11 users:</p> <ul style="list-style-type: none"> Select a Slot ID in the Slot List, log in as Generic and then create a Security Officer (SO), that is, click Slot Management > Init Token > Init Token. The selected slot ID is used for generating a slot name. This slot name is automatically assigned to the Security Officer user as the key group. Select a Slot ID in the Slot List, log in as the Security Officer SO and then create a user, that is, click Slot Management > Init PIN > Init PIN. The selected slot ID is used for generating a slot name. This slot name is automatically assigned to the user as the key group.
p11tool2	<p>Only for generating PKCS#11 users:</p> <ul style="list-style-type: none"> Perform a <code>p11tool2 InitToken</code> command to create a Security Officer (SO). The chosen Slot parameter is automatically assigned to the key group of the Security Officer user. Perform a <code>p11tool2 InitPIN</code> command to create a normal user. The chosen Slot parameter is automatically assigned to the key group of the normal user.

Table 9: Key group assignment at user generation

Verify the assignment of a key group, for example, `CXI_GROUP=1234`, to a user by performing one of the following actions.

Tool	Description
csadm	<code>csadm Dev=... ListUser</code>
cxitool	-
P11CAT	-
p11tool2	-

Table 10: Verifying the key group assignment to a user

The defined value is assigned to the user as the `A[]` attribute, for example, `A[CXI_GROUP=test1]`.

Example output of a `csadm ListUser` command:

```

Name      Permission Mechanism  Attributes
ADMIN     220000000  RSA sign  Z[0]I[0]
KeyMgr    000000020  HMAC passwd Z[0]I[0]A[CXI_GROUP=1234]
KeyMgrAll 000000020  HMAC passwd Z[0]I[0]A[CXI_GROUP=*]
KeyUsr    000000002  HMAC passwd Z[0]I[0]A[CXI_GROUP=1234]
KeyUsrAll 000000002  HMAC passwd I[0]A[CXI_GROUP=*]
SO        000000200  HMAC passwd Z[0]I[0]A[CXI_GROUP=1234]
SOAll     000000200  HMAC passwd Z[0]I[0]A[CXI_GROUP=*]
SO_0000   000000200  HMAC passwd Z[0]I[0]A[CXI_GROUP=SLOT_0000]L[CryptoServer PKCS11 Token ]
USR_0000  000000002  HMAC passwd Z[0]I[0]A[CXI_GROUP=SLOT_0000]
```

```
UsrMgr      200000000    HMAC passwd Z[0]I[0]A[CXI_GROUP=1234]  
UsrMgrAll  200000000    HMAC passwd Z[0]I[0]A[CXI_GROUP=*]
```



Cryptographic keys and configuration data are stored in the u.trust Anchor cHSM's internal key database, `CXIKEY.db`. This database is deleted on alarm occurrence. For details about alarms, see *Alarms* in the u.trust Anchor - Containerized Hardware Security Module (cHSM) - Administration Manual.

We highly recommend creating a backup of the `CXIKEY.db` file with the `csadm BackupDatabase` command, see *BackupDatabase* in the u.trust Anchor - csadm Manual or using the `cxitool BackupKey` command, so you can restore the cryptographic keys and the configuration data. In FIPS mode, csadm cannot be used for backup purposes. In this case, only the `cxitool BackupKey` command backs up cryptographic keys and the configuration data.

We also highly recommend ensuring that MBK (Master Backup Key) shares are available when backing up and restoring the `CXIKEY.db` database. These shares are necessary to make the restored database usable again. Otherwise, all data in this database remains inaccessible. For details about MBKs, see *Commands for Managing the Master Backup Keys* in the u.trust Anchor - csadm Manual and *Master Backup Key* in the u.trust Anchor - Containerized Hardware Security Module (cHSM) - Administration Manual.

3 Commands and Parameters

The following sections describe the cxitool commands. Special parameters are described in [Special Parameters](#). These are also described as part of the commands they are used for.



Certain types of shell processes treat certain characters (for example, commas, colons, semi-colons) differently. If the execution of a cxitool command fails with an error message from the shell about missing parameter (`algo` , `size`) or illegal parameter format, quoting parameter values may be necessary. For example, a correct command entry in the Microsoft PowerShell:

```
cxitool Export=allow Name=RSA2048 GenerateKey="RSA,2048"
```

3.1 Basic Commands

3.1.1 Help

If called without any parameter, this command shows a list of all available cxitool parameters and commands. If called with a parameter, this command gives details about this command.

Syntax	<pre>cxitool Help or cxitool cxitool Help=<parameter> cxitool Help=<command></pre>
---------------	--

Parameter	Description
<parameter>	Specific cxitool parameter
<command>	Specific cxitool command to display help for

Example	<p>Example 1</p> <pre>cxitool Help</pre> <p>This command shows an overview of all parameters and commands.</p> <p>Example 2</p> <pre>cxitool Help=GenerateKey</pre> <p>This command shows the help of the <code>cxitool GenerateKey</code> command including the parameters offered or needed for this command, e.g., Name, Group, Spec.</p>
----------------	--

Output	Upon successful execution, the command returns a list of all cxitool commands, or specific information on command and parameters if a cxitool command was specified.
---------------	--

3.1.2 Version

This command shows the version numbers of the cxitool.

Syntax	cxitool Version
Parameter	Description
No parameters are given for input.	
Example	cxitool Version
Output	Upon successful execution, the command returns the version of the currently installed cxitool. cxitool 1.6.0 [win-x64] (May 9 2018) cxi-API 1.7.4.0

3.1.3 Authentication Commands

Most cxitool commands must be authenticated by a user, see *Permissions and Authentication Status* and *Secure Messaging* in the u.trust Anchor - Containerized Hardware Security Module (cHSM) - Administration Manual. For details about user groups, see *User Management, User Groups and Authentication* in the u.trust Anchor - Containerized Hardware Security Module (cHSM) - Administration Manual.

The u.trust Anchor cHSM provides a variety of command authentication mechanisms. For details about the authentication concept, see *Authentication Mechanisms* in the u.trust Anchor - Containerized Hardware Security Module (cHSM) - Administration Manual.

Most predefined security-relevant commands require the authentication level 2 in a specific user group. This enables you to enforce a two-person rule by creating users with permission level 1 in the respective user group, this means that two users of the specific user group are necessary to authenticate the command. For this reason, the specific command requires authentication according to the two-person rule, i.e., authentication by two independent users is required for the command to be executed. However, the predefined ADMIN user has permission 2 in the user groups 7 and 6, that is, for the user management and for the system management commands).



If a command requires multi-user authentication according to the n-person rule ($n=1\dots16$), all n users have to apply their authentication prior to command execution:

```
cxitool <Authentication_1> <Authentication_2> ... <Authentication_n>
<command>
```

The users may even use different authentication mechanisms.

Example: `cxitool LogonSign=KeyUsr01,:cs2:cjo:USB0
LogonPass=KeyUsr02,ask Group=1234 ListKeys`

The authentication concept of the device contains eight different user groups (0 to 7) that each belongs to a list of available commands. Each command has a permission level. Every user is assigned a permission level as well, which determines if they are allowed to execute commands on their own or need other users for authentication. Each authentication level can vary from 0 (no permission/authentication) to 15 (highest level of permission/authentication).

Example

Value of authentication state	2	0	0	1	0	3	0	1
Authentication Level for User Group	7	6	5	4	3	2	1	0

In this example, the device has reached authentication level 2 in user group 7, level 0 in user group 6, (...), and authentication level 1 in user group 0.

After a user has successfully authenticated towards the device, the authentication state will be augmented by the permissions of the user:

Example

Initial authentication status (no user logged in)	00000000
Permissions of the Admin1	21000000
Permissions of the Admin2	33000000
Permissions of the Admin3	51000000
Permissions of the Admin4	51000000
Permissions of the User	00000002
Authentication status: Admin1, Admin2, Admin3, Admin4 and User logged in	F6000002

Consider that the authentication statuses can be added without any further restriction as described above only if the involved users are logged in to perform a command

- of a firmware module that is not the CXI firmware module or
- of the CXI firmware module and the cryptographic key (CXI key) that is used to perform this command does not belong to any key group.

If however, the used cryptographic key in the CXI firmware module is assigned to a key group, only the authentication statuses of those logged in users can be summed up that belong to the same key group.

Example:

userA has authentication status 0x00000001 and is assigned to the key group A.

userB has authentication status 0x00000001 and is assigned to the key group B.

Three cryptographic keys are created. keyA is assigned to key group A, keyB to key group B and keyAB to key group AB.

If userA and userB are simultaneously logged in, only those commands of the CXI firmware module can be performed that only need

- an authentication status 0x00000001 for using keyA or
- an authentication status 0x00000001 for using keyB.

If a command needs

- an authentication status 0x00000001 or higher for using keyAB or
- an authentication status 0x00000002 or higher for using keyA or
- an authentication status 0x00000002 or higher for using keyB,

it cannot be performed.



The authentication commands described in [LogonSign](#) and [LogonPass](#) do not only perform command authentication, but also provide a Secure Messaging session for the protection of the confidentiality of the command data.

In the following subsections, the syntax of all authentication mechanisms is explained. These authentication commands (leading to the necessary authentication status) can be inserted for the placeholder `<auth>`, which is given in the syntax of all security-relevant commands described below.



Certain types of shell processes treat certain characters (commas, colons, semi-colons) differently. If the execution of a cxitool command fails with an error message from the shell about an illegal parameter format, quoting parameter values may be necessary.

3.1.3.1 LogonPass

With this command, a user with a password-based authentication mechanism (i.e., authentication mechanism HMAC Password) opens an authenticated Secure Messaging session for the given command. The session key (32-byte AES key) is generated using the Diffie-Hellman key agreement.



The authenticated Secure Messaging session is automatically closed when cxitool is closed.

A user uses an HMAC password-based key-derived function (HMAC-PBKDF) for authentication. The HMAC-PBKDF according to NIST SP 800-132 does not use the HMAC password itself for authentication but a function derived from the HMAC password. For HMAC-PBKDF, 1000 iterations are used here. This number of iterations, as used for the key derivation from a given password, is fixed and not configurable. HMAC-PBKDF is only applied if on both sides, the host side and the firmware side, HMAC-PBKDF is applied. If it is not available on one of these sides, the legacy version of the HMAC password-based mechanism is applied, whereby the user's password is used directly as an HMAC key.

In FIPS mode, using HMAC-PBKDF is mandatory.

Syntax	<code>cxitool [Dev=<device>] LogonPass=<user>,<password> [<commands>]</code>
Parameter	Description
<code><device></code>	Device address This parameter can be omitted if the device address has been set as the CRYPTOSERVER environment variable.

Parameter	Description
<user>	Name of the user to be logged in
<password>	User password <ul style="list-style-type: none">User password entry in plaintext. We do not recommend this.ask Entering the password is requested (Enter Passphrase:) at the execution time of the <code>LogonPass</code> command. The entered password is not shown in the command line. This procedure is recommended.
<commands>	The commands to be executed in the Secure Messaging session. The session will be closed afterwards.

Example	Example 1: <code>cxitool Dev=/dev/cs2.0.1 LogonPass=KeyMgr,123456 Group=1234 Name=RSA2048 GenerateKey=RSA,2048</code> Example 2: <code>cxitool Dev=/dev/cs2.0.1 LogonPass=KeyMgr,ask Group=1234 Name=RSA2048 GenerateKey=RSA,2048</code>
---------	---

Output	Upon successful execution, the command returns no specific output and executes the command specified within the secure messaging session.
--------	---

Once a user with HMAC password authentication (or more precise: HMAC password-based key-derived function (HMAC-PBKDF) for authentication) has been created, this user cannot perform commands this user needs an authentication for, except for the `csadm ChangeUser` command. The `csadm ListUser` command shows the `I[1]` attribute value for this user. To enable this user to perform commands he/she needs an authentication for, he/she must change the credentials by performing the `csadm ChangeUser` command. It is important that the changes are performed by the user himself/herself and not, for example, by an administrator. Then the `csadm ListUser` command shows the `I[0]` attribute value for this user.

3.1.3.2 LogonSign

With this command, a user with a signature-based authentication mechanism (i.e., authentication mechanism RSA Signature or ECDSA Signature) opens an authenticated Secure Messaging session for the given command. The session key (32-byte AES key) is generated using the Diffie-Hellman key agreement according to PKCS#3.

If the private part of the user's authentication key is stored on a smartcard, the user will be prompted at the PIN pad to insert his smartcard and enter the PIN.

For RSA or ECDSA signature authentication with a smartcard, the PIN pad has to be connected to the computer where the cxitool is running (USB port) or to another computer, see *Using a PIN Pad* in the u.trust Anchor - Containerized Hardware Security Module (cHSM) - Administration Manual.



The authenticated Secure Messaging session is automatically closed when cxitool is closed.

Syntax

```
cxitool [Dev=<device>] LogonSign=<user>,< keyspec > [<commands>]
```

Parameter	Description
<device>	Device address of the u.trust Anchor This parameter can be omitted if the device address has been set as the CRYPTOSERVER environment variable.
<user>	Name of the user to be logged in
<keyspec>	<p>Location where the private part of the user's authentication key should be loaded from (specifier of smartcard or keyfile, see Authentication Key Specifiers)</p> <ul style="list-style-type: none"> Smartcard specifier: for example, <code>:cs2:cjo:USB0</code> For RSA and ECDSA signature authentication, the following applies: If you append # and a PIN to the smartcard specifier, the user is automatically logged in without any user interaction at the PIN pad. Example: <code>:cs2:cjo:USB0#123456</code> For RSA smartcard authentication, the following applies: <ul style="list-style-type: none"> The smartcard specifier can be omitted. The comma after the user name can be omitted as well (recommended). For an automatic login, replace the smartcard specifier by #<PIN>. Storage location and name of the keyfile (<code>*.key</code>) In case of an encrypted keyfile: <code>*.key[#<password>]</code> where <code><password></code> is the password used for protecting the keyfile. It might be entered: <ul style="list-style-type: none"> in clear text, for example, <code>MyKey.key#mypwd</code> or as string ask for hidden password entry, for example, <code>MyKey.key#ask</code>.
<commands>	The commands to be executed in the Secure Messaging session. The session will be closed afterwards.

Example	<p>Example 1:</p> <pre>cxitool Dev=PCI:0.1 LogonSign=KeyMgr,:cs2:cjo:USB0 Group=1234 Name=RSA2048 GenerateKey=RSA,2048</pre> <p>The following alternatives are possible if <code>KeyMgr</code> uses RSA smartcard authentication:</p> <p>Shorter (no smartcard specifier) but with the same effect:</p> <pre>cxitool Dev=PCI:0.1 LogonSign=KeyMgr, Group=1234 Name=RSA2048 GenerateKey=RSA,2048</pre> <p>Also possible (no comma after the user name):</p> <pre>cxitool Dev=PCI:0.1 LogonSign=KeyMgr Group=1234 Name=RSA2048 GenerateKey=RSA,2048</pre> <p>With automatic login using the PIN 123456:</p> <pre>cxitool Dev=PCI:0.1 LogonSign=KeyMgr,#123456 Group=1234 Name=RSA2048 GenerateKey=RSA,2048</pre> <p>Example 2:</p> <pre>cxitool Dev=PCI:0.1 LogonSign=KeyMgr,C: \Utimaco\utruster\KeyMgr.key Name=RSA2048 GenerateKey=RSA,2048</pre> <p>Example 3:</p> <pre>cxitool Dev=4001@127.0.0.1 LogonSign=KeyMgr,E: \myKeys\myRSA.key#ask Group=1234 Name=RSA2048 GenerateKey=RSA,2048</pre>
Output	<p>Upon successful execution, the command returns no specific output and executes the command specified within the secure messaging session.</p>

3.2 Special Parameters

3.2.1 Export

This Export parameter defines whether a cryptographic key can be exported or not. It can have the `DENY` (`0x00000000`) value, the `allow` (`0x00000001`) value, or the `allow_plain` (`0x00000002`) value. It is an optional parameter for the `cxitool GenerateKey` command and the `cxitool ImportP12` command. In these commands, the cryptographic key is specified by the combination of Name, Group and Spec.

Once the cryptographic key exists, the export property of the cryptographic key can be shown by performing the `cxitool KeyInfo` command. The `cxitool KeyInfo` command shows the numerical representations `0x00000000`, `0x00000001` and `0x00000002` of `deny`, `allow` and `allow_plain`. In addition to that, the `cxitool KeyInfo` command can show the `0x00010000` (`deny_backup`) value. This value cannot be set by performing the `cxitool Export` parameter. It can only be set by the CXI API, see also [KeyInfo](#).



Exporting a cryptographic key may be restricted by the `AuthPlain` parameter in the global configuration or the key group configuration, see [SetConfig](#) for details.

The `deny` value, the `allow` value, the `allow_plain` value, and the `deny_backup` value are the CXI representation of the export property. This representation corresponds to the PKCS#11 representation, which uses the `Extractable` parameter and the `Sensitive` parameter. Both representations are equivalent to each other. The `cxitool Export` parameter only uses the CXI representation and the `cxitool KeyInfo` command shows both representations.

<i>CXI representation</i>	<i>PKCS#11 representation</i>
<code>Export=allow_plain</code>	<code>Extractable = 1 and Sensitive = 0</code>
<code>Export=allow</code>	<code>Extractable = 1 and Sensitive = 1</code>
<code>Export=deny</code>	<code>Extractable = 0 and Sensitive = 0 or 1</code>

Table 11: Export representations in CXI and PKCS#11

Syntax	<code>cxitool Export=[deny allow allow_plain]</code>
---------------	--

<i>Parameter</i>	<i>Description</i>
<code><deny></code>	Exporting the key is not allowed (Default, if <code>Export</code> is not used), corresponds to <code>0x00000000</code> .
<code><allow></code>	Exporting the key is allowed in encrypted form, corresponds to <code>0x00000001</code> .
<code><allow_plain></code>	Exporting the key in plaintext is allowed, corresponds to <code>0x00000002</code> .

Example	<code>cxitool Dev=/dev/cs2.0.1 LogonPass=KeyMgr,123456 Name=RSA2048 Group=1234 Export=allow GenerateKey=RSA,2048</code>
----------------	---

Output	Upon successful execution of the command, no output is given.
---------------	---

3.2.2 Usage

The `Usage` parameter sets the cryptographic functions the cryptographic key can be used for, for example, an RSA key may be used for signing but not for encryption. The cryptographic

key is specified by the combination of Name, Group and Spec of the `cxitool` command the `Usage` parameter has been added to.



Once set, the `Usage` parameter cannot be changed.



`Usage`, a `cxitool` proprietary parameter of the `cxitool GenerateKey` command, must not be confused with the `KeyUsage` parameter, an X.509 parameter of the `cxitool SelfSignedCert` command. `Usage` defines the scope of application of a key, and `KeyUsage` defines the scope of application of the corresponding certificate. For details about the `KeyUsage` parameter, see [SelfSignedCert](#).



The `Usage` parameter must not be confused with the `cxitool SetFipsUsage` command.

If the u.trust Anchor cHSM is in FIPS mode, the `cxitool SetFipsUsage` command must be performed in addition to the `Usage` parameter before a cryptographic key can perform any cryptographic operation. The FIPS usage set by the `cxitool SetFipsUsage` command must correspond to the value of the `Usage` parameter. For details, see [SetFipsUsage](#).

If the u.trust Anchor cHSM is not in FIPS mode, the `cxitool SetFipsUsage` command is irrelevant.

Syntax	<code>cxitool Usage=[ENCRYPT DECRYPT SIGN VERIFY WRAP UNWRAP DERIVE]</code>
---------------	---

Parameter	Description
<ENCRYPT>	Encryption
<DECRYPT>	Decryption
<SIGN>	Signature generation
<VERIFY>	Signature verification

Parameter	Description
<WRAP>	Wrapping (exporting) a key
<UNWRAP>	Unwrapping (importing) a key
<DERIVE>	Key derivation Key derivation cannot be done by using cxitool.



Multiple usage values are separated by commas.

Example	<p>Example 1: Single usage value</p> <pre>cxitool Dev=/dev/cs2.0.1 LogonPass=KeyMgr,123456 Name=RSA2048 Group=1234 Usage=ENCRYPT GenerateKey=RSA,2048</pre> <p>Example 2: Multiple usage values</p> <pre>cxitool Dev=/dev/cs2.0.1 LogonPass=KeyMgr,123456 Name=RSA2048 Group=1234 Usage=ENCRYPT,DECRYPT GenerateKey=RSA,2048</pre>
----------------	--

Output	Upon successful execution of the command, no output is given.
---------------	---

The usage property of the cryptographic key can be shown by performing the `cxitool KeyInfo` command. This command shows, among others, a summarized bit field as a CXI representation of the usage but as well a PKCS#11 representation of the usage.

Definition	CXI representation	PKCS#11 representation	Description
Decrypt	0x00000001	Decrypt: 1	The key may be used for decryption. If you want to decrypt an encrypted key, you have to use "Unwrap" instead.
Sign	0x00000002	Sign: 1	The key may be used for signature creation.
Derive	0x00000004	Derive: 1	The key may be used for key derivation.
Wrap	0x00000008	Wrap: 1	The key may be used to encrypt/wrap other keys.
Encrypt	0x00000010	Encrypt: 1	The key may be used for encryption. If you want to encrypt a key, you have to use "Wrap" instead.
Verify	0x00000020	Verify: 1	The key may be used for signature verification.
Unwrap	0x00000080	Unwrap: 1	The key may be used to decrypt/unwrap other keys.

Definition	CXI representation	PKCS#11 representation	Description
Verify recover	0x00000200	Verify_Rec.: 1	The key may be used for signature verification and recovering the message from the signature. This parameter is irrelevant because it is ignored by cxitool. It cannot be set by using the Usage parameter.

Table 12: Usage representations in CXI and PKCS#11 representation

In case of multiple usages, the values are added. Example: A value of `0x0000000B` = `0x00000001` + `0x00000002` + `0x00000008` means that the cryptographic key can be used for decryption, signing and wrapping.

If a cryptographic key is generated by the `cxitool GenerateKey` command without any Usage parameter, the following default usage values are set:

Algorithm	Usage	
	CXI representation	PKCS#11 representation
DES	0x000000BB	Encrypt: 1, Decrypt: 1, Sign: 1, Verify: 1, Verify_Rec.: 0, Wrap: 1, Unwrap: 1, and Derive: 0
AES		
RSA	0x000002BB	Encrypt: 1, Decrypt: 1, Sign: 1, Verify: 1, Verify_Rec.: 1, Wrap: 1, Unwrap: 1, and Derive: 0
DSA	0x00000222	Encrypt: 0, Decrypt: 0, Sign: 1, Verify: 1, Verify_Rec.: 1, Wrap: 0, Unwrap: 0, and Derive: 0
EC	0x000002BF	Encrypt: 1, Decrypt: 1, Sign: 1, Verify: 1, Verify_Rec.: 1, Wrap: 1, Unwrap: 1, and Derive: 1

Table 13: Default usage values in CXI and PKCS#11 representation

3.2.3 Timeout

This parameter sets the time cxitool waits for u.trust Anchor cHSM responses. It applies to all commands. However, it is not shown in the documentation of these commands.

Syntax	<code>cxitool Timeout=<time></code>
---------------	---

Parameter	Description
<time>	Duration in milliseconds Default value: 3 minutes In practice, the timeout can reach approximately twice the timeout value specified in the <code>cxitool timeout</code> command.
Example	<code>cxitool Timeout=10000</code>
Output	Upon successful execution of the command, no output is given.

3.3 CXI Firmware Configuration

3.3.1 GetConfig

This command retrieves the values of the global configuration or a group-specific (local) configuration. If no group is specified, the values of the global configuration object are retrieved. If a global configuration object has been created using the `cxitool SetConfig` command, the `cxitool GetConfig` command retrieves the global configuration values from this object. Otherwise, the default values are retrieved. If a group-specific configuration object has been created, the `cxitool GetConfig` command retrieves the group-specific configuration values from this object. Otherwise, the default values are retrieved.

Because configuration objects are stored in the u.trust Anchor cHSM's internal key database, `CXIKEY.db`, configuration objects are handled as keys in this database. Therefore, a configuration object can be listed in the output of a `cxitool ListKeys` command.

For PKCS#11, the `cxitool GetConfig` command is similar to the `p11tool2 ListConfig`, `p11tool2 GetGlobalConfig`, `p11tool2 GetLocalConfig` and the `p11tool2 GetSlotConfig` command. For details about the p11tool2 commands, see the u.trust Anchor - PKCS#11 p11tool2 - Reference Manual. Another alternative is the PKCS#11 CryptoServer Administration Tool (P11CAT). Open **Config Management > Global Configuration** or **Config Management > Slot Configuration** to get the configuration parameters, see the CryptoServer – PKCS#11 P11CAT Manual.

Syntax	<code>cxitool [Dev=<dev>] <auth> [Group=<group>] GetConfig</code>
---------------	---

Authentication	<p>This command must be authenticated by at least one of the following user roles:</p> <ul style="list-style-type: none">▪ User manager, level 2 in user group 7, permission mask 20000000▪ Security officer, SO, level 2 in user group 2, permission mask 00000200▪ Key manager; either level 2 in user group 1, permission mask 00000020 or level 2 in user group 0, permission mask 00000002 (default) The permission mask that is applied to the key manager of a key group is defined in the configuration object of this key group. The permission mask can be set by performing the <code>cxitool ... SetConfig=AuthKeym,...</code> command▪ Key user, level 2 in user group 0, permission mask 00000002 <p>If the Group parameter is not used, the global configuration is shown. In this case, it is irrelevant to which key group the user has been assigned. If the Group parameter specifies a specific key group and if the user has been assigned to this key group (<code>CXI_GROUP=<group> property</code>), the key group's configuration is shown.</p>
-----------------------	---

Parameter	Description
<dev>	Device address. This parameter can be omitted if the device address has been set as the CRYPTOSERVER environment variable.
<group>	<p>The configuration of this key group is retrieved. If no key group is specified, the global configuration is retrieved. The * and ? wildcards are not supported.</p> <p>A <code>cxitool ... SetConfig=AllowGroups,false</code> command causes the key group configuration to be ignored but not to be overwritten. Then, a <code>cxitool ... Group=<group name> GetConfig</code> command shows only global configuration values (except for the <code>SecureGroupPassHash</code> parameter as always). If then a <code>cxitool ... SetConfig=AllowGroups,true</code> command and a successive <code>cxitool ... Group=<group name> GetConfig</code> command are performed, the previous group-specific configuration is shown again.</p> <p>If PKCS#11 is used, the key group is a PKCS#11 slot name, for example, <code>SLOT_0000</code> or <code>SLOT_0001</code>.</p>

Example	<p>Configuration of the 1234 group:</p> <pre>cxitool Dev=/dev/cs2.0.1 LogonPass=KeyUsr,123456 Group=1234 GetConfig</pre>
----------------	--

Output

Upon successful execution, the parameters of the defined configuration are returned. SecureGroupPassHash is only shown if Group has been set. The SecureGroupPassHash parameter is set by performing the `cxitool SecureGroupPass` command. All other parameters are set by performing the `cxitool SetConfig` command.

```

90 AllowGroups           : true
91 CheckValidityPeriod   : false
92 AuthPlain             : 0x00000002
93 WrapPolicy            : false
94 AuthKeym              : 0x00000002
96 SecureDerivation      : false
97 SecureImport          : false
98 SecureRSAComponents   : true
99 P11R3BackwardsCompatible: false
100 EnforceBlinding       : false
101 SecureGroupBackup     : false
107 EnforceExtKeys        : false
108 AllowWeakDesKeys      : false
102 SecureGroupPassHash   : not set

```

3.3.2 SetConfig

This command sets a configuration value of the global configuration object or a group-specific (local) configuration object. A group-specific setting overrides the corresponding global setting for the specified group. If no group is specified, the values of the global configuration object are set.



Cryptographic keys and configuration data are stored in the u.trust Anchor cHSM's internal key database, `CXIKEY.db`. This database is deleted on alarm occurrence. For details about alarms, see *Alarms* in the u.trust Anchor - Containerized Hardware Security Module (cHSM) - Administration Manual.

We highly recommend creating a backup of the `CXIKEY.db` file with the `csadm BackupDatabase` command, see *BackupDatabase* in the u.trust Anchor - csadm Manual or using the `cxitool BackupKey` command, so you can restore the cryptographic keys and the configuration data. In FIPS mode, `csadm` and `CAT` cannot be used for backup purposes. In this case, only the `cxitool BackupKey` command backs up cryptographic keys and the configuration data.

We also highly recommend ensuring that MBK (Master Backup Key) shares are available when backing up and restoring the `CXIKEY.db` database. These shares are necessary to make the restored database usable again. Otherwise, all data in this database remains inaccessible. For details about MBKs, see *Commands for Managing the Master Backup Keys* in the u.trust Anchor - csadm Manual and *Master Backup Key* in the u.trust Anchor - Containerized Hardware Security Module (cHSM) - Administration Manual.

For PKCS#11, the `cxitool SetConfig` command is similar to the `p11tool2 SetGlobalConfig` and the `p11tool2 SetSlotConfig` command. For details about the `p11tool2` commands, see the [u.trust Anchor - PKCS#11 p11tool2 - Reference Manual](#). Another alternative is the PKCS#11 CryptoServer Administration Tool (P11CAT). Open **Config Management > Global Configuration** or **Config Management > Slot Configuration** to set the configuration parameters, see the [CryptoServer – PKCS#11 P11CAT Manual](#).

The execution of the `cxitool SetConfig` command has an immediate effect. No restart of the device is needed.


Syntax	<code>cxitool [Dev=<dev>] <auth> [Group=<group>] SetConfig=<key>,<value></code>
---------------	---


Authentication	<p>The * and ? wildcards are not supported.</p> <ul style="list-style-type: none"> Global configuration If the Group parameter is not used, the global configuration is set, that is, this setting is set for all key groups. This command must be authenticated by at least a user manager (level 2 in user group 7, permission mask 20000000). The user manager does not have to be assigned to a key group. Group configuration Prerequisite: The global <code>AllowGroups</code> property must have been set to true. A group configuration can be done only by a security officer (SO, level 2 in user group 2, permission mask 00000200) and this security officer must have been assigned to the group he wants to change.
-----------------------	---

Parameter	Description
<dev>	Device address. This parameter can be omitted if the device address has been set as the CRYPTOSERVER environment variable.

Parameter	Description
<group>	<p>Key group for which a configuration object shall be set. The * and ? wildcards are not supported.</p> <ul style="list-style-type: none">▪ Global configuration If the Group parameter is not used, the global configuration is set, that is, this setting is set for all key groups. This command must be authenticated by at least a user manager (level 2 in user group 7, permission mask 20000000). The user manager does not have to be assigned to a key group.▪ Group configuration Prerequisite: The global AllowGroups property must have been set to true. A group configuration can be done only by a security officer (SO, level 2 in user group 2, permission mask 00000200) and this security officer must have been assigned to the group he wants to change.

Parameter	Description
<key>	<p>The parameter to be set to <value> . The <code>AllowGroups</code> parameter can be used only for the global configuration.</p> <p>Instead of using the named values for <key> , their numerical representations can be used as well.</p> <pre> AllowGroups: 90 CheckValidityPeriod: 91 AuthPlain: 92 WrapPolicy: 93 AuthKeym: 94 SecureDerivation: 96 SecureImport: 97 SecureRSAComponents: 98 P11R3BackwardsCompatible: 99 EnforceBlinding: 100 SecureGroupBackup: 101 EnforceExtKeys: 107 AllowWeakDesKeys: 108 </pre> <p>Valid keys are:</p> <ul style="list-style-type: none"> ▪ <code>AllowGroups</code> If <code>true</code> , a key group configuration is allowed to be changed by a security officer (level 2 in user group 2, permission mask 00000200) (default: <code>false</code>) . If <code>false</code> , only the global configuration is used and any key group configuration is ignored except for the <code>SecureGroupPassHash</code> . A <code>cxitool ... SetConfig=AllowGroups,false</code> command causes the key group configuration to be ignored but not to be overwritten. If then a <code>cxitool ... SetConfig=AllowGroups,true</code> command and a successive <code>cxitool ... Group=<group name> GetConfig</code> command are performed, the previous key group-specific configuration is shown again. Even if the <code>AllowGroups</code> parameter is <code>false</code> , the key group-specific <code>cxitool SecureGroupPass</code> command or a key group-specific <code>cxitool ResetConfig</code> command may be performed. The <code>AllowGroups</code> parameter may only be set by a user manager with level 2 in user group 7 (permission mask 20000000). The <code>AllowGroups</code> parameter is only supported for the global configuration. To set it, the <code>Group</code> parameter must be omitted. The <code>Group=*</code> assignment is not supported here.

Parameter	Description
	<ul style="list-style-type: none"> ▪ CheckValidityPeriod If <code>true</code>, the start and end date of the validity period is verified before a key is used (default: <code>false</code>). If the key's property list (<code>cxitool KeyInfo</code> command) does not contain a start date or an end date, the beginning or ending of the validity period is not verified regardless of this configuration property. ▪ AuthPlain Required minimum permission mask needed to export or import a key in plaintext (default: <code>0x00000002</code>). For example, <code>0x00000001</code> is supported but <code>0x00000000</code> is not supported. To avoid an inappropriate value, ensure not to forget the leading <code>0x</code>. <p>Do not use AuthPlain because plaintext import and plaintext export of cryptographic keys are not allowed.</p> <ul style="list-style-type: none"> ▪ WrapPolicy If <code>true</code>, the algorithm strength of the wrapping (exporting) key is verified and has to be greater than or equal to the strength of the wrapped (exported) key. For example, a 256-bit AES key cannot be encrypted with a 1024-bit RSA key. This ensures that a key can be securely exported outside the u.trust Anchor CHSM. If <code>false</code>, the strength of the wrapping key is not verified (default: <code>false</code>). ▪ AuthKeym Minimum required permission mask of the key manager. <hr/> <div>  <p>If you generate a user to perform cryptographic operations (encrypt, decrypt, hash, sign and verify), this user can by default manage cryptographic keys (generate, delete, back up, restore etc. keys) as well. In this case, a key user always is as well a key manager. This means as well that both a key user and a key manager have the same minimum permission mask <code>00000002</code>. This case is indicated by an AuthKeym value of <code>0x00000002</code> (default). If you want, for example, due to security reasons, that a user always either is a key user or a key manager, set AuthKeym to a value of <code>0x00000020</code>. Then, the minimum permission mask for a key user is <code>00000002</code>, and the minimum permission mask for a key manager is <code>00000020</code>. Ensure not to forget the leading <code>0x</code>. Other values are not supported.</p> <p>If you set the value to <code>0x00000020</code>, ensure that a key manager with this permission mask has been created. <i>AddUser</i> in the u.trust Anchor – csadm Manual gives an example of creating a key manager by using the csadm command line CryptoServer Administration Tool.</p> </div> <hr/>

Parameter	Description
	<div> If a user with the permission mask 00000020 performs a cxitool command, ensure that AuthKeym of the key group that is applied in this command has been set to 0x00000020. Otherwise, the permission denied error message is shown.</div> <div><ul style="list-style-type: none">SecureDerivation<p>If this parameter is <code>true</code>, the <code>DeriveKey()</code> function of the CXI firmware module cannot use weak derivation mechanisms (default: <code>false</code>).</p><p>This parameter prohibits the use of the following key derivation mechanisms, and prevents reduced key space attacks:</p><ul style="list-style-type: none">CKM_XOR_BASE_AND_DATACKM_CONCATENATE_DATA_AND_BASECKM_CONCATENATE_BASE_AND_DATACKM_CONCATENATE_BASE_AND_KEYCKM_EXTRACT_KEY_FROM_KEY<p>See PKCS11CMS for a detailed description of the mechanisms.</p>SecureImport<p>If this parameter is <code>true</code> (default: <code>false</code>), it prevents simple key extraction attacks by performing additional strict verifications on wrapping (exporting) keys. For more details about the additional verifications, see <i>Global CryptoServer Configuration Object</i> in the CryptoServer - PKCS#11 R3 - Developer Guide provided in the product bundle under <code>\Documentation\Crypto_APIs\PKCS11_R3\</code>.</p>SecureRSAComponents<p>If this parameter is <code>true</code>, new RSA keys with public exponents below 0x10001 cannot be created or imported (default: <code>true</code>).</p>P11R3BackwardsCompatible<p>If this parameter is <code>true</code> (default: <code>false</code>), keys generated by using an EC scheme or Diffie-Hellman algorithm can be used as base keys for key derivation (PKCS#11 standard non-compliant legacy). This may be necessary for some integrations.</p></div>

Parameter	Description
	<ul style="list-style-type: none"> ▪ <code>EnforceBlinding</code> If this parameter is <code>false</code> (default value), measures against side-channel analysis (SCA) attacks are seized by the physical construction (power supply and housing). In addition to that, timing analysis measurements are prevented. If this parameter is <code>true</code> (default: <code>false</code>), side-channel analysis attacks are prevented by an additional protection in the mathematical layer by computing with blinded values (keys and plaintext). However, the measures for SCA resistance negatively affect the performance of the cryptographic operations on RSA and ECDSA keys. Therefore, they are disabled by default, and can be enabled, if necessary. ▪ <code>SecureGroupBackup</code> <code>SecureGroupBackup</code> is irrelevant because Tenant Backup Keys (TBKs) are not supported. ▪ <code>EnforceExtKeys</code> If <code>EnforceExtKeys</code> (default: <code>false</code>) is set to <code>true</code>, an object (for example, a cryptographic key) is always created in the external keystore. A cryptographic key is created by the following actions: generate a key, generate a key pair, derive a key, restore a key, import a key and unwrap a key. <code>EnforceExtKeys</code> is irrelevant for key usage functions. For example, signing with an already existing internal key is supported even if <code>EnforceExtKeys</code> is set to <code>true</code>. If <code>EnforceExtKeys</code> is set to <code>true</code>, the <code>KeysExternal</code> parameter in the <code>cfg</code> file must be set to <code>true</code> and the <code>KeyStore</code> parameter in the same file must be set as well. For details, see <i>Editing the <code>cs_pkcs11_R3.cfg</code> Configuration File</i> in the <i>CryptoServer – PKCS#11 P11CAT Manual</i>. If <code>EnforceExtKeys</code> is set to <code>true</code> and <code>KeysExternal</code> is set to <code>false</code>, no cryptographic key but an error message is generated. ▪ <code>AllowWeakDesKeys</code> If <code>AllowWeakDesKeys</code> (default: <code>false</code>) is set to <code>true</code>, importing weak DES keys is supported. <code>AllowWeakDesKeys</code> might be set for a key group (slot) or for the entire u.trust Anchor CHSM. A weak DES key is a 2DES key that has two equal parts or a 3DES key that has two parts that are pairwise equal. ▪ The <code>SecureGroupPassHash</code> parameter cannot be set by performing the <code>cxitool SetConfig</code> command but by performing the <code>cxitool SecureGroupPass</code> command, see below. The result can be retrieved by the <code>cxitool GetConfig</code> command.
<value>	The value the <code><key></code> parameter is set to. It is either <code>true</code> or <code>false</code> except for the <code>AuthPlain</code> and <code>AuthKeym</code> parameters that are set to a permission mask, an 8-digit hexadecimal number, for example, <code>0x00000002</code> . Ensure not to forget the leading <code>0x</code> .

Example	<p>Example 1: Initiating the usage of a Tenant Backup Key for all groups</p> <pre>cxitool Dev=/dev/cs2.0.1 LogonSign=ADMIN,C: \Utimaco\CryptoServer\ADMIN.key SetConfig=SecureGroupBackup,true</pre> <p>The same example but with the numerical representation of SecureGroupBackup:</p> <pre>cxitool Dev=/dev/cs2.0.1 LogonSign=ADMIN,C: \Utimaco\CryptoServer\ADMIN.key SetConfig=101,true</pre> <p>Example 2: Initiating the usage of a Tenant Backup Key for the SLOT_0000 group</p> <pre>cxitool Dev=/dev/cs2.0.1 LogonPass=SO_0000,123456 Group=SLOT_0000 SetConfig=SecureGroupBackup,true</pre> <p>Example 3: Setting the permission mask of the key manager to 0x00000020, that is, separating the key manager from the key user (0x00000002).</p> <pre>cxitool Dev=/dev/cs2.0.1 LogonPass=UsrMgr,123456 SetConfig=AuthKeym,0x00000020</pre>
----------------	--

Output	<p>Upon successful execution of the command, no output is given.</p> <p>If the permission denied error message is shown for a <code>cxitool SetConfig</code> command that is applied to a group, verify whether the authentication is correct, whether the authenticating user(s) belong(s) to the specified key group, and whether <code>AllowGroups</code> has been set to <code>true</code>.</p> <p>The verification of the result is done by using the <code>cxitool GetConfig</code> command.</p> <p>If the global configuration or the configuration for a group has been set at least once, the corresponding configuration object is shown in the output list of a specific <code>cxitool ListKeys</code> command. The global configuration object is shown, if this <code>cxitool</code> command is authenticated as a user manager (level 2 in user group 7, permission mask 20000000), and a group configuration object is shown if this <code>cxitool</code> command is authenticated by a security officer (SO, level 2 in user group 2, permission mask 00000200), see ListKeys for details. For a group configuration object, ensure that the authenticated user of the <code>cxitool ListKeys</code> command has been assigned upon user creation to either all key groups (<code>CXI_GROUP=*</code>) or at least to the key group specified by the <code><Group></code> parameter.</p>
---------------	---

3.3.3 ResetConfig

This command resets the global configuration or a key group configuration to the default values and removes the corresponding configuration object, if available.

The execution of this command has an immediate effect. No restart of the device is needed.

Syntax	<code>cxitool [Dev=<dev>] <auth> [Group=<group>] ResetConfig</code>
---------------	---

Authentication	<ul style="list-style-type: none"> ▪ Global configuration A global configuration reset must be authenticated by at least a user manager (level 2 in user group 7, permission mask 20000000). It is irrelevant to which key group the user manager has been assigned. ▪ Key group configuration A key group configuration reset can be done only by a security officer (SO, level 2 in user group 2, permission mask 00000200) and this security officer must have been assigned to the key group he wants to reset
-----------------------	--

Parameter	Description
<dev>	Device address. This parameter can be omitted if the device address has been set as the CRYPTOSERVER environment variable.
<group>	<p>Key group name for cryptographic keys</p> <p>The configuration of this key group is reset. Even if the <code>AllowGroups</code> parameter is false, a key group-specific <code>cxitool ResetConfig</code> command may be performed. The * and ? wildcards are not supported. If PKCS#11 is used, the key group is a PKCS#11 slot name, for example, <code>SLOT_0000</code> or <code>SLOT_0001</code>.</p> <p>If no key group is specified, the global configuration is reset. The Group=* assignment is not supported here</p>

Example	<pre>cxitool Dev=/dev/cs2.0.1 LogonPass=UsrMgr,123456 ResetConfig cxitool Dev=/dev/cs2.0.1 LogonPass=SO_0000,123456 Group=SLOT_0000 ResetConfig</pre>
----------------	---

Output	<p>Upon successful execution of the command, no output is given.</p> <p>The verification of the result is done by performing the <code>cxitool GetConfig</code> command. Using the <code>cxitool ListKeys</code> command, you verify that the corresponding configuration object has been removed.</p>
---------------	--

3.3.4 SecureGroupPass



The `cxitool SecureGroupPass` command is irrelevant because Tenant Backup Keys (TBKs) are not supported.

3.4 Key Management Commands

This chapter deals with the commands provided to handle cryptographic keys. These keys are stored in the u.trust Anchor cHSM's internal key database, `CXIKEY.db`.

Consider the following rule of thumb: if a key has been generated by either P11CAT or p11tool2, we recommend handling this key exclusively with either P11CAT or p11tool2. The analog handling applies for keys generated by cxitool.

3.4.1 ListKeys

This command lists all objects available in the u.trust Anchor cHSM's internal key database, CXIKEY.db , or in an external keystore that match the specified combination of <name> , <group> and <specifier> .

User keys that are used by PKCS#11 can be listed by using the PKCS#11 Administration Tool (P11CAT) and the PKCS#11 command-line Administration Tool (p11tool2) as well.

Other keys, for example, the Default Administrator Key or Master Backup Keys (MBKs), are not listed by the cxitool. Use the u.trust Anchor cHSM's command-line administration tool (csadm) to list, for example, the Master Backup Keys.

Syntax	<code>cxitool [Dev=<dev>] <auth> [Name=<name>] [Group=<group>] [Spec=<spec>] [KeyStoreType=<keystoretype> KeyStoreParam=<keystoreparam>] ListKeys</code>
---------------	--

Authentication	<p>This command is performed independently of the authentication but the items shown in the list depend on the used authentication. The following list describes the shown items:</p> <ul style="list-style-type: none"> ▪ User manager, level 2 in user group 7, 20000000: Global configuration ▪ System manager, level 2 in user group 6, 02000000: Empty list ▪ NTP manager, level 2 in user group 5, 00200000: Empty list ▪ Security officer (SO, key group manager), level 2 in user group 2, 00000200: Key group configuration ▪ Key manager, level 2 in user group 1, 00000020, or level 2 in user group 0, 00000002 (default): Cryptographic keys ▪ Key user, level 2 in user group 0, 00000002: Cryptographic keys <p>The following applies if the items for a group should be shown: Ensure that the authenticated user is assigned (<code>CXI_GROUP</code> user attribute) to the key group(s) specified by the <code><group></code> parameter described above. For more information, see Cryptographic Keys and Key Groups. If the <code>Group=*</code> assignment is used in the <code>cxitool ListKeys</code> command, this does not mean that the keys or configuration objects of the group <code>*</code> are listed, but all keys are listed. Use the <code>csadm ListUser</code> command to verify the key group the user is assigned to. For details, see the u.trust Anchor - csadm Manual. Key groups must not be confused with user groups.</p>
-----------------------	---

Parameter	Description
<dev>	Device address. This parameter can be omitted if the device address has been set as the CRYPTOSERVER environment variable.
<name>	Cryptographic key name. The * and the ? wildcards are supported.
<group>	Key group. The * and the ? wildcards are supported.
<spec>	Key specifier, an integer value. No wildcards are supported.
<keystoretype>	<p>Type of an external keystore Its value is either SDB (for legacy .sdb files) or ODBC. If <code>KeyStoreType</code> is used, <code>KeyStoreParam</code> must be used as well. If <code>KeyStoreType</code> is used, it must be used before <code>KeyStoreParam</code>. If <code>KeyStoreType</code> is used, this cxitool command only lists the keys in the external keystore. In this case, an unspecified key specifier is shown as empty. If <code>KeyStoreType</code> is not used, this cxitool command only lists keys the u.trust Anchor CHSM's internal key database, <code>CXIKEY.db</code>.</p>

Parameter	Description
<keystoreparam>	<p>Configuration parameter for an external keystore</p> <p>If <code>KeyStoreType</code> is used, <code>KeyStoreParam</code> must be used as well.</p> <p>If <code>KeyStoreType</code> is used, it must be used before <code>KeyStoreParam</code>.</p> <ul style="list-style-type: none"> If <code>KeyStoreType=SDB</code> <code>KeyStoreParam</code> specifies the path to an external keystore file (<code>/path/to/store.sdb</code>). If <code>KeyStoreType=ODB</code> <code>KeyStoreParam</code> specifies the data source name as defined in format <code>DSN=DATA SOURCE NAME</code> or the complete ODBC connection string.

Example	<p>Example 1: List the global configuration object.</p> <pre>cxitool Dev=/dev/cs2.0.1 LogonPass=UsrMgr,12345678 ListKeys</pre> <p>Because the <code>UsrMgr</code> user has the permission mask 20000000, only the global configuration object is shown. No keys are shown. Configuration objects are only shown if they have been created before by using the <code>cxitool SetConfig</code> command.</p> <p>Example 2: List all keys a user assigned to the 1234 key group has access to plus the keys that are not assigned to any group. Do not show the configuration objects.</p> <pre>cxitool Dev=/dev/cs2.0.1 LogonPass=KeyUsr,12345678 ListKeys</pre> <p>The <code>KeyUsr</code> user (permission mask: 00000002) must have been assigned to the 1234 key group upon creation. Because of the permission mask 00000002, only keys and no configuration objects are shown.</p> <p>Example 3: List all keys a user assigned to the 1234 key group and a user assigned to the SLOT_0000 group have access to plus the keys that are not assigned to any group. Do not show the configuration objects.</p> <pre>cxitool Dev=/dev/cs2.0.1 LogonPass=KeyUsr,12345678 LogonPass=USR_0000,12345678 ListKeys</pre> <p>The <code>KeyUsr</code> user (permission mask: 00000002) must have been assigned to the 1234 key group upon creation, and the <code>USR_0000</code> user (permission mask: 00000002) must have been assigned to the SLOT_0000 key group. Because of the permission mask 00000004, only keys and no configuration objects are shown.</p> <p>Example 4: List the global configuration object and the configuration object of the 1234 key group and the SLOT_0000 group.</p> <pre>cxitool Dev=/dev/cs2.0.1 LogonPass=UsrMgr,12345678 LogonPass=SO1234,12345678 LogonPass=SO_0000,12345678 ListKeys</pre> <p>Because of the <code>UsrMgr</code> user (permission mask: 20000000), the global configuration object is shown.</p> <p>The <code>SO1234</code> user (permission mask: 00000200) must have been assigned to the 1234 key group upon creation, and the <code>SO_0000</code> user (permission mask: 00000200) must have been assigned to the SLOT_0000 key group. Because of the permission mask 00000400, only configuration objects of the corresponding groups and no keys are shown.</p> <p>Global and group-specific configuration objects are only shown if they have been created before by using the <code>cxitool SetConfig</code> command.</p>
----------------	---

--	--

Upon successful execution of the command, a list of keys is returned. The parameters are explained below in detail.

The cryptographic key's usage and the availability of a certificate for this key is not shown in the output list. Use the `cxitool KeyInfo` command to show the usage and the certificate of a single cryptographic key.

Example 1:

idx	algo	size	type	group	name	spec
1	RAW	0	conf			-1

Example 2:

idx	algo	size	type	group	name	spec
1	AES	256	sec		AES_256	11
2	RSA	2048	pub+prv		RSA_2048	12
3	RSA	2048	pub+prv	1234		19
4	RSA	2048	pub+prv	1234		20

Example 3:

idx	algo	size	type	group	name	spec
1	AES	256	sec		AES_256	11
2	RSA	2048	pub+prv		RSA_2048	12
3	RSA	2048	pub+prv	1234		19
4	RSA	2048	pub+prv	1234		20
5	AES	256	sec	SLOT_0000		2
6	DES	56	sec	SLOT_0000		3
7	RSA	2048	pub	SLOT_0000		4
8	RSA	2048	prv	SLOT_0000		5
9	ECDSA	192	pub	SLOT_0000		6
10	ECDSA	192	prv	SLOT_0000		7
11	RSA	512	pub+prv	SLOT_0000		1
12	DSA	512	pub+prv	SLOT_0000		10

Example 4:

idx	algo	size	type	group	name	spec
1	RAW	0	conf			-1
2	RAW	0	conf	1234		-1
3	RAW	0	conf	SLOT_0000		-1

3.4.2 KeyInfo

This command retrieves key information such as the key name, the corresponding key group, the key usage, the encryption algorithm and the key's size.

Syntax	<code>cxitool [Dev=<dev>] <auth> [Name=<name>] [Group=<group>] [Spec=<spec>] [KeyStoreType=<keystoretype> KeyStoreParam=<keystoreparam>] KeyInfo</code>
---------------	---

Authentication	<p>This command must be authenticated by at least one of the following user roles. This applies to keys and configuration objects.</p> <ul style="list-style-type: none"> Security officer, SO, level 2 in user group 2, permission mask 00000200 Key manager; either level 2 in user group 1, permission mask 00000020 or level 2 in user group 0, permission mask 00000002 (default) Key user, level 2 in user group 0, permission mask 00000002 <p>Ensure that the authenticated user is assigned (<code>CXI_GROUP</code> user attribute) to the key group(s) specified by the <code><group></code> parameter. For more information, see Cryptographic Keys and Key Groups.</p> <p>A user is assigned to a key group on creation. Use the <code>csadm ListUser</code> command to verify the key group the user is assigned to. For details, see the u.trust Anchor - csadm Manual.</p> <p>Key groups must not be confused with user groups.</p>
-----------------------	--

Parameter	Description
<code><dev></code>	Device address. This parameter can be omitted if the device address has been set as the CRYPTOSERVER environment variable.
<code><name></code>	Cryptographic key name The combination of <code><name></code> , <code><group></code> and <code><spec></code> specifies the cryptographic key and must be unique. The * and ? wildcards are not supported.
<code><group></code>	Key group name for cryptographic keys
<code><spec></code>	Cryptographic key specifier
<code><keystoretype></code>	Type of an external keystore Its value is either <code>SDB</code> (for legacy .sdb files) or <code>ODBC</code> . If <code>KeyStoreType</code> is used, <code>KeyStoreParam</code> must be used as well. If <code>KeyStoreType</code> is used, it must be used before <code>KeyStoreParam</code> . If <code>KeyStoreType</code> is used, this cxitool command only lists the keys in the external keystore. In this case, an unspecified key specifier is shown as empty. If <code>KeyStoreType</code> is not used, this cxitool command only lists keys the u.trust Anchor's internal key database, <code>CXIKEY.db</code> .

Parameter	Description
<keystoreparam>	<p>Configuration parameter for an external keystore If <code>KeyStoreType</code> is used, <code>KeyStoreParam</code> must be used as well. If <code>KeyStoreType</code> is used, it must be used before <code>KeyStoreParam</code>.</p> <ul style="list-style-type: none"> If <code>KeyStoreType=SDB</code> <code>KeyStoreParam</code> specifies the path to an external keystore file (<code>/path/to/store.sdb</code>). If <code>KeyStoreType=ODB</code> <code>KeyStoreParam</code> specifies the data source name as defined in format <code>DSN=DATA SOURCE NAME</code> or the complete ODBC connection string.

Example	<code>cxitool Dev=/dev/cs2.0.1 LogonPass=KeyMgr,12345678 Name=Key25 Group=1234 KeyInfo</code>
----------------	---

Output	<p>Upon successful execution of the command, the information of keys is returned. Some of the attributes are explained below in detail.</p> <pre> Group : 1234 Name : Key25 Specifier : 12 Algo : RSA Size : 2048 Export : 0x00000000 Extractable : 0 Sensitive : 1 Usage : 0x000002bb Encrypt : 1 Decrypt : 1 Sign : 1 Verify : 1 Verify_Rec. : 1 Wrap : 1 Unwrap : 1 Derive : 0 BlockLen : 256 Type : pub+prv DateGen : 20171206111359Z DateExp : 20491231235959Z Label : Modulus: 0 b6f53a32 19d91947 fbe7895f 06a78258 :2 G - X 10 7c8ad448 f1b759ab 8a11e4e9 ee379870 H Y - 7 p 20 9425c454 e4541ae2 af926474 3c5b90fe % T T dt<[30 cfa52304 8b3d0453 a4ec1e33 c9fbf41b # = S 3 40 061265ef fb13df67 08c221f0 3aea518a e g ! : Q 50 07decc55 d3d31d1a bcd76cf5 747ec7fd U , 1 t~ 60 8dd7fe91 d79f60fa e5050cb9 ce1bebec ' 70 a1371dd7 be4bc343 39cd7356 b8a6c051 7 K C9 sV Q 80 a800078b a5102592 1ee9d604 3b241410 % ;\$ </pre>
---------------	--

90	fbfeca0f	3fd9acfe	c8b37449	8196cb6c		?	tI	l	
a0	6661a4c1	f7b27519	698da38f	a7193dd7		fa	u	i	=
b0	0f9652ff	0b4988e8	f706b5c0	fbbfede9		R	I		
c0	0c01b857	661cc91f	9c5e7c2d	c53cc0f4		Wf	^		<
d0	2b49f2d0	d571dd3d	36153318	6a31d3d3		+I	q	=6	3 j1
e0	a1d0ffc9	c3ba4b5d	6cbff2f0	69f72b45			K]l	i	+E
f0	f6b21636	6499f4ba	dfa5df5b	c0ae0655		6d		[U
Exponent:									
	0	010001							
Certificate:									
0	308203e2	308202ca	a0030201	02020201		0	0		
10	23300d06	092a8648	86f70d01	010b0500		#0	*	H	
20	3081a031	1a301806	03550403	16114d79		0	1	0	U My
30	43415369	676e696e	674b6579	5f303131		CASigningKey_011			
40	0c300a06	03550405	13033132	33310b30		0	U	1231	0
50	09060355	04061302	4445310f	300d0603		U	DE1	0	
60	55040713	06416163	68656e31	0c300a06		U	Aachen1	0	
... clip ... clip ...									
360	bace5667	0924885d	aa58cfb8	99976e7b		Vg	\$]	X n{
370	0684e369	069b7a81	2e4507a4	ac2d15ca		i	z	.E	-
380	f4cdb7eb	46ef16c0	5b7aeab8	c3a16ec6		F	[z		n
390	3e59aa8f	441f8d72	a06b43f6	62215655		>Y	D	r	kC b!VU
3a0	87c585f4	13996814	5e93e7f1	70847204			h	^	p r
3b0	78eb2577	c95717ff	0cc115fc	e981758f		x	%w	W	u
3c0	4fcd8e6	aec0e3a4	43c09da8	93478136		0		C	G 6
3d0	e57be585	b741313b	29f4e724	f3f92528		{	A1;)	\$ %(
3e0	f127bf87	a4ad				'			

Parameter	Description
Export	<p>Under <code>Export</code>, the export property is shown in two representations. The export property indicates if and under which circumstances a cryptographic key may be exported. Both representations are equivalent to each other, see Export.</p> <ul style="list-style-type: none"> ▪ CXI representation The hexadecimal <code>Export</code> value is the CXI representation of the export property. <ul style="list-style-type: none"> • <code>0x00000000</code> (<code>deny</code>) Exporting the keys is not allowed. • <code>0x00000001</code> (<code>allow</code>) Exporting the key is allowed in encrypted form. • <code>0x00000002</code> (<code>allow_plain</code>) Exporting the key in plaintext is allowed. • <code>0x00010000</code> (<code>deny_backup</code>) This value indicates that the <code>cxitool BackupKey</code> command cannot be performed. However, the <code>csadm BackupDatabase</code> command can be performed. The <code>deny_backup</code> value can be shown here but it cannot be set by performing the <code>cxitool Export</code> parameter because it can be set only by the CXI API. ▪ PKCS#11 representation <code>Extractable</code> and <code>Sensitive</code> are the PKCS#11 representation of the export property. See Export for details.
Usage	<p>The hexadecimal <code>Usage</code> value is a bit field indicating the purposes a cryptographic key may be used for, see Usage for details. This value is the <code>cxitool</code> representation of usage properties. The corresponding PKCS#11 representations of usage properties are shown in the next lines (<code>Encrypt</code>, <code>Decrypt</code>, <code>Sign</code>, <code>Verify</code>, <code>Verify_Rec.</code>, <code>Wrap</code>, <code>Unwrap</code> and <code>Derive</code>). Again, see Usage for details. <code>1</code> means true, and <code>0</code> means false. <code>Verify_Rec.</code> means that the signed message can be recovered from the signature. This is irrelevant here because it is ignored by <code>cxitool</code>. Once set, the <code>Usage</code> parameter cannot be changed.</p>
BlockLen	Block length of the encryption algorithm. This is relevant for DES and AES only.
DataGen	Date and time of the cryptographic key generation. <code>Z</code> indicates the Zulu time, the Coordinated Universal Time (UTC).
DataExp	Date and time of the cryptographic key expiration. <code>Z</code> indicates the Zulu time, the Coordinated Universal Time (UTC).

Parameter	Description
Label	<p>Key label</p> <p>For PKCS#11, this label is identical to the key label shown at Object Management > List Attributes in the PKCS#11 CryptoServer Administration Tool (P11CAT) or by performing the <code>p11tool2 ListObjects</code> command. For details about the P11CAT, see the CryptoServer – PKCS#11 P11CAT Manual and for details about the p11tool2 commands, see the u.trust Anchor – PKCS#11 p11tool2 Reference Manual.</p>
Modulus/Exponent	<p>Modulus (m) and exponent (e) used for the modular exponentiation ($b^e \bmod m$) in public-key cryptography</p> <p>The output for an AES or a DES key has no <code>Modulus</code> and <code>Exponent</code> property.</p>
Certificate	<p>If a certificate has been imported for the specified key (see the <code>cxitool ImportCert</code> command or the <code>cxitool ImportP12</code> command), <code>Certificate</code> shows the contents of the certificate, otherwise, it is empty.</p> <p>Example:</p> <pre>commonName=1234 Key35 serialNumber=123 countryName=DE localityName=Aachen stateOrProvinceName=NRW organizationName=Utimaco IS GmbH organizationalUnitName=Support title=Support description=Test emailAddress=support@utimaco.com</pre> <p>If the u.trust Anchor is in FIPS mode, the FIPS usage is shown as well:</p> <p>Example:</p> <pre>... Label : Fips usage : RSA_PKCS Modulus: ...</pre> <p>For details about the FIPS usage, see SetFipsUsage. The <code>cxitool SetFipsUsage</code> command must neither be confused with the <code>Usage</code> parameter, see Usage, nor with the <code>KeyUsage</code> parameter of the <code>cxitool SelfSignedCert</code> command, see SelfSignedCert.</p>

Parameter	Description
Different Output	<ul style="list-style-type: none">The output for a DSA key slightly differs. It has no Modulus and Exponent property but a P, Q, G and a PubKey property.

Parameter	Description
	<ul style="list-style-type: none"> The output for an AES or a DES has no <code>Modulus</code> and <code>Exponent</code> property.
	Example:
	P: <pre> 0 df015430 961e433f 515ad1c6 dfe49bad T0 C?QZ 10 c96b63dd 6bedbd44 305e7f6c 97f16da5 kc k D0^ l m 20 9bdbd0dd edde58be 5be03deb 50f19cbf X [= P 30 0325d370 8629be5a 6be42e75 68ff78b7 % p) Zk .uh x </pre>
	Q: <pre> 0 fe97740a 9b2009e6 6518ab55 2a35a096 t e U*5 10 79eef8a9 y </pre>
	G: <pre> 0 cf3785ee 21d440d3 c911adfd 22ce5f25 7 ! @ " _% 10 990d7a5b 7471b776 0ccb175 389e4c3e z[tq v u8 L> 20 27b169e3 f81b35c5 ffa6a0f3 5388af7b ' i 5 S { 30 7c832d74 337d126b 8f73dd6b 63597aba -t3} k s kcYz </pre>
	PubKey: <pre> 0 93287718 f0629aeb 79824312 74d38194 (w b y C t 10 5e435e7a 067ec8e1 b663653f 5ccb5e31 ^C^z ~ ce?\ ^1 20 45cea594 3950de58 6bcb55f8 d0fd397a E 9P Xk U 9z 30 ab134a52 f936217a 9ee50703 9c423bf8 JR 6!z B; </pre>
	<ul style="list-style-type: none"> The output for an ECDSA key slightly differs. It has no <code>Modulus</code> and <code>Exponent</code> property but a <code>Curve</code> and a <code>PubKey</code> property.
	Example:

Parameter	Description
	Curve: 0 73656370 31393272 31 secp192r1 PubKey: 0 043beae1 c2a0cfbb 8f11c1e6 2e564f20 ; .V0 10 e3b7edd8 1b2b5357 289a694d 4ab65867 +SW(iMJ Xg 20 1232d1db ce050218 810a5fac 082b3d91 2 - += 30 8b

Table 14: Output parameters of cxitool KeyInfo

3.4.3 GenerateKey

This command generates a cryptographic key. See the `algo` parameter for the supported algorithms.



Cryptographic keys and configuration data are stored in the u.trust Anchor cHSM's internal key database, `CXIKEY.db`. This database is deleted on alarm occurrence. For details about alarms, see *Alarms* in the u.trust Anchor - Containerized Hardware Security Module (cHSM) - Administration Manual.

We highly recommend creating a backup of the `CXIKEY.db` file with the `csadm BackupDatabase` command, see *BackupDatabase* in the u.trust Anchor - csadm Manual or using the `cxitool BackupKey` command, so you can restore the cryptographic keys and the configuration data. In FIPS mode, csadm and CAT cannot be used for backup purposes. In this case, only the `cxitool BackupKey` command backs up cryptographic keys and the configuration data.

We also highly recommend ensuring that MBK (Master Backup Key) shares are available when backing up and restoring the `CXIKEY.db` database. These shares are necessary to make the restored database usable again. Otherwise, all data in this database remains inaccessible. For details about MBKs, see *Commands for Managing the Master Backup Keys* in the u.trust Anchor - csadm Manual and *Master Backup Key* in the u.trust Anchor - Containerized Hardware Security Module (cHSM) - Administration Manual.


If the `Name`, `Group` and `Spec` parameters of an already existing key are identical to the corresponding parameters of the key to be generated, the existing key is overwritten. This default behavior can be changed by configuring the `Overwrite` parameter.

When using the `cxitool GenerateKey` command to create a cryptographic key in an external database, consider that this external database is encrypted with the current Master Backup Key (MBK) in MBK slot 3. If you then change the MBK in this MBK slot (`csadm`

MBKImportKey command) and you then try to show the available cryptographic keys (cxitool ListKeys command), the following error message is shown.

```
Error B0680080
CryptoServer module CXI
key blob encrypted with different MBK
```

For details about MBKs, see *Commands for Managing the Master Backup Keys* in the u.trust Anchor – csadm Manual.




Certain types of shell processes treat certain characters (for example, commas, colons, semi-colons) differently. If the execution of the `cxitool GenerateKey` command fails with an error message from the shell about missing parameter (algo, size) or illegal parameter format, quoting parameter values may be necessary.

For example, a correct command entry in the Microsoft PowerShell:

```
cxitool Export=allow Name=DemoRSA GenerateKey="RSA,2048"
```

Use the `cxitool ListKeys` command to verify the result of the `cxitool GenerateKey` command.

Syntax	<pre>cxitool [Dev=<dev>] <auth> [Name=<name>] [Group=<group>] [Spec=<spec>] [Overwrite=<overwrite>] [Export=<export>] [Usage=<usage>] [PubKeyFile=<filename>] [KeyStoreType=<keystoretype> KeyStoreParam=<keystoreparam>] GenerateKey=<algo>,<size/curve></pre>
Authentication	<p>This command must be authenticated by a key manager. His level may either be 2 in user group 0 (permission mask: 00000002, default value) or the level has been changed to level 2 in user group 1 (00000020). That is, a user of this user group must be authenticated by including a <code>LogonSign</code> command or <code>LogonPass</code> command into the <code>GenerateKey</code> command.</p> <p>Ensure that the authenticated user is assigned (<code>CXI_GROUP</code> user attribute) to the key group(s) specified by the <code><group></code> parameter. For more information, see <i>Cryptographic Keys and Key Groups</i>.</p> <p>A user is assigned to a key group on creation. Use the <code>csadm ListUser</code> command to verify the key group the user is assigned to. For details, see the u.trust Anchor – csadm Manual.</p> <p>Key groups must not be confused with user groups.</p>

Parameter	Description
<dev>	Device address. This parameter can be omitted if the device address has been set as the CRYPTOSERVER environment variable.
<name>	<p>Cryptographic key name</p> <p>The combination of <name>, <group> and <spec> specifies the cryptographic key to be generated. At least one of these parameters is mandatory.</p> <p>If a key name is specified, the minimum key name length is 1 character, the maximum is 256 characters. When using the external keystore, the use of any kind of brackets within the key name, e.g. () [] { } , is not supported. All other printable ASCII characters except the * and ? wildcards are supported.</p>
<group>	<p>Key group name for cryptographic keys</p> <p><group> must match the key group name that has been assigned to a user on the user creation (CXI_GROUP=<key_group_name>) for this user to be able to generate the cryptographic key. In this CXI_GROUP assignment, the * and ? wildcards are supported.</p> <p>If the Group parameter is not used, the cryptographic key to be generated is not assigned to any group.</p> <p>If a key group name is specified, the minimum key group name length is 1 character, the maximum is 256 characters.</p> <p>Never use a blank or one or more of the following characters in a key group name: *, ?, [,], {, }, <, >, :, ^, ", !, /, \, . Apart from these characters, all printable ASCII characters are supported for key group names.</p>
<spec>	<p>Cryptographic key specifier</p> <p><spec> is used to differentiate between keys if the key name and the key group are identical or empty. <spec> is typically not set manually. For details, see Cryptographic Keys and Key Groups.</p>
<overwrite>	<ul style="list-style-type: none"> 1 <p>If a key with the same combination of name, group and specifier exists already, it is overwritten (Default). The <code>cxitool GenerateKey</code> command is executed immediately. You are not asked whether you really want to overwrite the already existing key.</p> <hr/>  If the <code>Overwrite</code> parameter is set to 1, the original key is overwritten, and all data that has been encrypted in the past with the original key cannot be decrypted anymore unless the key has been backed up. <hr/> 0 <p>If the key to be generated exists already, it is not overwritten and an error message is shown.</p>

Parameter	Description
<export>	<p>This parameter defines whether the key to be imported can be exported or not, see Export.</p> <ul style="list-style-type: none"> ▪ <code>deny</code> Exporting the key is not allowed (Default), corresponds to <code>0x00000000</code>. ▪ <code>allow</code> Exporting the key in encrypted form, corresponds to <code>0x00000001</code>.
<usage>	<p>Usage of the cryptographic key, see Usage. The <code>Usage</code> parameter must neither be confused with the <code>cxitool SetFipsUsage</code> command (see SetFipsUsage) nor with the <code>KeyUsage</code> parameter of the <code>cxitool SelfSignedCert</code> command, see SelfSignedCert.</p>
<filename>	<p>For RSA and EC keys, the generated public key is written to this file. For other key types, an error message is shown.</p>
<keystoretype>	<p>Type of an external keystore Its value is either SDB (for legacy .sdb files) or ODBC. If <code>KeyStoreType</code> is used, <code>KeyStoreParam</code> must be used as well. If <code>KeyStoreType</code> is used, it must be used before <code>KeyStoreParam</code>. If <code>KeyStoreType</code> is used, this <code>cxitool</code> command only lists the keys in the external keystore. In this case, an unspecified key specifier is shown as empty. If <code>KeyStoreType</code> is not used, this <code>cxitool</code> command only lists keys the u.trust Anchor CHSM's internal key database, <code>CXIKKEY.db</code>.</p>
<keystoreparam>	<p>Configuration parameter for an external keystore If <code>KeyStoreType</code> is used, <code>KeyStoreParam</code> must be used as well. If <code>KeyStoreType</code> is used, it must be used before <code>KeyStoreParam</code>.</p> <ul style="list-style-type: none"> ▪ If <code>KeyStoreType=SDB</code> <code>KeyStoreParam</code> specifies the path to an external keystore file (<code>/path/to/store.sdb</code>). ▪ If <code>KeyStoreType=ODBC</code> <code>KeyStoreParam</code> specifies the data source name as defined in format <code>DSN=DATA SOURCE NAME</code> or the complete ODBC connection string.
<algo>	<p>The algorithm the key shall be generated for. AES, RSA and EC are supported. If the <code>Export</code> parameter has been set to <code>allow</code>, there might be restrictions to the supported algorithms. For details, see Export. The <code>SecureRSAComponents</code> configuration parameter is always set to <code>true</code> so that new RSA keys with public exponents below <code>0x10001</code> cannot be created or imported. This configuration parameter can be global or group-specific, see SetConfig.</p>

Parameter	Description
<size/curve>	<p>Key size in bit for the AES and RSA algorithms, and elliptic curve for the EC algorithm</p> <ul style="list-style-type: none"> AES: 128, 192 or 256 RSA: From 2048 up to 16384 in steps of 2 bit The higher the number the longer it takes to generate the key. EC: Raw name or oid of any built-in elliptic curve. By using oid syntax, oid:<curvename>, the curve oid is saved as curve parameter to ensure PKCS11 compliance. For example: <ul style="list-style-type: none"> SEC 2 curves, e.g., secp256k1 or oid:secp256k1 brainpool curves, e.g., brainpoolP320t1 or oid:brainpoolP320t1 NIST curves, e.g., NIST-P256 or oid:NIST-P256 All supported elliptic curve names are listed in Built-in Elliptic Curves.

Example	<pre> cxitool Dev=/dev/cs2.0.1 LogonPass=KeyMgr,12345678 Group=1234 GenerateKey=RSA,2048 cxitool Dev=/dev/cs2.0.1 LogonPass=KeyMgr,12345678 Name=RSA4000 Group=1234 GenerateKey=RSA,4000 cxitool Dev=/dev/cs2.0.1 LogonPass=KeyMgr,12345678 Name=EC_secp192k1 Group=1234 GenerateKey=EC,secp192k1 cxitool Dev=/dev/cs2.0.1 LogonPass=KeyMgr,12345678 Name=Key100 Group=1234 Export=ALLOW GenerateKey=RSA,2048 cxitool Dev=/dev/cs2.0.1 LogonPass=KeyMgr,12345678 Name=Key1002 Group=1234 Usage=ENCRYPT GenerateKey=AES,256 cxitool Dev=/dev/cs2.0.1 LogonPass=KeyMgr,12345678 Export=allow Usage=SIGN,VERIFY PubKeyFile=ec_pub_verify.key Name=EC_sign_key GenerateKey=EC,oid:FRP256v1 </pre>
---------	---

Output	<p>Upon successful execution of the command, no output is given. Error message <code>Permission denied</code>: The authentication has failed due to a user group that is not appropriate, the key manager is not logged in, or the user and the cryptographic key belong to different key groups.</p>
--------	---

3.4.4 DeleteKey

This command deletes one or several cryptographic keys.

Syntax	<pre> cxitool [Dev=<dev>] <auth> [Name=<name>] [Group=<group>] [Spec=<spec>] [KeyStoreType=<keystoretype>] KeyStoreParam=<keystoreparam>] DeleteKey </pre>
--------	--

Authentication	<ul style="list-style-type: none"> ▪ User manager, level 2 in user group 7, 20000000: Deleting the global configuration ▪ Security officer (SO, key group manager), level 2 in user group 2, 00000200: Deleting the key group configuration ▪ Key manager, level 2 in user group 1, 00000020, or level 2 in user group 0, 00000002 (default): Deleting a cryptographic key <p>The following applies to the security officer and the key manager: Ensure that the authenticated user is assigned (<code>CXI_GROUP</code> user attribute) to the key group(s) specified by the <code><group></code> parameter. For more information, see Cryptographic Keys and Key Groups. A user is assigned to a key group on creation. Use the <code>csadm ListUser</code> command to verify the key group the user is assigned to. For details, see the u.trust Anchor – csadm Manual. Key groups must not be confused with user groups.</p>
-----------------------	---

Parameter	Description
<dev>	Device address. This parameter can be omitted if the device address has been set as the CRYPTOSERVER environment variable.
<name>	<p>Cryptographic key name At least one of <code><name></code>, <code><group></code> or <code><spec></code> is mandatory. The * and ? wildcards are supported. If <code>Name=""</code> or <code>Name=</code>, only keys without a name are deleted. Combinations of empty names and groups with wildcards are supported, for example:</p> <ul style="list-style-type: none"> ▪ <code>Group="1234"</code> and <code>Name="*"</code> deletes all keys of group 1234. ▪ <code>Group="1234"</code> and <code>Name=""</code> deletes all keys with empty names of group 1234. <p>If you want to delete a global configuration object or a key group configuration object, omit the <code>Name</code> parameter because configuration objects do not have names. As described above, make sure to use a user with the correct permission mask (20000000 for deleting the global configuration object or 00000200 for deleting the key group configuration object) for the authentication of the command.</p>
<group>	<p>Key group name for cryptographic key. If <code>Group=""</code> or <code>Group=</code>, only a global configuration object (authenticated user with the permission mask 20000000) or keys without a group are deleted. Again, combinations of empty names and groups and with wildcards are supported, for example:</p> <ul style="list-style-type: none"> ▪ <code>Group="1234"</code> and <code>Name="*"</code> deletes all keys of group 1234. ▪ <code>Group="1234"</code> and <code>Name=""</code> deletes all keys with empty names of group 1234.

Parameter	Description
<spec>	Cryptographic key specifier
<keystoretype>	<p>Type of an external keystore Its value is either SDB (for legacy .sdb files) or ODBC.</p> <p>If <code>KeyStoreType</code> is used, <code>KeyStoreParam</code> must be used as well. If <code>KeyStoreType</code> is used, it must be used before <code>KeyStoreParam</code>.</p> <p>If <code>KeyStoreType</code> is used, this cxitool command only lists the keys in the external keystore. In this case, an unspecified key specifier is shown as empty.</p> <p>If <code>KeyStoreType</code> is not used, this cxitool command only lists keys the u.trust Anchor cHSM's internal key database, <code>CXIKEY.db</code>.</p>
<keystoreparam>	<p>Configuration parameter for an external keystore</p> <p>If <code>KeyStoreType</code> is used, <code>KeyStoreParam</code> must be used as well. If <code>KeyStoreType</code> is used, it must be used before <code>KeyStoreParam</code>.</p> <ul style="list-style-type: none"> If <code>KeyStoreType=SDB</code> <code>KeyStoreParam</code> specifies the path to an external keystore file (<code>/path/to/store.sdb</code>). If <code>KeyStoreType=ODBC</code> <code>KeyStoreParam</code> specifies the data source name as defined in format <code>DSN=DATA SOURCE NAME</code> or the complete ODBC connection string.

Example	<p>Example: Delete the cryptographic key with Name=KeyShi01.</p> <pre>cxitool Dev=/dev/cs2.0.1 LogonPass=KeyMgr,12345678 Name=KeyShi01 DeleteKey</pre>
----------------	--

Output	<p>Upon successful execution of the command, the report on the deleted key or key group is returned.</p> <p>Example:</p> <pre> idx algo size group name spec ----- 1 RSA 2048 1234 KeyShi01 12 1 key(s) deleted </pre>
---------------	---

3.4.5 BackupKey

This command backs up one or several cryptographic keys from the u.trust Anchor cHSM's internal key database, `CXIKEY.db`, or an external keystore to a keyfile (`*.kbk`). An already existing keyfile is overwritten. This keyfile is encrypted with the u.trust Anchor cHSM's Master Backup Key (MBK).



Perform the `csadm MBKListKeys` command to determine which Master Backup Key (MBK) is currently in use in MBK slot 3 by u.trust Anchor cHSM. This MBK is used by the `cxitool BackupKey` command to encrypt the backup file to be generated. If the MBK in MBK slot 3 is the autogenerated MBK named `AUTO-GEN`, the `cxitool BackupKey` command cannot be performed. Import a different MBK into MBK slot 3 using the `csadm MBKImportKey` command, see the u.trust Anchor – csadm Manual.

It is important to note down which MBK has been used because for a successful restoring of this backup file at a later date it is necessary that the same MBK is in MBK slot 3 or after an MBK rollover in an MBK slot ≥ 3 .

Otherwise, the backup file is inaccessible. This might be the result of the execution of a `csadm MBKImportKey` command, see *Master Backup Key Rollover* in the u.trust Anchor - Containerized Hardware Security Module (cHSM) - Administration Manual.

It is not possible to retrieve the MBK by which a backup file has been generated from this backup file.



If the `Export` parameter of a key has been set to `deny_backup` value (`0x00010000`), the `cxitool BackupKey` command cannot be performed. However, the `csadm BackupDatabase` command can be performed. The `deny_backup` value cannot be set by the `cxitool Export` parameter. It can be set only by the CXI API. However, it can be shown by the `cxitool KeyInfo` command. For details, see [KeyInfo](#).

Syntax

```
cxitool [Dev=<dev>] <auth> [Name=<name>] [Group=<group>]
[Spec=<spec>] [KeyStoreType=<keystoretype>]
KeyStoreParam=<keystoreparam>] [OutDir=<outdir>] BackupKey
```

Authentication	<ul style="list-style-type: none"> ▪ User manager, level 2 in user group 7, 20000000: Backing up the global configuration ▪ Security officer (SO, key group manager), level 2 in user group 2, 00000200: Backing up the key group configuration ▪ Key manager, level 2 in user group 1, 00000020, or level 2 in user group 0, 00000002 (default): Backing up a cryptographic key <p>The following applies to the security officer and the key manager:</p> <p>Ensure that the authenticated user is assigned (<code>CXI_GROUP</code> user attribute) to the key group(s) specified by the <code><group></code> parameter. For more information, see Cryptographic Keys and Key Groups.</p> <p>A user is assigned to a key group on creation. Use the <code>csadm ListUser</code> command to verify the key group the user is assigned to. For details, see the u.trust Anchor – csadm Manual.</p> <p>Key groups must not be confused with user groups.</p>
-----------------------	--

Parameter	Description
<dev>	Device address. This parameter can be omitted if the device address has been set as the CRYPTOSERVER environment variable.
<name>	<p>Cryptographic key name</p> <p>At least one of <code><name></code>, <code><group></code> or <code><spec></code> is mandatory. The * and ? wildcards are supported.</p> <p>If <code>Name=""</code> or <code>Name=</code>, only keys without a name are backed up.</p> <p>Combinations of empty names and groups with wildcards are supported, for example:</p> <ul style="list-style-type: none"> ▪ <code>Group="1234"</code> and <code>Name="*"</code> backs up all keys of group 1234. ▪ <code>Group="1234"</code> and <code>Name=""</code> backs up all keys with empty names of group 1234. <p>If you want to back up a global configuration object or a key group configuration object, omit the <code>Name</code> parameter because configuration objects do not have names. As described above, make sure to use a user with the correct permission mask (20000000 for backing up the global configuration object or 00000200 for backing up the key group configuration object) for the authentication of the command.</p>
<group>	<p>Key group name for cryptographic key.</p> <p>If <code>Group=""</code> or <code>Group=</code>, only a global configuration object (authenticated user with the permission mask 20000000) or keys without a group are backed up. Again, combinations of empty names and groups and with wildcards are supported, for example:</p> <ul style="list-style-type: none"> ▪ <code>Group="1234"</code> and <code>Name="*"</code> backs up all keys of group 1234. ▪ <code>Group="1234"</code> and <code>Name=""</code> backs up all keys with empty names of group 1234.
<spec>	Cryptographic key specifier

Parameter	Description
<keystoretype>	<p>Type of an external keystore Its value is either SDB (for legacy .sdb files) or ODBC.</p> <p>If <code>KeyStoreType</code> is used, <code>KeyStoreParam</code> must be used as well.</p> <p>If <code>KeyStoreType</code> is used, it must be used before <code>KeyStoreParam</code>.</p> <p>If <code>KeyStoreType</code> is used, this <code>cxitool</code> command only lists the keys in the external keystore. In this case, an unspecified key specifier is shown as empty.</p> <p>If <code>KeyStoreType</code> is not used, this <code>cxitool</code> command only lists keys the u.trust Anchor's internal key database, <code>CXIKEY.db</code>.</p>
<keystoreparam>	<p>Configuration parameter for an external keystore</p> <p>If <code>KeyStoreType</code> is used, <code>KeyStoreParam</code> must be used as well.</p> <p>If <code>KeyStoreType</code> is used, it must be used before <code>KeyStoreParam</code>.</p> <ul style="list-style-type: none"> If <code>KeyStoreType=SDB</code> <code>KeyStoreParam</code> specifies the path to an external keystore file (<code>/path/to/store.sdb</code>). If <code>KeyStoreType=ODBC</code> <code>KeyStoreParam</code> specifies the data source name as defined in format <code>DSN=DATA SOURCE NAME</code> or the complete ODBC connection string.
<outdir>	<p>Name of the directory the keyfile (<code>*.kbk</code> file) is stored in. The current directory of the command line is the default directory.</p> <p>The name of a keyfile is built according to the following pattern: <code>[<group>_][<name>_][<spec>].kbk</code></p> <p>The key group name is omitted if it is not available. The key name is omitted if it is not available. The key specifier is omitted if its value is -1, the default value.</p>

Example	<p>Example 1: Backing up one cryptographic key specified by <code>Group=1234</code> <code>Name=RSA2048</code> <code>cxitool Dev=/dev/cs2.0.1 LogonPass=KeyMgr1234,123456</code> <code>Group=1234 Name=RSA2048 OutDir=C:\Utimaco\CryptoServer BackupKey</code></p> <p>Example 2: Backing up all cryptographic keys of Group <code>SLOT 0000</code> <code>cxitool Dev=/dev/cs2.0.1 LogonPass=KeyMgrSlot0,123456</code> <code>Group=SLOT_0000 OutDir=C:\Utimaco\CryptoServer BackupKey</code></p>
----------------	--

Output	<p>Upon successful execution of the command, the location of the backup file along with the backed up keys or key groups are returned.</p> <p>Example 1: <code>C:\Utimaco\CryptoServer\1234_RSA2048.kbk</code> <code>1 key(s) backed up</code></p> <p>Example 2: <code>C:\Utimaco\CryptoServer\SLOT 0000 9.kbk</code> <code>C:\Utimaco\CryptoServer\SLOT_0000_1.kbk</code> <code>C:\Utimaco\CryptoServer\SLOT_0000_9.kbk</code> <code>3 key(s) backed up</code></p>
---------------	---

3.4.6 RestoreKey

A cryptographic key may have been backed up to a keyfile (*.kbk). This keyfile is encrypted with a Master Backup Key (MBK).

If this MBK is available in the u.trust Anchor cHSM, the `cxitool RestoreKey` command can restore the cryptographic key from the keyfile to the internal key database of the u.trust Anchor cHSM, `CXIKEY.db`, or an external keystore. Thus, multiple u.trust Anchor cHSMs can be aligned, for example, to contain the same CA's signature key.

All parameters but the filename of the backup file are optional if the CRYPTOSERVER variable has been set, otherwise, the device parameter is mandatory. The optional parameters can be specified if other attributes than the ones saved in the backup file should be assigned to the restored key. If a cryptographic key with the same `Group`, `Name` and `Spec` parameters is already available in the internal key database or the external keystore, it is overwritten.

The `cxitool RestoreKey` command does not support the `Usage` parameter. See [Usage](#) for details. The usage information that is stored in the keyfile is restored.

Syntax	<code>cxitool [Dev=<dev>] <auth> [Name=<name>] [Group=<group>] [Spec=<spec>] [KeyStoreType=<keystoretype> KeyStoreParam=<keystoreparam>] RestoreKey=<filename></code>
---------------	---

Authentication	<ul style="list-style-type: none"> ▪ User manager, level 2 in user group 7, 20000000: Backing up the global configuration ▪ Security officer (SO, key group manager), level 2 in user group 2, 00000200: Backing up the key group configuration ▪ Key manager, level 2 in user group 1, 00000020, or level 2 in user group 0, 00000002 (default): Backing up a cryptographic key <p>The following applies to the security officer and the key manager: Ensure that the authenticated user is assigned (<code>CXI_GROUP</code> user attribute) to the key group(s) specified by the <code><group></code> parameter. For more information, see Cryptographic Keys and Key Groups. A user is assigned to a key group on creation. Use the <code>csadm ListUser</code> command to verify the key group the user is assigned to. For details, see the u.trust Anchor – csadm Manual. Key groups must not be confused with user groups.</p>
-----------------------	---

Parameter	Description
<code><dev></code>	Device address. This parameter can be omitted if the device address has been set as the CRYPTOSERVER environment variable.

Parameter	Description
<name>	<p>Cryptographic key name that shall be used to store the restored cryptographic key in the u.trust Anchor cHSM's internal database, <code>CXIKEY.db</code>, or in an external keystore.</p> <p>This name does not need to match the cryptographic key name that is part of the name of the kbk file containing the key to be restored. The * and ? wildcards are not supported.</p> <p>If <name> is omitted, the name of the backed up key is assigned to the restored key.</p>
<group>	<p>The key group the restored cryptographic key is assigned to</p> <p>If the key group of the backed up key differs from the Group parameter in the <code>cxitool RestoreKey</code> command, ensure that the user authenticating this command has been assigned to both key groups using wildcards. As an alternative, you can use two users for authenticating this command with one user being assigned to the key group of the backed up cryptographic key and the other user being assigned to the key group of the restored cryptographic key. If these conditions are not fulfilled, a permission denied error message is shown.</p> <p>If the <group> parameter is omitted in the <code>cxitool RestoreKey</code> command, the cryptographic key is automatically assigned to the group stored in the backup file.</p> <p>If the cryptographic key has been backed up without being assigned to any group, the cryptographic key can be restored without being assigned to any group or with being assigned to an arbitrary group. Consider in the latter case that the user performing the restore command must have been assigned to the corresponding key group.</p> <p>If the group name had been available for the key backup, the group name is part of the keyfile name.</p> <p>If <group> is omitted, the key group name of the backed up key is assigned to key group name of the restored key.</p>
<spec>	<p>Cryptographic key specifier that shall be used to store the restored cryptographic key in the u.trust Anchor cHSM's internal database, <code>CXIKEY.db</code>, or the external keystore.</p> <p>This cryptographic key specifier does not need to match the key specifier that is part of the name of the kbk file containing the key to be restored.</p> <p>If <spec> is omitted, the key specifier of the backed up key is assigned to key specifier of the restored key.</p>
<keystoretype>	<p>Type of an external keystore</p> <p>Its value is either SDB (for legacy .sdb files) or ODBC.</p> <p>If <code>KeyStoreType</code> is used, <code>KeyStoreParam</code> must be used as well.</p> <p>If <code>KeyStoreType</code> is used, it must be used before <code>KeyStoreParam</code>.</p> <p>If <code>KeyStoreType</code> is used, this cxitool command only lists the keys in the external keystore. In this case, an unspecified key specifier is shown as empty.</p> <p>If <code>KeyStoreType</code> is not used, this cxitool command only lists keys the u.trust Anchor cHSM's internal key database, <code>CXIKEY.db</code>.</p>

Parameter	Description
<keystoreparam>	<p>Configuration parameter for an external keystore.</p> <p>If <code>KeyStoreType</code> is used, <code>KeyStoreParam</code> must be used as well.</p> <p>If <code>KeyStoreType</code> is used, it must be used before <code>KeyStoreParam</code>.</p> <ul style="list-style-type: none"> If <code>KeyStoreType=SDB</code> <code>KeyStoreParam</code> specifies the path to an external keystore file (<code>/path/to/store.sdb</code>). If <code>KeyStoreType=ODB</code> <code>KeyStoreParam</code> specifies the data source name as defined in format <code>DSN=DATA SOURCE NAME</code> or the complete ODBC connection string.
<filename>	Path and name of the <code>.kbk</code> file the cryptographic key is restored from

Example	<pre>cxitool Dev=/dev/cs2.0.1 LogonPass=KeyMgrSlot0,123456 Group=SL0T_0000 Spec=24 RestoreKey=C: \Utimaco\CryptoServer\SL0T_0000_24.kbk</pre>
----------------	---

Output	Upon successful execution of the command, no output is given.
---------------	---

3.4.7 ExportPubKeyFile

This command exports the public component of a cryptographic key to a keyfile.

Syntax	<pre>cxitool [Dev=<dev>] <auth> [Name=<name>] [Group=<group>] [Spec=<spec>] [KeyStoreType=<keystoretype> KeyStoreParam=<keystoreparam>] ExportPubKeyFile=<filename></pre>
---------------	---

Authentication	<p>This command must be authenticated by one of the following user roles:</p> <ul style="list-style-type: none"> Key manager, level 2 in user group 1, 00000020, or level 2 in user group 0, 00000002 (default): Cryptographic keys Key user, level 2 in user group 0, 00000002: Cryptographic keys <p>Ensure that the authenticated user is assigned (<code>CXI_GROUP</code> user attribute) to the key group(s) specified by the <code><group></code> parameter. For more information, see Cryptographic Keys and Key Groups.</p> <p>A user is assigned to a key group on creation. Use the <code>csadm ListUser</code> command to verify the key group the user is assigned to. For details, see the u.trust Anchor – csadm Manual.</p> <p>Key groups must not be confused with user groups.</p>
-----------------------	--

Parameter	Description
<code><dev></code>	Device address. This parameter can be omitted if the device address has been set as the CRYPTOSERVER environment variable.
<code><name></code>	Cryptographic key name The combination of <code><name></code> , <code><group></code> and <code><spec></code> specifies the cryptographic key to be used for the export. Supported key algorithms: RSA and EC.
<code><group></code>	The key group for cryptographic keys
<code><spec></code>	Cryptographic key specifier
<code><keystoretype></code>	Type of an external keystore Its value is either SDB (for legacy .sdb files) or ODBC. If <code>KeyStoreType</code> is used, <code>KeyStoreParam</code> must be used as well. If <code>KeyStoreType</code> is used, it must be used before <code>KeyStoreParam</code> . If <code>KeyStoreType</code> is used, this cxitool command only lists the keys in the external keystore. In this case, an unspecified key specifier is shown as empty. If <code>KeyStoreType</code> is not used, this cxitool command only lists keys the u.trust Anchor cHSM's internal key database, <code>CXIKEY.db</code> .
<code><keystoreparam></code>	Configuration parameter for an external keystore If <code>KeyStoreType</code> is used, <code>KeyStoreParam</code> must be used as well. If <code>KeyStoreType</code> is used, it must be used before <code>KeyStoreParam</code> . <ul style="list-style-type: none"> If <code>KeyStoreType=SDB</code> <code>KeyStoreParam</code> specifies the path to an external keystore file (<code>/path/to/store.sdb</code>). If <code>KeyStoreType=ODB</code> <code>KeyStoreParam</code> specifies the data source name as defined in format <code>DSN=DATA SOURCE NAME</code> or the complete ODBC connection string.
<code><filename></code>	Path and name of the file the public component of the cryptographic key is exported to. If no path is specified, the file in the current directory is used.

Example	<p>Example 1:</p> <pre>cxitool Dev=/dev/cs2.0.1 LogonPass=KeyUsr,123456 Name=RSA ExportPubKeyFile=ExportPubKeyFileRsa.txt</pre> <p>Example 2:</p> <pre>cxitool Dev=/dev/cs2.0.1 LogonPass=KeyUsrSlot0,123456 Group=SL0T_0000 Spec=6 ExportPubKeyFile=C: \Utlimaco\CryptoServer\ExportPubKeyFile.txt</pre>
----------------	---

Output	<p>Upon successful execution of the command, a success message along with the content of the written file is returned.</p> <p>Example 1:</p> <pre>Wrote public key file ExportPubKeyFileRsa.txt The generated file has the following content: # RSA key MOD=D141C13F1277977BAD60D49D87E7B1A4F35C3BD03A07F16AC36A9F4 8A1E626B88BB2717FDEA7E7B2EC810F810B8F11CF1A2A7142D94DB7DE57 CDFE62F637074CBCF78AAA2251843B03716D6B2CD53111229EA6117FED5 0DF638811892F015F44F77C689FBF8D35B0237FE9F73875C10D471D6FEB EAAC334D60320553EE8EDFE21FA4EC20C44F083A6CE02514EAB43622CB2 36D0EB1803A09C960BF89800386B0B3BC196A479350C619B49B75D45BE4 BE61D55C894902A3755A4A7999F37DC109DA1BD71530682CC67A435495D 6442D88BCB31F0B56DD6952BE277645368C1887236B2709CD0369C5E33B 62409466F148D4C070F7107AA4B274E04204F939ABFB PEXP=010001</pre> <p>Example 2:</p> <pre>Wrote public key file C: \Utimaco\CryptoServer\ExportPubKeyFile.txt The generated file has the following content: # ECC key CURVE=secp192k1 PUB=04C8B2A7955EA1C62C2F06F326D2217608E395D8E52D14BBC243D26 A5B7DE8D963419BD6AF757810E6599AB9459AF5874A</pre>
---------------	---

3.4.8 CopyKeyStore

Copies the src external key store to any other external key store.

Syntax	<pre>cxitool [Dev=<dev>] <auth> SrcKeyStoreType=<srckeystoretype> SrcKeyStoreParam=<srckeystoreparam> KeyStoreType=<keystoretype> KeyStoreParam=<keystoreparam> CopyKeyStore</pre>
---------------	--

Authentication	<p>This command must be authenticated by one of the following user roles:</p> <ul style="list-style-type: none"> ▪ Key manager, level 2 in user group 1, 00000020, or level 2 in user group 0, 00000002 (default) ▪ Key user, level 2 in user group 0, 00000002 <p>Ensure that the authenticated user is assigned (<code>CXI_GROUP</code> user attribute) to the key group(s) specified by the <code><group></code> parameter. For more information, see Cryptographic Keys and Key Groups.</p> <p>A user is assigned to a key group on creation. Use the <code>csadm ListUser</code> command to verify the key group the user is assigned to. For details, see the u.trust Anchor – csadm Manual.</p> <p>Key groups must not be confused with user groups.</p>
-----------------------	--

Parameter	Description
<dev>	Device address. This parameter can be omitted if the device address has been set as the CRYPTOSERVER environment variable.
<auth>	Authentication commands See LogonPass and LogonSign .
<srckeystoretype>	Either SDB (for legacy .sdb files), EKM-SDB (for .sdb files of EKM provider) or ODBC.
<srckeystoreparam>	For KeyStoreType SDB and EKM-SDB: A path to an external key store file: /path/to/store.sdb. For KeyStoreType ODBC: The data source name in the format "DSN=DATA SOURCE NAME" or complete ODBC connection string.
<keystoretype>	Either SDB (for legacy .sdb file) or ODBC
<keystoreparam>	For KeyStoreType SDB: A path to an external key store file: /path/to/store.sdb. For KeyStoreType ODBC: The data source name in the format "DSN=DATA SOURCE NAME" or complete ODBC connection string.

Example	<p>Example 1:</p> <pre>cxitool Dev=/dev/cs2.0.1 LogonPass=KeyUsr,123456 Name=RSA ExportPubKeyFile=ExportPubKeyFileRsa.txt</pre> <p>Example 2:</p> <pre>cxitool Dev=/dev/cs2.0.1 LogonPass=KeyUsrSlot0,123456 Group=SL0T_0000 Spec=6 ExportPubKeyFile=C:\Utimaco\CryptoServer\ExportPubKeyFile.txt</pre>
----------------	---

Output	Upon successful execution of the command, no output is given.
---------------	---

3.4.9 KeyMigration

This command updates CKK_EC keys with curves Edwards25519 or Edwards448 to CKK_EC_EDWARDS.

- The migration is performed for all keys the logged-in user has access to.
- The original keys are deleted after the migration. If there is an error while migrating a key, the original key remains.
- The difference between the original and the migrated keys is only the pre-specified migration fields and timestamp.

- The keys to be migrated might be available either in the internal or in an external keystore.
- The user receives a notification that X of Y keys were successfully migrated. The user can list the keys (`cxitool ListKeys` command) to verify them before and after the command execution.
 - Before the `cxitool KeyMigration` command execution: value `ECDSA` in the `algo` column
 - After the `cxitool KeyMigration` command execution: value `Edwards` in the `algo` column

Syntax	<code>cxitool [Dev=<dev>] <auth> [KeyStoreType=<keystoretype> KeyStoreParam=<keystoreparam>] KeyMigration=<migration type></code>
---------------	---

Authentication	Key manager, level 2 in user group 1, 00000020, or level 2 in user group 0, 00000002 (default)
-----------------------	--

Parameter	Description
<dev>	Device address. This parameter can be omitted if the device address has been set as the CRYPTOSERVER environment variable.
<keystoretype>	Type of an external keystore Its value is either SDB (for legacy .sdb files) or ODBC. If <code>KeyStoreType</code> is used, <code>KeyStoreParam</code> must be used as well. If <code>KeyStoreType</code> is used, it must be used before <code>KeyStoreParam</code> . If <code>KeyStoreType</code> is used, this <code>cxitool</code> command only lists the keys in the external keystore. In this case, an unspecified key specifier is shown as empty. If <code>KeyStoreType</code> is not used, this <code>cxitool</code> command only lists keys the u.trust Anchor's internal key database, <code>CXIQUEY.db</code> .
<keystoreparam>	Configuration parameter for an external keystore If <code>KeyStoreType</code> is used, <code>KeyStoreParam</code> must be used as well. If <code>KeyStoreType</code> is used, it must be used before <code>KeyStoreParam</code> . <ul style="list-style-type: none"> ▪ If <code>KeyStoreType=SDB</code> <code>KeyStoreParam</code> specifies the path to an external keystore file (<code>/path/to/store.sdb</code>). ▪ If <code>KeyStoreType=ODB</code> <code>KeyStoreParam</code> specifies the data source name as defined in format <code>DSN=DATA SOURCE NAME</code> or the complete ODDBC connection string.
<migration type>	Supported migration type(s): <code>ECtoEdwards</code>

Example	<p>Migrate all keys of key type EC to key type EDWARDS that the user KeyMgrSlot0 has access to in the internal key database.</p> <pre>cxitool Dev=/dev/cs2.0.1 LogonPass=KeyMgrSlot0,12345678 KeyMigration=ECtoEdwards</pre>
Output	<p>If one key has been migrated:</p> <pre>1 key(s) have been identified for migration. 1 key(s) have been successfully migrated.</pre> <p>If the logged-in user does not have access to any relevant key, the output is as follows:</p> <pre>No suitable keys found for migration.</pre>

3.5 Certificate Management Commands

3.5.1 ExportP10

This command exports a certificate request for a cryptographic key in ASN.1 encoded PKCS#10 format into a CSR file (certificate signing request).

Syntax	<pre>cxitool [Dev=<dev>] <auth> [Name=<name>] [Group=<group>] [Spec=<spec>] [KeyStoreType=<keystoretype>] KeyStoreParam=<keystoreparam>] [Config=<cfgfile>] [DN=<dn>] [HashAlgo=<hash>] [Padding=<pad>] ExportP10[=<filename>]</pre>
Authentication	<p>This command must be authenticated by a user with level 2 in user group 0 (permission mask: 00000002). This can either be a key manager with default permission mask or a key user. A key manager whose level has been changed to level 2 in user group 1 (00000020) is not allowed.</p> <p>Ensure that the authenticated user is assigned (<code>CXI_GROUP</code> user attribute) to the key group(s) specified by the <code><group></code> parameter. For more information, see Cryptographic Keys and Key Groups.</p> <p>A user is assigned to a key group on creation. Use the <code>csadm ListUser</code> command to verify the key group the user is assigned to. For details, see the u.trust Anchor – csadm Manual.</p> <p>Key groups must not be confused with user groups.</p>
Parameter	Description
<code><dev></code>	Device address. This parameter can be omitted if the device address has been set as the CRYPTOSERVER environment variable.

Parameter	Description
<name>	Cryptographic key name. The combination of <name>, <group> and <spec> specifies the cryptographic key the certificate request shall be exported for. RSA keys and ECDSA keys are supported.
<group>	Key group name for cryptographic keys
<spec>	Cryptographic key specifier
<keystoretype>	Type of an external keystore Its value is either SDB (for legacy .sdb files) or ODBC. If KeyStoreType is used, KeyStoreParam must be used as well. If KeyStoreType is used, it must be used before KeyStoreParam. If KeyStoreType is used, this cxitool command only lists the keys in the external keystore. In this case, an unspecified key specifier is shown as empty. If KeyStoreType is not used, this cxitool command only lists keys the u.trust Anchor cHSM's internal key database, CXIKEY.db.
<keystoreparam>	Configuration parameter for an external keystore If KeyStoreType is used, KeyStoreParam must be used as well. If KeyStoreType is used, it must be used before KeyStoreParam. <ul style="list-style-type: none"> If KeyStoreType=SDB KeyStoreParam specifies the path to an external keystore file (/path/to/store.sdb). If KeyStoreType=ODBC KeyStoreParam specifies the data source name as defined in format DSN=DATA SOURCE NAME or the complete ODBC connection string.

Parameter	Description
<cfgfile>	<p>The absolute path and/or name of the distinguished name configuration file. If no path is specified, the file is searched in the current directory. Either the <code>Config</code> parameter or the <code>DN</code> parameter must be available. The following distinguished name fields are supported:</p> <ul style="list-style-type: none"> ▪ <code>commonName</code> Specifies an identifier of an object. ▪ <code>serialNumber</code> Serial number of the distinguished name. Must be a hexadecimal value without a leading "0x" but with up to 40 digits. ▪ <code>countryName</code> The ISO code consisting of two capital letters for the country where the organization is located, for example, <code>GB</code>, <code>FR</code> or <code>US</code> ▪ <code>localityName</code> Town or city ▪ <code>stateOrProvinceName</code> Province, region, county or state, for example, <code>Sussex</code>, <code>Normandy</code> or <code>New Jersey</code>. Do not use abbreviations here. ▪ <code>organizationName</code> Company name including suffixes such as <code>Inc.</code> or <code>Corp.</code> ▪ <code>organizationalUnitName</code> Department name, for example, <code>HR</code>, <code>Finance</code> or <code>IT</code> ▪ <code>title</code> ▪ <code>description</code> ▪ <code>emailAddress</code> An email address to contact the organization <p>The order of the fields is irrelevant. At least one field must be available. Each field may be available several times.</p> <p>Example 1 of the contents of the configuration file:</p> <pre>commonName=1234_Key25 serialNumber=123 countryName=DE localityName=Aachen stateOrProvinceName=Rhineland organizationName=Utimaco IS GmbH organizationalUnitName=Support title=Support description=Test emailAddress=support@utimaco.com</pre> <p>Example 2:</p>

Parameter	Description
	<p>organizationName=Utimaco IS GmbH CommonName=test</p> <p>The following short distinguished names can be used:</p> <p>CN: CommonName L: localityName C: countryName ST: stateOrProvinceName O: organizationName OU: organizationalUnitName</p> <p>Example 3: O=Utimaco IS GmbH CN=test</p>
<dn>	<p>Sequence of distinguished name fields separated by a comma. Either the <code>Config</code> parameter or the <code>DN</code> parameter must be available. The description of the <code><cfgfile></code> parameter applies to the <code><dn></code> parameter in an analog way. If a distinguished name field value contains a blank, the complete DN value must be set in quotation marks. Example 1: DN="organizationName=Utimaco IS GmbH,CommonName=test"</p> <p>Example 2 (with short distinguished names): DN="O=Utimaco IS GmbH,CN=test"</p>
<hash>	<p>Can be SHA224, SHA256, SHA384, SHA512, SHA3_224, SHA3_256, SHA3_384 and SHA3_512. SHA256 is the default value.</p>
<pad>	<p>The padding is supported for RSA only. The following padding mechanisms are supported: PKCS1 and PSS PKCS1 is the default value.</p>
<filename>	<p>The absolute path and/or name of the certificate request file to be generated. If the file already exists, it is overwritten. If a path is specified, a filename must be specified as well. If no path is specified, the file is created in the current directory. The default value is <code><group>_<name>.csr</code> even if the Spec parameter has been set. The default value is <code>_ .csr</code> if the Spec parameter is specified, the Group parameter is empty (Group=) and the Name parameter is empty (Name=).</p>

Example	<p>Example 1: This command generates a <code>1234_Key25.csr</code> file in the current directory.</p> <pre>cxitool Dev=/dev/cs2.0.1 LogonPass=KeyUsr,123456 Name=Key25 Group=1234 Config=C: \Utimaco\CryptoServer\cert.conf ExportP10</pre> <p>Example 2:</p> <pre>cxitool Dev=/dev/cs2.0.1 LogonPass=KeyUsr,123456 Group=1234 Key=22 Config=my.cfg ExportP10=C: \Utimaco\CryptoServer\request_22.csr</pre>
----------------	---

Output	<p>Upon successful execution of the command, no output is given.</p> <p>If the <code>illegal key usage</code> error message is shown, perform the <code>cxitool KeyInfo</code> command to the cryptographic key specified by the combination of <code><name></code>, <code><group></code> and <code><spec></code> and verify whether the <code>Export</code> property has been set for this key.</p> <p>If the <code>Initial FIPS usage must be set</code> error message is shown, perform the <code>cxitool KeyInfo</code> command to the cryptographic key specified by the combination of <code><name></code>, <code><group></code> and <code><spec></code> and verify that the FIPS usage is set to the value matching the key type and for RSA keys also the padding. For details, see SetFipsUsage.</p>
---------------	---

The result can be verified by using the `openssl` command.

Verifying example 2:

<pre>openssl req -text -inform der -in request_22.csr grep "Subject:"</pre>

3.5.2 ExportCert

If a cryptographic key has a certificate property, the `cxitool ExportCert` command exports a DER-encoded X.509 certificate for this cryptographic key to a certificate file. The cryptographic key may be an RSA key, a DSA key or an ECDSA key. A certificate property for a cryptographic key can be created by performing the `cxitool ImportCert` command.

Syntax	<pre>cxitool [Dev=<dev>] <auth> [Name=<name>] [Group=<group>] [Spec=<spec>] [KeyStoreType=<keystoretype>] KeyStoreParam=<keystoreparam>] ExportCert[=<filename>]</pre>
---------------	--

Authentication	<p>This command must be authenticated by one of the following user roles:</p> <ul style="list-style-type: none"> ▪ Security officer (SO, key group manager), level 2 in user group 2, 00000200 ▪ Key manager, level 2 in user group 1, 00000020, or level 2 in user group 0, 00000002 (default) ▪ Key user, level 2 in user group 0, 00000002 <p>Ensure that the authenticated user is assigned (<code>CXI_GROUP</code> user attribute) to the key group(s) specified by the <code><group></code> parameter. For more information, see Cryptographic Keys and Key Groups.</p> <p>A user is assigned to a key group on creation. Use the <code>csadm ListUser</code> command to verify the key group the user is assigned to. For details, see the u.trust Anchor – csadm Manual.</p> <p>Key groups must not be confused with user groups.</p>
-----------------------	---

Parameter	Description
<dev>	Device address. This parameter can be omitted if the device address has been set as the CRYPTOSERVER environment variable.
<name>	Cryptographic key name. The combination of <code><name></code> , <code><group></code> and <code><spec></code> specifies the cryptographic key the certificate request shall be exported for. RSA keys and ECDSA keys are supported.
<group>	Key group name for cryptographic keys
<spec>	Cryptographic key specifier
<keystoretype>	<p>Type of an external keystore Its value is either SDB (for legacy .sdb files) or ODBC.</p> <p>If <code>KeyStoreType</code> is used, <code>KeyStoreParam</code> must be used as well.</p> <p>If <code>KeyStoreType</code> is used, it must be used before <code>KeyStoreParam</code>.</p> <p>If <code>KeyStoreType</code> is used, this cxitool command only lists the keys in the external keystore. In this case, an unspecified key specifier is shown as empty.</p> <p>If <code>KeyStoreType</code> is not used, this cxitool command only lists keys the u.trust Anchor CHSM's internal key database, <code>CXIKEY.db</code>.</p>
<keystoreparam>	<p>Configuration parameter for an external keystore</p> <p>If <code>KeyStoreType</code> is used, <code>KeyStoreParam</code> must be used as well.</p> <p>If <code>KeyStoreType</code> is used, it must be used before <code>KeyStoreParam</code>.</p> <ul style="list-style-type: none"> ▪ If <code>KeyStoreType=SDB</code> <code>KeyStoreParam</code> specifies the path to an external keystore file (<code>/path/to/store.sdb</code>). ▪ If <code>KeyStoreType=ODBC</code> <code>KeyStoreParam</code> specifies the data source name as defined in format <code>DSN=DATA SOURCE NAME</code> or the complete ODBC connection string.

Parameter	Description
<filename>	<p>The absolute path and/or name of the certificate file to be generated. If the file already exists, it is overwritten.</p> <p>If a path is specified, a filename must be specified as well. If no path is specified, the file is created in the current directory.</p> <p>The default value is <code><group>_<name>.der</code> even if the Spec parameter has been set.</p> <p>The default value is <code>_ .der</code> if the Spec parameter is specified, the Group parameter is empty (<code>Group=</code>) and the Name parameter is empty (<code>Name=</code>).</p>
Example	<pre>cxitool Dev=/dev/cs2.0.1 LogonPass=KeyUsrAllGroups,123456 Group=1234 Name=Key35 ExportCert=1234_Key35.der</pre>
Output	Upon successful execution of the command, no output is given.

3.5.3 ImportCert

This command imports an X.509 certificate for an RSA key, a DSA key or an ECDSA key from a certificate file. DER-encoded and BER-encoded binary X.509 certificates, that is, `*.crt` files, `*.cer` files and `*.der` files, are supported.

The certificate file contains the following data:

- Identification information

For example, with the following parameters: `commonName`, `serialNumber`, `countryName`, `localityName`, `stateOrProvinceName`, `organizationName`, `organizationalUnitName`, `title`, `description`, `emailAddress`

- Public key
- Signature/Certificate

This signature has been done over the rest of the file, that is, the identification information and the public key.

The public key that is stored in the certificate and that is certified must be the same that is specified implicitly by the combination of `Name`, `Group` and `Spec`.

Syntax	<code>cxitool [Dev=<dev>] <auth> [Name=<name>] [Group=<group>] [Spec=<spec>] [KeyStoreType=<keystoretype> KeyStoreParam=<keystoreparam>] ImportCert=<filename></code>
---------------	---

Authentication	<p>This command must be authenticated by a key manager. His level may either be 2 in user group 0 (permission mask: 00000002, default value) or the level has been changed to level 2 in user group 1 (00000020).</p> <p>Ensure that the authenticated user is assigned (<code>CXI_GROUP</code> user attribute) to the key group(s) specified by the <code><group></code> parameter. For more information, see Cryptographic Keys and Key Groups.</p> <p>A user is assigned to a key group on creation. Use the <code>csadm ListUser</code> command to verify the key group the user is assigned to. For details, see the u.trust Anchor – csadm Manual.</p> <p>Key groups must not be confused with user groups.</p>
-----------------------	--

Parameter	Description
<code><dev></code>	Device address. This parameter can be omitted if the device address has been set as the CRYPTOSERVER environment variable.
<code><name></code>	Cryptographic key name. The combination of <code><name></code> , <code><group></code> and <code><spec></code> specifies the cryptographic key the certificate request shall be exported for. RSA keys and ECDSA keys are supported.
<code><group></code>	Key group name for cryptographic keys
<code><spec></code>	Cryptographic key specifier
<code><keystoretype></code>	<p>Type of an external keystore Its value is either SDB (for legacy .sdb files) or ODBC.</p> <p>If <code>KeyStoreType</code> is used, <code>KeyStoreParam</code> must be used as well.</p> <p>If <code>KeyStoreType</code> is used, it must be used before <code>KeyStoreParam</code>.</p> <p>If <code>KeyStoreType</code> is used, this cxitool command only lists the keys in the external keystore. In this case, an unspecified key specifier is shown as empty.</p> <p>If <code>KeyStoreType</code> is not used, this cxitool command only lists keys the u.trust Anchor CHSM's internal key database, <code>CXIKEY.db</code>.</p>
<code><keystoreparam></code>	<p>Configuration parameter for an external keystore</p> <p>If <code>KeyStoreType</code> is used, <code>KeyStoreParam</code> must be used as well.</p> <p>If <code>KeyStoreType</code> is used, it must be used before <code>KeyStoreParam</code>.</p> <ul style="list-style-type: none"> If <code>KeyStoreType=SDB</code> <code>KeyStoreParam</code> specifies the path to an external keystore file (<code>/path/to/store.sdb</code>). If <code>KeyStoreType=ODB</code> <code>KeyStoreParam</code> specifies the data source name as defined in format <code>DSN=DATA SOURCE NAME</code> or the complete ODBC connection string.

Parameter	Description
<filename>	The absolute path and/or name of the certificate file to be imported. If no path is specified, the file is searched in the current directory. This file is, for example, a self-signed certificate created by the <code>cxitool SelfSignedCert</code> command. *.crt files, *.cer files and *.der files are supported.
Example	<code>cxitool Dev=/dev/cs2.0.1 LogonPass=KeyMgr,123456 Group=1234 Name=Key25 ImportCert=C:\Utlimaco\CryptoServer\1234_Key25.crt</code>
Output	Upon successful execution of the command, no output is given. Once the certificate has been imported, the <code>cxitool KeyInfo</code> command can be used to show the contents of the certificate.

3.5.4 ImportP12

This command imports a key from a PKCS#12 (*.p12 or *.pfx) or PKCS#8 (*.p8) encoded keyfile. A PKCS#12 keyfile stores cryptography objects as a single file, for example, a private key and its X.509 certificate. A PKCS#8 keyfile stores private key information.

The `cxitool ImportP12` command does not support the `Usage` parameter.

Syntax	<code>cxitool [Dev=<dev>] <auth> [Name=<name>] [Group=<group>] [Spec=<spec>] [Export=<export>] [KeyStoreType=<keystoretype>] KeyStoreParam=<keystoreparam>] ImportP12=<filename>,<password></code>
Authentication	<p>This command must be authenticated by a key user (level 2 in user group 0, permission mask: 00000002).</p> <p>Ensure that the authenticated user is assigned (CXI_GROUP user attribute) to the key group(s) specified by the <group> parameter. For more information, see Cryptographic Keys and Key Groups.</p> <p>A user is assigned to a key group on creation. Use the <code>csadm ListUser</code> command to verify the key group the user is assigned to. For details, see the u.trust Anchor – csadm Manual.</p> <p>Key groups must not be confused with user groups.</p>
Parameter	Description
<dev>	Device address. This parameter can be omitted if the device address has been set as the CRYPTOSERVER environment variable.

Parameter	Description
<name>	Cryptographic key name. The combination of <name>, <group> and <spec> specifies the cryptographic key the certificate request shall be exported for. RSA keys and ECDSA keys are supported.
<group>	Key group name for cryptographic keys
<spec>	Cryptographic key specifier
<export>	This parameter defines whether the key to be imported can be exported or not. See Export for details. <ul style="list-style-type: none"> deny Exporting the key is not allowed (Default), corresponds to 0x00000000. allow Exporting the key is allowed in encrypted form, corresponds to 0x00000001.
<keystoretype>	Type of an external keystore Its value is either SDB (for legacy .sdb files) or ODBC . If KeyStoreType is used, KeyStoreParam must be used as well. If KeyStoreType is used, it must be used before KeyStoreParam . If KeyStoreType is used, this cxitool command only lists the keys in the external keystore. In this case, an unspecified key specifier is shown as empty. If KeyStoreType is not used, this cxitool command only lists keys the u.trust Anchor CHSM's internal key database, CXIKEY.db .
<keystoreparam>	Configuration parameter for an external keystore If KeyStoreType is used, KeyStoreParam must be used as well. If KeyStoreType is used, it must be used before KeyStoreParam . <ul style="list-style-type: none"> If KeyStoreType=SDB KeyStoreParam specifies the path to an external keystore file (/path/to/store.sdb). If KeyStoreType=ODBC KeyStoreParam specifies the data source name as defined in format DSN=DATA SOURCE NAME or the complete ODBC connection string.
<filename>	The PKCS#12 (*.p12 or *.pfx) or PKCS#8 (*.p8) encoded keyfile to be imported
<password>	Password for decrypting the keyfile. For hidden password entry, specify the string ask instead of the password.

Example	<code>cxitool Dev=/dev/cs2.0.1 LogonPass=KeyUsr,123456 Group=1234 Name=Key25 ImportP12=1411134668.p12,test</code>
----------------	---

Output	Upon successful execution of the command, no output is given.
---------------	---

3.5.5 SelfSignedCert

This command creates and exports a self-signed X.509 certificate in DER format for an RSA key or an ECDSA key.

Syntax	<pre>cxitool [Dev=<dev>] <auth> [Name=<name>] [Group=<group>] [Spec=<spec>] [KeyStoreType=<keystoretype>] KeyStoreParam=<keystoreparam>] [Config=<cfgfile>] [DN=<dn>] [HashAlgo=<hash>] [Padding=<pad>] [KeyUsage=<keyusage>] SelfSignedCert=<serialnumber>,<validfrom>,<validto>[,<filename>]</pre>
---------------	--

Authentication	<p>This command must be authenticated by a user with level 2 in user group 0 (permission mask: 00000002). This can either be a key manager with default permission mask or a key user. A key manager whose level has been changed to level 2 in user group 1 (00000020) is not able to generate a self-signed certificate.</p> <p>Ensure that the authenticated user is assigned (<code>CXI_GROUP</code> user attribute) to the key group(s) specified by the <code><group></code> parameter. For more information, see Cryptographic Keys and Key Groups.</p> <p>A user is assigned to a key group on creation. Use the <code>csadm ListUser</code> command to verify the key group the user is assigned to. For details, see the u.trust Anchor – csadm Manual.</p> <p>Key groups must not be confused with user groups.</p>
-----------------------	---

Parameter	Description
<dev>	Device address. This parameter can be omitted if the device address has been set as the CRYPTOSERVER environment variable.
<name>	Cryptographic key name. The combination of <code><name></code> , <code><group></code> and <code><spec></code> specifies the cryptographic key the certificate request shall be exported for. RSA keys and ECDSA keys are supported.
<group>	Key group name for cryptographic keys
<spec>	Cryptographic key specifier
<keystoretype>	<p>Type of an external keystore Its value is either SDB (for legacy .sdb files) or ODBC.</p> <p>If <code>KeyStoreType</code> is used, <code>KeyStoreParam</code> must be used as well.</p> <p>If <code>KeyStoreType</code> is used, it must be used before <code>KeyStoreParam</code>.</p> <p>If <code>KeyStoreType</code> is used, this cxitool command only lists the keys in the external keystore. In this case, an unspecified key specifier is shown as empty.</p> <p>If <code>KeyStoreType</code> is not used, this cxitool command only lists keys the u.trust Anchor cHSM's internal key database, <code>CXIKEY.db</code>.</p>

Parameter	Description
<keystoreparam>	<p>Configuration parameter for an external keystore</p> <p>If <code>KeyStoreType</code> is used, <code>KeyStoreParam</code> must be used as well.</p> <p>If <code>KeyStoreType</code> is used, it must be used before <code>KeyStoreParam</code>.</p> <ul style="list-style-type: none">▪ If <code>KeyStoreType=SDB</code> <code>KeyStoreParam</code> specifies the path to an external keystore file (<code>/path/to/store.sdb</code>).▪ If <code>KeyStoreType=ODBC</code> <code>KeyStoreParam</code> specifies the data source name as defined in format <code>DSN=DATA SOURCE NAME</code> or the complete ODBC connection string.

Parameter	Description
<cfgfile>	<p>The absolute path and/or name of the distinguished name configuration file. If no path is specified, the file is searched in the current directory. Either the <code>Config</code> parameter or the <code>DN</code> parameter must be available. The following distinguished name fields are supported:</p> <ul style="list-style-type: none"> ▪ <code>commonName</code> Specifies an identifier of an object. ▪ <code>serialNumber</code> Serial number of the distinguished name. Must be a hexadecimal value without a leading "0x" but with up to 40 digits. ▪ <code>countryName</code> The ISO code consisting of two capital letters for the country where the organization is located, for example, <code>GB</code>, <code>FR</code> or <code>US</code> ▪ <code>localityName</code> Town or city ▪ <code>stateOrProvinceName</code> Province, region, county or state, for example, <code>Sussex</code>, <code>Normandy</code> or <code>New Jersey</code>. Do not use abbreviations here. ▪ <code>organizationName</code> Company name including suffixes such as <code>Inc.</code> or <code>Corp.</code> ▪ <code>organizationalUnitName</code> Department name, for example, <code>HR</code>, <code>Finance</code> or <code>IT</code> ▪ <code>title</code> ▪ <code>description</code> ▪ <code>emailAddress</code> An email address to contact the organization <p>The order of the fields is irrelevant. At least one field must be available. Each field may be available several times.</p> <p>Example 1 of the contents of the configuration file:</p> <pre>commonName=MyCASSigningKey_01 serialNumber=123 countryName=DE localityName=Aachen stateOrProvinceName=Rhineland organizationName=Utimaco IS GmbH organizationalUnitName=Support title=Support description=Test emailAddress=support@utimaco.com</pre> <p>Example 2:</p>

Parameter	Description
	<p>organizationName=Utimaco IS GmbH CommonName=test</p> <p>The following short distinguished names can be used:</p> <p>CN: CommonName L: localityName C: countryName ST: stateOrProvinceName O: organizationName OU: organizationalUnitName</p> <p>Example 3: O=Utimaco IS GmbH CN=test</p>
<dn>	<p>Sequence of distinguished name fields separated by a comma. Either the <code>Config</code> parameter or the <code>DN</code> parameter must be available. The description of the <code><cfgfile></code> parameter applies to the <code><dn></code> parameter in an analog way. If a distinguished name field value contains a blank, the complete DN value must be set in quotation marks. Example 1: DN="organizationName=Utimaco IS GmbH,CommonName=test"</p> <p>Example 2 (with short distinguished names): DN="O=Utimaco IS GmbH,CN=test"</p>
<hash>	<p>hash can be SHA224, SHA256, SHA384, SHA512, SHA3_224, SHA3_256, SHA3_384 and SHA3_512. SHA256 is the default value except for DSA keys, for which only SHA1 is possible.</p>
<pad>	<p>The padding is supported for RSA only. The following padding mechanisms are supported: PKCS1 and PSS PKCS1 is the default value.</p>

Parameter	Description
<keyusage>	<p>Limits the scope of application of the certificate, that means that the application that verifies the certificate and the signature based on the certification key should also verify whether the <code>KeyUsage</code> parameter allows the scope of application. <code>KeyUsage</code>, an X.509 parameter of the <code>cxitool SelfSignedCert</code> command, must not be confused with <code>Usage</code>, a <code>cxitool</code> proprietary parameter of the <code>cxitool GenerateKey</code> command. <code>KeyUsage</code> limits the scope of application of the certificate, and <code>Usage</code> limits the scope of application of the corresponding key. For details, see Usage.</p> <p>The <code>KeyUsage</code> parameter must not be confused with the <code>cxitool SetFipsUsage</code> command either.</p> <p>If the u.trust Anchor cHSM is in FIPS mode, the <code>cxitool SetFipsUsage</code> command must be performed in addition to the <code>Usage</code> parameter before a cryptographic key can perform any cryptographic operation. The FIPS usage set by the <code>cxitool SetFipsUsage</code> command must correspond to the value of the <code>Usage</code> parameter. For details, see SetFipsUsage.</p> <p><code>KeyUsage</code> is a comma-separated list with the following possible values.</p> <ul style="list-style-type: none"> ▪ <code>digitalsignature</code> ▪ <code>nonrepudiation</code> ▪ <code>keyencipherment</code> ▪ <code>dataencipherment</code> ▪ <code>keyagreement</code> ▪ <code>keycertsign</code> ▪ <code>crlsign</code> ▪ <code>encipheronly</code> ▪ <code>decipheronly</code> <p>For details, see https://tools.ietf.org/html/rfc5280#section-4.2.1.3. <code>keycertsign</code> is the default value.</p>
<serialnumber>	<p>Serial number of the certificate.</p> <p>Must be a hexadecimal value without a leading "0x" but with up to 40 digits.</p>
<validfrom>	<p>Start date of the certificate's validity period</p> <p>Format: <code>YYMMDD</code></p>
<validto>	<p>Expiration date of the certificate's validity period</p> <p>Format: <code>YYMMDD</code></p>

Parameter	Description
<filename>	<p>The absolute path and/or name of the certificate file to be generated. If the file already exists, it is overwritten.</p> <p>If a path is specified, a filename must be specified as well. If no path is specified, the file is created in the current directory.</p> <p>The default value is < group>_<name>.cert even if the Spec parameter has been set.</p> <p>The default value is <code>_ .cert</code> if the Spec parameter is specified, the Group parameter is empty (<code>Group=</code>) and the Name parameter is empty (<code>Name=</code>).</p>
Example	<pre>cxitool Dev=/dev/cs2.0.1 LogonPass=KeyUsr,123456 Name=Key25 Group=1234 Config=C:\Utimaco\CryptoServer\cert.conf SelfSignedCert=123,171205,491231,C: \Utimaco\CryptoServer\1234_Key25.crt</pre>
Output	<p>Upon successful execution of the command, no output is given.</p> <p>If the <code>illegal key usage</code> error message is shown, perform the <code>cxitool KeyInfo</code> command to the cryptographic key specified by the combination of <name>, <group> and <spec> and verify whether the <code>Sign</code> property has been set for this key.</p> <p>If the <code>Initial FIPS usage must be set</code> error message is shown, perform the <code>cxitool KeyInfo</code> command to the cryptographic key specified by the combination of <name>, <group> and <spec> and verify that the FIPS usage is set to the value matching the key type and for RSA keys also the padding. For details, see SetFipsUsage.</p> <p>The new certificate is created in a file, not in u.trust Anchor cHSM 's internal key database, <code>CXIKEY.db</code>. Use, for example, OpenSSL, to show the contents of the certificate. As an alternative, proceed as follows: If the created certificate is imported for a cryptographic key by using the <code>cxitool ImportCert</code> command, the <code>cxitool KeyInfo</code> command can be used to show the contents of the certificate.</p>

The result can be verified by using the `openssl` command.

Verifying the example:

```
openssl x509 -text -inform der -in 1234_Key25.crt | grep "Subject:"
```

3.6 Cryptography Commands

3.6.1 Encrypt

This command encrypts the contents of the file specified by the `InFile` parameter. The result is saved in the file specified by the `OutFile` parameter.

Syntax	<code>cxitool [Dev=<dev>] <auth> [Name=<name>] [Group=<group>] [Spec=<spec>] [IV=<iv>] InFile=<infile> OutFile=<outfile> [KeyStoreType=<keystoretype> KeyStoreParam=<keystoreparam>] Encrypt=<mode> , <pad></code>
---------------	--

Authentication	<p>This command must be authenticated by a user with level 2 in user group 0 (permission mask: 00000002). This can either be a key manager with default permission mask or a key user. A key manager whose level has been changed to level 2 in user group 1 (00000020) is not able to encrypt data.</p> <p>Ensure that the authenticated user is assigned (<code>CXI_GROUP</code> user attribute) to the key group(s) specified by the <code><group></code> parameter. For more information, see Cryptographic Keys and Key Groups.</p> <p>A user is assigned to a key group on creation. Use the <code>csadm ListUser</code> command to verify the key group the user is assigned to. For details, see the u.trust Anchor – csadm Manual.</p> <p>Key groups must not be confused with user groups.</p>
-----------------------	---

Parameter	Description
<code><dev></code>	Device address. This parameter can be omitted if the device address has been set as the CRYPTOSERVER environment variable.
<code><name></code>	Cryptographic key name. The combination of <code><name></code> , <code><group></code> and <code><spec></code> specifies the cryptographic key the certificate request shall be exported for. At least one of these parameters is mandatory. Only AES and DES cryptographic keys are supported.
<code><group></code>	Key group name for cryptographic keys
<code><spec></code>	Cryptographic key specifier
<code><iv></code>	Initialization vector for the encryption The initialization vector is only relevant if <code><mode></code> is equal to <code>CBC</code> . If the combination of <code><name></code> , <code><group></code> and <code><spec></code> specifies a DES key, the initialization vector has a size of 8 byte (16 hexadecimal digits), and if the combination of <code><name></code> , <code><group></code> and <code><spec></code> specifies an AES key, the initialization vector has a size of 16 byte (32 hexadecimal digits). Use a hexadecimal value without a leading " 0x " .
<code><infile>>/ <outfile></code>	<code><infile></code> specifies the path and name of the file to be encrypted. <code><outfile></code> specifies the path and name of the encrypted output. If no path is specified, the file in the current directory is used. An already existing file is overwritten.
<code><keystoretype></code>	Type of an external keystore Its value is either <code>SDB</code> (for legacy .sdb files) or <code>ODBC</code> . If <code>KeyStoreType</code> is used, <code>KeyStoreParam</code> must be used as well. If <code>KeyStoreType</code> is used, it must be used before <code>KeyStoreParam</code> . If <code>KeyStoreType</code> is used, this cxitool command only lists the keys in the external keystore. In this case, an unspecified key specifier is shown as empty. If <code>KeyStoreType</code> is not used, this cxitool command only lists keys the u.trust Anchor cHSM's internal key database, <code>CXIKEY.db</code> .

Parameter	Description
<keystoreparam>	<p>Configuration parameter for an external keystore</p> <p>If <code>KeyStoreType</code> is used, <code>KeyStoreParam</code> must be used as well.</p> <p>If <code>KeyStoreType</code> is used, it must be used before <code>KeyStoreParam</code>.</p> <ul style="list-style-type: none"> If <code>KeyStoreType=SDB</code> <code>KeyStoreParam</code> specifies the path to an external keystore file (<code>/path/to/store.sdb</code>). If <code>KeyStoreType=ODBC</code> <code>KeyStoreParam</code> specifies the data source name as defined in format <code>DSN=DATA SOURCE NAME</code> or the complete ODBC connection string.
<mode>	<code>ECB</code> (Electronic codebook chaining) or <code>CBC</code> (Cipher block chaining)
<pad>	<code>PKCS5</code> (Padding according to PKCS#5) or <code>ISO7816</code> (Padding according to ISO 7816)

Example	<p>Example 1:</p> <pre>cxitool Dev=/dev/cs2.0.1 LogonPass=USER,123456 Name=mykey InFile=data.txt OutFile=data.enc Encrypt=CBC,PKCS5</pre> <p>Example 2:</p> <pre>cxitool Dev=/dev/cs2.0.1 LogonPass=USER,123456 Name=mykey IV=BE56E057F20F883E InFile=data.txt OutFile=data.enc Encrypt=CBC,PKCS5</pre> <p>Example 3:</p> <pre>cxitool Dev=/dev/cs2.0.1 LogonPass=KeyUsr,123456 Name=Key25 Group=1234 InFile=C:\Utimaco\CryptoServer\InData.txt OutFile=C:\Utimaco\CryptoServer\OutData.txt Encrypt=ECB,PKCS5</pre>
----------------	---

Output	<p>Upon successful execution of the command, no output is given.</p> <p>If the <code>illegal key usage</code> error message is shown, perform the <code>cxitool KeyInfo</code> command to the cryptographic key specified by the combination of <code><name></code>, <code><group></code> and <code><spec></code> and verify whether the <code>Encrypt</code> property has been set for this key.</p> <p>If the <code>Initial FIPS usage must be set</code> error message is shown, perform the <code>cxitool KeyInfo</code> command to the cryptographic key specified by the combination of <code><name></code>, <code><group></code> and <code><spec></code> and verify that the FIPS usage is set to <code>NO_SIGN</code>. For details, see SetFipsUsage.</p>
---------------	---

3.6.2 Decrypt

This command decrypts the contents of the file specified by the `InFile` parameter. The plaintext result is saved in the file specified by the `OutFile` parameter.

Syntax	<code>cxitool [Dev=<dev>] <auth> [Name=<name>] [Group=<group>] [Spec=<spec>] [IV=<iv>] InFile=<infile> OutFile=<outfile> [KeyStoreType=<keystoretype> KeyStoreParam=<keystoreparam>] Decrypt=<mode> , <pad></code>
---------------	--

Authentication	<p>This command must be authenticated by a user with level 2 in user group 0 (permission mask: 00000002). This can either be a key manager with default permission mask or a key user. A key manager whose level has been changed to level 2 in user group 1 (00000020) is not able to decrypt data.</p> <p>Ensure that the authenticated user is assigned (<code>CXI_GROUP</code> user attribute) to the key group(s) specified by the <code><group></code> parameter. For more information, see Cryptographic Keys and Key Groups.</p> <p>A user is assigned to a key group on creation. Use the <code>csadm ListUser</code> command to verify the key group the user is assigned to. For details, see the u.trust Anchor – csadm Manual.</p> <p>Key groups must not be confused with user groups.</p>
-----------------------	---

Parameter	Description
<code><dev></code>	Device address. This parameter can be omitted if the device address has been set as the CRYPTOSERVER environment variable.
<code><name></code>	<p>Cryptographic key name.</p> <p>The combination of <code><name></code>, <code><group></code> and <code><spec></code> specifies the cryptographic key the certificate request shall be exported for. At least one of these parameters is mandatory. Only AES and DES cryptographic keys are supported.</p>
<code><group></code>	Key group name for cryptographic keys
<code><spec></code>	Cryptographic key specifier
<code><iv></code>	<p>Initialization vector for the decryption</p> <p>The initialization vector is only relevant if <code><mode></code> is equal to CBC. Ensure that you specify the same initialization vector that has been used for encryption as well.</p> <p>If the combination of <code><name></code>, <code><group></code> and <code><spec></code> specifies a DES key, the initialization vector has a size of 8 byte (16 hexadecimal digits), and if the combination of <code><name></code>, <code><group></code> and <code><spec></code> specifies an AES key, the initialization vector has a size of 16 byte (32 hexadecimal digits). Use a hexadecimal value without a leading " 0x ".</p>
<code><infile>>/ <outfile></code>	<p><code><infile></code> specifies the path and name of the file to be decrypted. <code><outfile></code> specifies the path and name of the decrypted output. If no path is specified, the file in the current directory is used.</p>

Parameter	Description
<keystoretype>	<p>Type of an external keystore Its value is either <code>SDB</code> (for legacy <code>.sdb</code> files) or <code>ODBC</code> . If <code>KeyStoreType</code> is used, <code>KeyStoreParam</code> must be used as well. If <code>KeyStoreType</code> is used, it must be used before <code>KeyStoreParam</code> . If <code>KeyStoreType</code> is used, this <code>cxitool</code> command only lists the keys in the external keystore. In this case, an unspecified key specifier is shown as empty. If <code>KeyStoreType</code> is not used, this <code>cxitool</code> command only lists keys the u.trust Anchor CHSM's internal key database, <code>CXIKEY.db</code> .</p>
<keystoreparam>	<p>Configuration parameter for an external keystore If <code>KeyStoreType</code> is used, <code>KeyStoreParam</code> must be used as well. If <code>KeyStoreType</code> is used, it must be used before <code>KeyStoreParam</code> .</p> <ul style="list-style-type: none"> If <code>KeyStoreType=SDB</code> <code>KeyStoreParam</code> specifies the path to an external keystore file (<code>/path/to/store.sdb</code>). If <code>KeyStoreType=ODBC</code> <code>KeyStoreParam</code> specifies the data source name as defined in format <code>DSN=DATA SOURCE NAME</code> or the complete ODBC connection string.
<mode>	<p><code>ECB</code> (Electronic codebook chaining) or <code>CBC</code> (Cipher block chaining) Ensure that you specify the same chaining mode that has been used for encryption as well.</p>
<pad>	<p><code>PKCS5</code> (Padding according to PKCS#5) or <code>ISO7816</code> (Padding according to ISO 7816) Ensure that you specify the same padding mechanism that has been used for encryption as well.</p>

Example	<p>Example 1: <code>cxitool dev=/dev/cs2.0.1 LogonPass=USER,123456 name=mykey InFile=data.enc OutFile=data.txt Decrypt=CBC,PKCS5</code></p> <p>Example 2: <code>cxitool dev=/dev/cs2.0.1 LogonPass=USER,123456 name=mykey IV=BE56E057F20F883E InFile=C:\Utimaco\CryptoServer\data.enc OutFile=C:\Utimaco\CryptoServer\OutData.txt Decrypt=CBC,PKCS5</code></p>
----------------	--

Output	<p>Upon successful execution of the command, no output is given. If the <code>illegal key usage</code> error message is shown, perform the <code>cxitool KeyInfo</code> command to the cryptographic key specified by the combination of <name>, <group> and <spec> and verify whether the <code>Decrypt</code> property has been set for this key. If the <code>Initial FIPS usage must be set</code> error message is shown, perform the <code>cxitool KeyInfo</code> command to the cryptographic key specified by the combination of <name>, <group> and <spec> and verify that the FIPS usage is set to <code>NO_SIGN</code> . For details, see SetFipsUsage.</p>
---------------	--

3.6.3 Hash

This command hashes the file specified by the `InFile` parameter. The output is saved in the file specified by the `Outfile` parameter.

Syntax	<code>cxitool [Dev=<dev>] [<auth>] InFile=<infile> OutFile=<outfile> Hash=<hash>,<hashdev></code>
Authentication	<p>The authentication of this command depends on <hashdev>.</p> <ul style="list-style-type: none"><code>on_host</code> No authentication but a connection to the u.trust Anchor cHSM is needed.<code>on_hsm</code> This command must be authenticated by a user with level 2 in user group 0 (permission mask: 00000002). This can either be a key manager with default permission mask or a key user. A key manager whose level has been changed to level 2 in user group 1 (00000020) is not able to hash. <p>Because no key group can be specified (no Group parameter), the key group of the user role that is used for authentication is irrelevant. Key groups must not be confused with user groups.</p>
Parameter	Description
<dev>	Device address. This parameter can be omitted if the device address has been set as the CRYPTOSERVER environment variable.
<infile>>/ <outfile>	Input data filename or output data filename including or without the absolute path If no path is specified, the current directory of the command line is used for reading the input file or writing to the output file.
<hash>	The following algorithms are supported: SHA224, SHA256, SHA384, SHA512, SHA3_224, SHA3_256, SHA3_384 and SHA3_512
<hashdev>	Hash device The hash can either be performed on the host computer (<code>on_host</code>) or on the u.trust Anchor cHSM (<code>on_hsm</code>). <code>on_host</code> provides higher performance, whereas <code>on_hsm</code> provides security
Example	<pre>cxitool Dev= /dev/cs2.0.1 LogonPass=USER,123456 InFile=data.txt OutFile=data.sha256 Hash=SHA256,on_hsm cxitool Dev= /dev/cs2.0.1 InFile=InData.txt OutFile=OutData.sha256 Hash=SHA256,on_host cxitool dev= /dev/cs2.0.1 InFile=C: \Utimaco\CryptoServer\InData.txt OutFile=C: \Utimaco\CryptoServer\OutData.sha256 Hash=SHA256,on_host</pre>

Output	Upon successful execution of the command, no output is given.
---------------	---

3.6.4 Sign

This command generates a signature for the file specified by the `InFile` parameter. The signature is saved in the file specified by the `Signature` parameter.

Syntax	<code>cxitool [Dev=<dev>] <auth> [Name=<name>] [Group=<group>] [Spec=<spec>] [KeyStoreType=<keystoretype> KeyStoreParam=<keystoreparam>] InFile=<infile> Signature=<sigfile>,<fmt> Sign=<hash>,<hashdev>[,<pad>]</code>
---------------	---

Authentication	<p>The authentication of this command depends on <code><hashdev></code>.</p> <ul style="list-style-type: none"> ▪ <code>on_host</code> No authentication but a connection to the u.trust Anchor cHSM is needed. ▪ <code>on_hsm</code> This command must be authenticated by a user with level 2 in user group 0 (permission mask: 00000002). This can either be a key manager with default permission mask or a key user. A key manager whose level has been changed to level 2 in user group 1 (00000020) is not able to sign data. <p>Ensure that the authenticated user is assigned (<code>CXI_GROUP</code> user attribute) to the key group(s) specified by the <code><group></code> parameter. For more information, see Cryptographic Keys and Key Groups. Key groups must not be confused with user groups.</p>
-----------------------	---

Parameter	Description
<code><dev></code>	Device address. This parameter can be omitted if the device address has been set as the CRYPTOSERVER environment variable.
<code><name></code>	Cryptographic key name. The combination of <code><name></code> , <code><group></code> and <code><spec></code> specifies the cryptographic key the certificate request shall be exported for. At least one of these parameters is mandatory. Only RSA and ECDSA cryptographic keys are supported.
<code><group></code>	Key group name for cryptographic keys
<code><spec></code>	Cryptographic key specifier

Parameter	Description
<keystoretype>	<p>Type of an external keystore Its value is either SDB (for legacy .sdb files) or ODBC. If <code>KeyStoreType</code> is used, <code>KeyStoreParam</code> must be used as well. If <code>KeyStoreType</code> is used, it must be used before <code>KeyStoreParam</code>. If <code>KeyStoreType</code> is used, this cxitool command only lists the keys in the external keystore. In this case, an unspecified key specifier is shown as empty. If <code>KeyStoreType</code> is not used, this cxitool command only lists keys the u.trust Anchor cHSM's internal key database, <code>CXIKEY.db</code>.</p>
<keystoreparam>	<p>Configuration parameter for an external keystore If <code>KeyStoreType</code> is used, <code>KeyStoreParam</code> must be used as well. If <code>KeyStoreType</code> is used, it must be used before <code>KeyStoreParam</code>.</p> <ul style="list-style-type: none"> If <code>KeyStoreType=SDB</code> <code>KeyStoreParam</code> specifies the path to an external keystore file (<code>/path/to/store.sdb</code>). If <code>KeyStoreType=ODB</code> <code>KeyStoreParam</code> specifies the data source name as defined in format <code>DSN=DATA SOURCE NAME</code> or the complete ODDBC connection string.
<infile>	The file a signature should be generated for
<sigfile>	The name of the output file containing the signature. The file name ending is arbitrary.
<fmt>	The format of the signature. <code>raw</code> (binary coded) and <code>asn1</code> (binary coded) are supported.
<hash>	<p>Hash algorithm The following algorithms are supported: <code>SHA224</code>, <code>SHA256</code>, <code>SHA384</code>, <code>SHA512</code>, <code>SHA3_224</code>, <code>SHA3_256</code>, <code>SHA3_384</code> and <code>SHA3_512</code>.</p>
<hashdev>	<p>Hash device The hash can either be performed on the host computer (<code>on_host</code>) or on the u.trust Anchor cHSM (<code>on_hsm</code>). <code>on_host</code> provides higher performance, whereas <code>on_hsm</code> provides security.</p>
<pad>	<p>Padding is supported for RSA only. For RSA, padding is mandatory. The following padding mechanisms are supported: <code>PKCS1</code> and <code>PSS</code>.</p>

Example	<p>RSA examples:</p> <pre>cxitool Dev=/dev/cs2.0.1 LogonPass=USER,123456 Name=myRSAkey InFile=data.txt Signature=signature.sig,raw Sign=SHA256,on_hsm,PKCS1 cxitool Dev=/dev/cs2.0.1 LogonPass=KeyUsr,123456 Name=RSA2048 InFile=C:\Utimaco\CryptoServer\InDataSign.txt Signature=C: \Utimaco\CryptoServer\OutDataSignRsaAsn1.sig,asn1 Sign=SHA256,on_hsm,PKCS1</pre> <p>ECDSA example:</p> <pre>cxitool dev=/dev/cs2.0.1 LogonPass=USER,123456 Name=myECDSAkey InFile=data.txt Signature=signature.sig,asn1 Sign=SHA256,on_hsm</pre>
----------------	---

Output	<p>Upon successful execution of the command, no output is given.</p> <p>If the <code>illegal key usage</code> error message is shown, perform the <code>cxitool KeyInfo</code> command to the cryptographic key specified by the combination of <code><name></code>, <code><group></code> and <code><spec></code> and verify whether the <code>Sign</code> property has been set for this key.</p> <p>If the <code>Initial FIPS usage must be set</code> error message is shown, perform the <code>cxitool KeyInfo</code> command to the cryptographic key specified by the combination of <code><name></code>, <code><group></code> and <code><spec></code> and verify that the FIPS usage is set to the value matching the key type and for RSA keys also the padding. For details, see SetFipsUsage.</p>
---------------	---

3.6.5 Verify

This command verifies a signature. Verifying a signature does not produce any output file but a success message or a failure message.

Syntax	<pre>cxitool [Dev=<dev>] <auth> [Name=<name>] [Group=<group>] [Spec=<spec>] [KeyStoreType=<keystoretype> KeyStoreParam=<keystoreparam>] InFile=<infile> Signature=<sigfile>,<fmt> Verify=<hash>,<hashdev>[,<pad>]</pre>
---------------	---

Authentication	<p>The authentication of this command depends on <code>< hashdev ></code>.</p> <ul style="list-style-type: none"> ▪ <code>on_host</code> No authentication but a connection to the u.trust Anchor cHSM is needed. ▪ <code>on_hsm</code> This command must be authenticated by a user with level 2 in user group 0 (permission mask: 00000002). This can either be a key manager with default permission mask or a key user. A key manager whose level has been changed to level 2 in user group 1 (00000020) is not able to verify data. <p>Ensure that the authenticated user is assigned (<code>CXI_GROUP</code> user attribute) to the key group(s) specified by the <code><group></code> parameter. For more information, see Cryptographic Keys and Key Groups.</p> <p>A user is assigned to a key group on creation. Use the <code>csadm ListUser</code> command to verify the key group the user is assigned to. For details, see the u.trust Anchor – <code>csadm</code> Manual.</p> <p>Key groups must not be confused with user groups.</p>
-----------------------	--

Parameter	Description
<dev>	Device address. This parameter can be omitted if the device address has been set as the CRYPTOSERVER environment variable.
<name>	<p>Cryptographic key name.</p> <p>The combination of <code><name></code>, <code><group></code> and <code><spec></code> specifies the cryptographic key the certificate request shall be exported for. At least one of these parameters is mandatory.</p> <p>Ensure that you specify the public key corresponding to the private key used for creating the signature. Only RSA and ECDSA cryptographic keys are supported.</p>
<group>	Key group name for cryptographic keys
<spec>	Cryptographic key specifier
<keystoretype>	<p>Type of an external keystore</p> <p>Its value is either <code>SDB</code> (for legacy .sdb files) or <code>ODBC</code>.</p> <p>If <code>KeyStoreType</code> is used, <code>KeyStoreParam</code> must be used as well.</p> <p>If <code>KeyStoreType</code> is used, it must be used before <code>KeyStoreParam</code>.</p> <p>If <code>KeyStoreType</code> is used, this cxitool command only lists the keys in the external keystore. In this case, an unspecified key specifier is shown as empty.</p> <p>If <code>KeyStoreType</code> is not used, this cxitool command only lists keys the u.trust Anchor cHSM's internal key database, <code>CXIKEY.db</code>.</p>

Parameter	Description
<keystoreparam>	<p>Configuration parameter for an external keystore</p> <p>If <code>KeyStoreType</code> is used, <code>KeyStoreParam</code> must be used as well.</p> <p>If <code>KeyStoreType</code> is used, it must be used before <code>KeyStoreParam</code>.</p> <ul style="list-style-type: none"> If <code>KeyStoreType=SDB</code> <code>KeyStoreParam</code> specifies the path to an external keystore file (<code>/path/to/store.sdb</code>). If <code>KeyStoreType=ODB</code> <code>KeyStoreParam</code> specifies the data source name as defined in format <code>DSN=DATA SOURCE NAME</code> or the complete ODBC connection string.
<infile>	The name of the input file containing the data that has been signed. The signature is not included in this file.
<sigfile>	The name of the input file containing the signature
<fmt>	The format of the signature file. <code>raw</code> and <code>asn1</code> are supported. Ensure that you specify the format that has been used for creating the signature.
<hash>	<p>Hash algorithm</p> <p>The following algorithms are supported: <code>SHA224</code>, <code>SHA256</code>, <code>SHA384</code>, <code>SHA512</code>, <code>SHA3_224</code>, <code>SHA3_256</code>, <code>SHA3_384</code> and <code>SHA3_512</code></p> <p>Ensure that you specify the hash algorithm that has been used for creating the signature.</p>
<hashdev>	<p>Hash device</p> <p>The hash can either be performed on the host computer (<code>on_host</code>) or on the u.trust Anchor cHSM (<code>on_hsm</code>). <code>on_host</code> provides higher performance, whereas <code>on_hsm</code> provides security.</p>
<pad>	<p>Padding is supported for RSA only. For RSA, padding is mandatory. The following padding mechanisms are supported: <code>PKCS1</code>, <code>X9.31</code> and <code>PSS</code></p> <p>Ensure that you specify the padding mechanism that has been used for creating the signature.</p>

Example	<p>RSA examples:</p> <pre>cxitool Dev=/dev/cs2.0.1 LogonPass=USER,123456 Name=myRSAkey InFile=data.txt Signature=signature.sig,raw Verify=SHA256,on_hsm,PKCS1 cxitool Dev=/dev/cs2.0.1 LogonPass=KeyUsr,123456 Name=RSA2048 InFile=C:\Utimaco\CryptoServer\InDataSign.txt Signature=C:\Utimaco\CryptoServer\OutDataSignRsaAsn1.sig,asn1 Verify=SHA256,on_hsm,PKCS1</pre> <p>ECDSA example:</p> <pre>cxitool Dev=/dev/cs2.0.1 LogonPass=USER,123456 Name=myECDSAkey InFile=data.txt Signature=signature.sig,asn1 Verify=SHA256,on_hsm</pre>
----------------	--

Output	<p>Upon successful execution of the command, <code>Message Verification successful</code>. Otherwise, <code>Verify= failed</code> on failure.</p> <p>If the <code>illegal key usage</code> error message is shown, perform the <code>cxitool KeyInfo</code> command to the cryptographic key specified by the combination of <code><name></code>, <code><group></code> and <code><spec></code> and verify whether the <code>Sign</code> property has been set for this key.</p> <p>If the <code>Initial FIPS usage must be set</code> error message is shown, perform the <code>cxitool KeyInfo</code> command to the cryptographic key specified by the combination of <code><name></code>, <code><group></code> and <code><spec></code> and verify that the FIPS usage is set to the value matching the key type and for RSA keys also the padding. For details, see SetFipsUsage.</p>
---------------	--

3.7 External Keystore Commands

3.7.1 ConfigODBC

This command is operating system-sensitive.

- Linux
Generates `odbc.ini` and `odbcinst.ini` files, which need to be included as environment variables `$ODBCINI` and `$ODBCINSTINI` in order to connect to a database over an ODBC driver. This overwrites the current `odbc.ini` and `odbcinst.ini` if present in `<inipath>`.
- Windows
Creates all required registry entries for connecting to the external key store database over an ODBC driver.

Syntax	<ul style="list-style-type: none">▪ Linux <code>cxitool ConfigODBC=<dsnname>,<hostname>,<odbcpath>,\<dbname>,<username>,<password>[,<inipath>[,<port>]]</code>▪ Windows <code>cxitool ConfigODBC=<dsnname>,<hostname>,<odbcpath>,\<dbname>,<username>,<password>[,<registry>[,<port>]]</code>
---------------	--

Parameter	Description
<code><dsnname></code>	The data source name for connecting to the database.

Parameter	Description
<hostname>	IP address or server name of the database
<odbcpath>	The name of an installed ODBC driver (All installed ODBC drivers can be retrieved using the <code>cxitoool ListODBCDrivers</code> command). Path to the ODBC driver library.
<dbname>	Name of the database to be connected to
<username>	Name of the user registered in the database
<password>	Authentication of the user for the specified database
<inipath>	Destination path for the generated files <code>odbc.ini</code> and <code>odbcinst.ini</code> . Default value: <code>/etc/odbc</code>
<registry>	<div> <div>USER</div> <div>– Add ODBC registry setting to HKEY_CURRENT_USER (default)</div> </div> <div> <div>SYSTEM</div> <div>– Add ODBC registry setting to HKEY_LOCAL_MACHINE</div> </div>
<port>	Port on which the database service is running in case it is different from the default

Example	<code>cxitoool ConfigODBC="Key Store",192.168.2.124,"PostgreSQL ANSI(x64)",utimaco_db,db_user,utimaco,SYSTEM,5432</code>
----------------	--

3.7.2 CreateDBSchema

This command generates the schema for the specified database using the specified configuration.

Syntax	<code>cxitoool dbConnString=<dbconnstr> CreateDBSchema=<configname>[,<overrideoption>]</code>
---------------	---

Parameter	Description
<dbconnstring>	The data source name as defined in the format <code>"DSN=DATA SOURCE NAME"</code> or the complete ODBC connection string
<configname>	The name of an integrated schema configuration or the name of any configuration file containing a valid schema configuration. The currently integrated configurations are: <ul style="list-style-type: none"> <code>postgres</code> : Generate a postgres compatible schema. <code>mssql</code> : Generate an mssql compatible schema.
<overrideoption>	<ul style="list-style-type: none"> <code>force</code> : Override the old DB schema including the old keys. <code>migrate</code> : Create the new schema and copy the old keys into it.

Example	<code>cxitool dbConnString="DSN=Key Store" CreateDBSchema=postgres</code>
----------------	---

3.7.3 ListODBCDrivers

This command lists all installed ODBC drivers.

Syntax	<code>cxitool ListODBCDrivers</code>
---------------	--------------------------------------

Example	<code>cxitool ListODBCDrivers</code>
----------------	--------------------------------------

Output	<code>SYSTEM: SQL Server %WINDIR%\system32\SQLSRV32.dll (NOT OK)</code>
---------------	---

3.8 FIPS-specific Commands

FIPS-specific commands are always visible in the output of the `cxitool Help` command but they are only executable if the u.trust Anchor cHSM is in FIPS mode.

To verify whether the u.trust Anchor cHSM is in FIPS mode, perform the `csadm GetState` command. The u.trust Anchor cHSM is in FIPS mode, if the output also contains the following line:

```
FIPS mode = ON
```

For details about `csadm`, see the u.trust Anchor - `csadm` Manual.

3.8.1 SetFipsUsage

This command sets the cryptographic usage (either sign/verify or other cryptographic operations). If an RSA key is used for signing/verifying, it sets the padding algorithm for this key.

The `cxitool SetFipsUsage` command must neither be confused with the Usage parameter, see [Usage](#), nor with the `KeyUsage` parameter of the `cxitool SelfSignedCert` command, see [SelfSignedCert](#).

If the u.trust Anchor cHSM is in FIPS mode, the `cxitool SetFipsUsage` command must be performed in addition to the Usage parameter before a cryptographic key can perform any cryptographic operation. The usage set by the `cxitool SetFipsUsage` command must

correspond to the value of the `Usage` parameter. If, for example, a cryptographic key on a u.trust Anchor cHSM in FIPS mode should be used for signing and verifying, the `Usage` parameter (`cxitool ... Usage=SIGN,VERIFY GenerateKey=...`) and the `cxitool SetFipsUsage` command must be performed as follows.

Examples:

```
cxitool ... SetFipsUsage=SIGN
cxitool ... SetFipsUsage=RSA_PKCS
cxitool ... SetFipsUsage=RSA_PSS
cxitool ... SetFipsUsage=RSA_X_9_31
```

If the u.trust Anchor cHSM is not in FIPS mode, the `cxitool SetFipsUsage` command is irrelevant.

If a cryptographic operation, for example, the `cxitool Sign` command, is performed on a u.trust Anchor cHSM in FIPS mode without the FIPS usage being set, the following error message is shown:

```
Error B0680110
```

```
CryptoServer module CXI
```

```
FIPS mode
```

```
initial FIPS usage must be set
```

```
at cxi::Cxi::exec
```



The `cxitool SetFipsUsage` command can be performed only once per cryptographic key. The effect is permanent for the entire lifetime of the cryptographic key.

Syntax

```
cxitool [Dev=<dev>] <auth> [Name=<name>] [Group=<group>]
[Spec=<spec>] [KeyStoreType=<keystoretype>]
KeyStoreParam=<keystoreparam>] SetFipsUsage=<usage flag>
```

Authentication	<p>This command must be authenticated by a user with level 2 in user group 0 (permission mask: 00000002). This can either be a key manager with default permission mask or a key user. A key manager whose level has been changed to level 2 in user group 1 (00000020) is not able to perform the <code>cxitool SetFipsUsage</code> command.</p> <p>Ensure that the authenticated user is assigned (<code>CXI_GROUP</code> user attribute) to the key group(s) specified by the <code><group></code> parameter. For more information, see Cryptographic Keys and Key Groups.</p> <p>A user is assigned to a key group on creation. Use the <code>csadm ListUser</code> command to verify the key group the user is assigned to. For details, see the u.trust Anchor – csadm Manual.</p> <p>Key groups must not be confused with user groups.</p>
-----------------------	--

Parameter	Description
<code><dev></code>	Device address. This parameter can be omitted if the device address has been set as the CRYPTOSERVER environment variable.
<code><name></code>	<p>Cryptographic key name.</p> <p>The combination of <code><name></code>, <code><group></code> and <code><spec></code> specifies the cryptographic key the certificate request shall be exported for. At least one of these parameters is mandatory. Only RSA, DSA and ECDSA cryptographic keys are supported.</p>
<code><group></code>	Key group name for cryptographic keys
<code><spec></code>	Cryptographic key specifier
<code><keystoretype></code>	<p>Type of an external keystore</p> <p>Its value is either SDB (for legacy .sdb files) or ODBC.</p> <p>If <code>KeyStoreType</code> is used, <code>KeyStoreParam</code> must be used as well.</p> <p>If <code>KeyStoreType</code> is used, it must be used before <code>KeyStoreParam</code>.</p> <p>If <code>KeyStoreType</code> is used, this cxitool command only lists the keys in the external keystore. In this case, an unspecified key specifier is shown as empty.</p> <p>If <code>KeyStoreType</code> is not used, this cxitool command only lists keys the u.trust Anchor cHSM's internal key database, <code>CXIKEY.db</code>.</p>
<code><keystoreparam></code>	<p>Configuration parameter for an external keystore</p> <p>If <code>KeyStoreType</code> is used, <code>KeyStoreParam</code> must be used as well.</p> <p>If <code>KeyStoreType</code> is used, it must be used before <code>KeyStoreParam</code>.</p> <ul style="list-style-type: none"> If <code>KeyStoreType=SDB</code> <code>KeyStoreParam</code> specifies the path to an external keystore file (<code>/path/to/store.sdb</code>). If <code>KeyStoreType=ODBC</code> <code>KeyStoreParam</code> specifies the data source name as defined in format <code>DSN=DATA SOURCE NAME</code> or the complete ODBC connection string.

Parameter	Description
<usage flag>	<p>FIPS usage flag Supported values:</p> <ul style="list-style-type: none">▪ NO_SIGN The cryptographic key is used for all cryptographic operations (encrypt, decrypt, etc.) except for signing and verifying. Ensure that the Usage parameter, see Usage, has been set at least to the corresponding value(s), for example, ENCRYPT.▪ SIGN The cryptographic key is used only for signing and verifying. Ensure that the Usage parameter, see Usage, has been set at least to SIGN and VERIFY. Only DSA and ECDSA keys are supported here.▪ RSA_PKCS The cryptographic key is used only for signing and verifying with PKCS1 padding. Ensure that the Usage parameter, see Usage, has been set at least to SIGN and VERIFY. Only RSA keys are supported here.▪ RSA_PSS The cryptographic key is used only for signing and verifying with PKCS1_PSS padding. Ensure that the Usage parameter, see Usage, has been set at least to SIGN and VERIFY. Only RSA keys are supported here.▪ RSA_X_9_31 The cryptographic key is used only for signing and verifying with X9.31 padding. Ensure that the Usage parameter, see Usage, has been set at least to SIGN and VERIFY. Only RSA keys are supported here.

Example	<code>cxitool Dev= /dev/cs2.0.1 LogonPass=USER,123456 Name=myRSAkey SetFipsUsage=RSA_PKCS</code>
----------------	--

Output	Upon successful execution of the command, no output is given. To verify the result, perform the <code>cxitool KeyInfo</code> command.
	Example:
	<code>cxitool Dev=/dev/cs2.0.1 LogonPass=KeyMgr,123456</code>
	<code>Name=myRSAkey KeyInfo</code>
	Example output:
	The FIPS usage is shown below the label.
	Group :
	Name : myRSAkey
	Specifier : -1
	Algo : RSA
	Size : 2048
	Export : 0x00000000
	Extractable : 0
	Sensitive : 1
	Usage : 0x000002bb
	Encrypt : 1
	Decrypt : 1
	Sign : 1
	Verify : 1
	Verify Rec. : 1
	Wrap : 1
	Unwrap : 1
	Derive : 0
	BlockLen : 256
	Type : pub+prv
	DateGen : 20171206111359Z
	DateExp : 20491231235959Z
	Label :
	Fips usage : RSA_PKCS
	Modulus:

3.9 CC-specific Commands

CC-specific commands are irrelevant for a non-CC u.trust Anchor.

3.9.1 SetCCUsage

The `cxitool SetCCUsage` command is irrelevant for a non-CC u.trust Anchor.

4 Built-in Elliptic Curves



You can find all sources given in this table in chapter (1.1.6) 2021-0073 References.

5 Contact Address for Support Queries

You can reach us from Monday to Friday, 09.00 a.m. to 05.00 p.m., Central European Time (CET).

Utimaco IS GmbH
Germanusstr. 4
52080 Aachen
Germany

RMA Query

If you need to send the device back to Utimaco IS GmbH, please open a new RMA case (Return Merchandise Authorization). We request that you use the following web address. RMA cases cannot be opened by email or phone.

<https://support.hsm.utimaco.com/support/rma/new>

Other Support Queries

- Mail (preferred contact method)
support@utimaco.com
Attach the diagnostic information to your email.
- Web portal
<https://support.hsm.utimaco.com/support/cases/new/>
The diagnostic information will be requested in our response if necessary.
- By phone
AMERICAS +1-844-UTIMACO (+1 844-884-6226)
EMEA +49 800-627-3081
APAC +81 800-919-1301
The diagnostic information will be requested in our response if necessary.