THALES

Linux User Guide

FOR CTE V7.6.0

CTE Agent for Windows Advanced Configuration Guide Release

This document covers the following information:

Overview of CTE

For very large data sets, initial encryption deployments can affect data availability, require unacceptable maintenance windows or require cloning and synchronizing data. Encrypting millions of files can span hours or even days, which can delay encryption, or require extra disk space and data synchronization, which can be labor-intensive. Rekeying large data sets can demand significant processing time and lengthy maintenance windows. Security and IT teams face tough tradeoffs, having to choose between security and availability.

CipherTrust Transparent Encryption operates with minimal disruption, effort, and cost. Its transparent approach enables security organizations to implement encryption without changing application, networking, or storage architectures. CipherTrust Live Data Transformation builds on these advantages, offering patented capabilities that deliver breakthroughs in availability, resiliency and efficiency.

CTE includes several unique utilities to help you encrypt and manage your data. It also integrates with several third-party platforms such as Oracle, Teradata Database Appliances, and Amazon S3.

This document describes the installation and advanced configuration options for CTE, as well as detailed information about how to integrate CTE with the supported third-party products.

CTE Terminology

The guide uses the following terminology:

Term	Description
CTE	CipherTrust Transparent Encryption is a suite of products that allow you to encrypt and guard your data. The main software component of CTE is the CTE Agent, which must be installed on every host whose devices you want to protect. **Notes** • This suite was originally called Vormetric Transparent Encryption (VTE), and some of the names in the suite still use "Vormetric". • For example, the default installation directory is /opt/vormetric/DataSecurityExpert/agent/ for Linux and AIX, and C:\Program Files\Vormetric\DataSecurityExpert\agent\ for Windows.
CTE Agent	The software that you install on a physical or virtual machine in order to encrypt and protect the data on that machine. After you have installed the CTE Agent on the machine, you can use CTE to protect any number of devices or directories on that machine.
key manager	An appliance that stores and manages data encryption keys, data access policies, administrative domains, and administrator profiles. Thales offers CipherTrust Manager - a key manager for use with CTE.
host / client	In this documentation, host and client are used interchangeably to refer to the physical or virtual machine on which the CTE Agent is installed.
GuardPoint	A device or directory to which a CTE data protection and encryption policy has been applied. CTE will control access to, and monitor changes in, this device and directory, encrypting new or changed information as needed.

CTE Components

The CTE solution consists of two parts:

- The *CTE Agent software* that resides on each protected virtual or physical machine (host). The CTE Agent performs the required data encryption and enforces the access policies sent to it by the *key manager*. The communication between the CTE Agent and the key manager is encrypted and secure.

  After the CTE Agent has encrypted a device on a host, that device is called a GuardPoint. You can use CTE to create GuardPoints on servers on-site, in the cloud, or a hybrid of both.
- A *key manager* that stores and manages data encryption keys, data access policies, administrative domains, and administrator profiles. After you install the CTE Agent on a host and register it with a key manager, you can use the key manager to specify which devices on the host that you want to protect, what encryption keys are used to protect those devices, and what access policies are enforced on those devices.

Note

For a list of CTE versions and supported operating systems, see the CTE Compatibility Portal.

How to Protect Data with CTE

CTE uses policies created in the associated key manager to protect data. You can create policies to specify file encryption, data access, and auditing on specific directories and drives on your protected hosts. Each GuardPoint must have one and only one associated policy, but each policy can be associated with any number of GuardPoints.

Policies specify:

- Whether or not the resting files are encrypted.
- Who can access decrypted files and when.
- What level of file access auditing is applied when generating fine-grained audit trails.

A Security Administrator accesses key manager through a web browser. You must have administrator privileges to create policies using key Manager. The CTE Agent then implements the policies once they are pushed to the protected host.

CTE can only enforce security and key selection rules on files inside a guarded directory. If a GuardPoint is disabled, access to data in the directory goes undetected and ungoverned. Disabling a GuardPoint and then allowing unrestricted access to that GuardPoint can result in data corruption.

Getting Started with CTE for Linux

This section describes how to install CTE for Linux, register it with your selected key manager, and then create a simple GuardPoint on the protected host. It contains the following topics:

Additional Considerations

The following sections describe some of the things to keep in mind when configuring CTE.

Tracking and Preventing Local User Creation

CTE audits any attempts to change user authentication files. It also allows you to prevent any change to user authentication files using the host settings `protect`. This includes, but is not limited to user creation, modification, and deletion, or to deny users.

- The `audit` setting is set to on by default. It logs access to the system credential files but does not prevent account modifications.

- The `protect` setting both audits and prevents local user account modifications. You must manually enable the `protect` setting for tracking and prevention of local user account creation.

The `protect` tag will prevent changes to the files mentioned below. In the absence of the `protect` tag in host/client settings, operations on these files are permitted. When a log entry is generated, it is tagged with an `[audit]` tag.

- /etc/passwd
- /etc/group
- /etc/shadow
- /etc/gshadow
- /etc/ssh/sshd_config
- /etc/ssh/sshrc

Note

You do not have to restart CTE after applying or removing these host settings.

Restricted Directories

Linux does not allow you to guard the following directories:

- <secfs install root>/agent/secfs/
- /etc
- /tmp
- /usr
- /usr/lib
- /usr/lib/pam
- /var/log/vormetric

Linux does not allow you to guard the following directories and all of their subdirectories:

- <install root>/agent/secfs/bin
- <secfs install root>/agent/vmd
- /etc/vormetric

- /etc/pam.d
- /etc/security
- /usr/lib/security
- /etc/rc*

Restricted Mode

> **Caution**
>
> **If you install or upgrade in restricted mode, you cannot revert to unrestricted mode without uninstalling CTE.**

You can install CTE in restricted mode. This mode prevents any user other than `root` from accessing the following directories:

- /var/log/vormetric
- /opt/vormetric/DataSecurityExpert

Restricted Mode also prevents non-`root` users from running the following utilities:

- agenthealth
- agentinfo
- check_host
- register_host
- secfsd
- vmd
- vmsec
- voradmin

Key Agents and Restricted Mode

- On systems where CTE is installed in restricted mode, you cannot install a key agent (pkcs11) or CipherTrust TDE Key Management.
- On systems where a key agent (pkcs11) or CipherTrust TDE are already installed, you cannot install CTE in restricted mode.

Restricted Mode Installation

To install in restricted mode, use the -r option.

```
./vee-fs-<release>-<build>-<system>.bin -r
```

For example:

```
./vee-fs-7.3.0-135-rh8-x86_64.bin -r
```

RPM Installation

If installing from an RPM directly, prior to installation, type:

```
export VOR_RESTRICTED_INSTALL_MODE=yes
```

Upgrade in Restricted Mode

The upgrade mode is the same as the installation mode.

CTE Agent Installation with UEFI Secure Boot

If you want to install the CTE Agent software on a Linux system that has UEFI Secure Boot enabled, you must first download the appropriate Thales public certificate for your version of CipherTrust Transparent Encryption, and add that certificate to the MOK (Machine Owner Key) list.

Note

- The Thales public certificate is valid for three years from the date of issuance. Six months before the current public certificate is set to expire, Thales will release an advisory, along with the new certificate, that will become valid after the six month grace period expires. All future builds of CipherTrust Transparent Encryption will be built with the new certificate.

Warning

- You can only use the new certificate if the CTE build has been signed with the new certificate. If you are using an older version of CipherTrust Transparent Encryption that was signed with a previous certificate, then you **must** use that certificate with CipherTrust Transparent Encryption.

Public Certificate Naming Convention

The Thales public certificate name is `CTE_Secure_Boot_Cert_MM-DD-YYYY.der`:

Example

```
CTE_Secure_Boot_Cert_05-15-2023.der
```

Getting the Current Public Certificate

There are two options for getting the certificate:

Option 1: Install the DER file

1. Download the CipherTrust Transparent Encryption binaries to your local system.
2. Use the `-e` option to extract the public certificate from the downloaded binary:

```
./<cte-binary-name>.bin -e
```

Example

```
./vee-fs-7.5.0-95-rh8-x86_64.bin -e
```

3. Add the certificate to the MOK list.

Option 2: Convert the PEM file to DER

1. Download the certificate from the Thales public directory CTE_Secure_Boot Repository

```
curl -O https://packages.vormetric.com/pub/CTE_Secure_Boot/<certificate_name>.pem
```

Example

```
curl -O https://packages.vormetric.com/pub/CTE_Secure_Boot/CTE_Secure_Boot_Cert_05-15-2023.pem
```

> **Note**
>
> You can also download it from the Thales Customer Support Portal: KB0027431.

2. Convert the certificate from a PEM file to a DER file using the following command:

```
openssl x509 -inform PEM -outform DER -in <certificate_name>.pem -out <certificate_name>.der
```

Example

```
openssl x509 -inform PEM -outform DER -in
CTE_Secure_Boot_Cert_05-15-2023.pem -out
CTE_Secure_Boot_Cert_05-15-2023.der
```

3. Add the certificate to the MOK list.

Adding the Certificate to the MOK List

Note

During this procedure, you **must** reboot the Linux host and then respond to a system prompt as soon as the host restarts.

1. Log into the host as `root`.
2. Add the certificate to the MOK list:

```
mokutil --import <cert-name>
```

Example

```
mokutil --import CTE_Secure_Boot_Cert_05-15-2023.der
```

3. Enter and confirm a password for this request when prompted.
4. Reboot the host and follow the instructions on the console when the host is back online. You will need to enter the password you created in the previous step when prompted. For detailed information, refer to the specific instructions from each linux distribution.

Note

If you do not respond to the system prompt to update the MOK when the host restarts, the prompt will time out and you will need to run the `mokutil` command again.

5. When prompted, reboot the host again.
6. After the host has rebooted for the second time, verify that the certificate has been properly added to the MOK list:

```
mokutil --test-key <cert-name>
```

Example

```
mokutil --test-key CTE_Secure_Boot_Cert_05-15-2023.der
```

Response

```
CTE_Secure_Boot_Cert_05-15-2023.der is already enrolled.
```

Linux Package Installation for Linux Distributions

- Linux Package Installation for RHEL
- Linux Package Installation for SLES
- Linux Package Installation for Ubuntu

Linux Package Installation for RHEL

This section describes how to access the Linux RPM installation package so that the CTE Agent installation integrates with the distribution software. To access the Linux RPM file, you can:

- Extract the RPM file from the CTE Agent install bin file. This is the easiest method, but the files in the package are not signed and therefore cannot be verified. For details, see Installing the Unsigned RPM Package.
- Download the package from the Yum repository. If you use Yum, the files in the package are signed and the signatures are automatically verified when the package is installed. For details, see Installing the Signed RPM Package with Yum.
- Manually download the RPM package outside of Yum and manually verify the package signatures. For details, see Installing the Signed RPM Package Manually.

Prerequisites

Note

Before you can download and install the package using the Yum repository, you must contact Thales Customer Support to get the username and password for the package repository on which the package resides.

Installing the Unsigned RPM Package

The CTE installation `bin` files contain the unsigned native packages. Extract them by running the `bin` file with the `-e` flag.

1. Log on to the host system as root and copy or mount the installation file to the host system.
2. Extract the RPM file using the following command:

   ```
   ./vee-fs-<release>-<build>-<distro>-<architecture>.bin -e
   ```

   Example

   ```
   ./vee-fs-7.5.0-68-rh8-x86_64.bin -e
   ```

   Response

   ```
   Contents extracted.
   ```

3. Verify that the package extracted correctly:

   ```
   ls *rpm
   ```

   Example Response

   ```
   vee-fs-7.5.0-68-rh8-x86_64.rpm
   ```

4. To start the installation using the RPM file, use the following command:

```
rpm -ivh vee-fs-<release>-<build>-<distro>-<architecture>.rpm
```

5. Follow the prompts to install and register CTE.
   For details about the installation and registration process, see the appropriate installation procedure.
   - **If you are going to register the system with a CipherTrust Manager, see** Configuring CTE for Linux with CipherTrust Manager**.**

Installing the Signed RPM Package with Yum

1. Create the repository file, `/etc/yum.repos.d/Vormetric-VTE.repo`, with the following contents:

```
[vormetric-vte]
releasever=REL_VERSION
name=Vormetric VTE Packages $releasever - $basearch - Source
baseurl=https://USER:PASSWORD@packages.vormetric.com/vte/VERSION/
rhel-$releasever/
gpgkey=https://packages.vormetric.com/pub/PKG-GPG-KEY-vormetric
enabled=1
gpgcheck=1
repo_gpgcheck=1
sslverify=1
```

   where:
   - **REL_VERSION**: RHEL release version. Ex: rhel-8, rhel-9
   - **USER:PASSWORD**: Username/password obtained from Thales Support
   - **VERSION**: CTE release version. Ex: 7.5.0

2. Clean up the Yum cache directory:

```
yum clean all
```

3. List all available versions of CTE:

```
yum list --showduplicates vee-fs
```

4. Use Yum to install the CTE binary. For example:
   If the CTE binary name is vee-fs-7.5.0.68-rh8-x86_64, type:

   ```
   yum install vee-fs-7.5.0.68-rh8-x86_64
   ```

   To install the latest version, type:

   ```
   yum install vee-fs
   ```

   Note

   The first time you install CTE through Yum, you will be asked to import the GPG key. Make sure that you download this key or the install will fail. For example:

   vormetric-vte/7Server/signature | 198 B 00:00:00 Retrieving key from https://packages.vormetric.com/pub/PKG-GPG-KEY-vormetric Importing GPG key 0x628536B7: Userid : "Vormetric (PKG-GPG-KEY) support@vormetric.com" Fingerprint: 7cb5 4f55 40d4 1b63 bf91 c896 f00a 13b0 6285 36b7 From : https://packages.vormetric.com/pub/PKG-GPG-KEY-vormetric Is this ok [y/N]: y

5. Follow the prompts to install and register CTE.
   For details about the installation and registration process, see the appropriate installation procedure.
   - **If you are going to register the system with a CipherTrust Manager, see** Configuring CTE for Linux with CipherTrust Manager**.**

Installing the Signed RPM Package Manually

If you want to manually verify the signed version of the CTE RPM package, you can download the public key from the Thales package repository and manually verify the rpm signature.

1. Download the `rpm` file, type:

```
$ wget -c --user USER --password PASSWORD https://
packages.vormetric.com/vte/VERSION/REL_VERSION/TARGET_RPM_NAME
```

Example

```
$ wget -c --user ctetest --password abcdef12345678ab https://
packages.vormetric.com/vte/7.4.0/rhel-8/vee-fs-7.5.0.68-rh8-
x86_64.rpm
```

where:

- **REL_VERSION**: RHEL release version. Ex: rhel-8, rhel-9
- **USER/PASSWORD**: Username/password obtained from Thales Support
- **VERSION**: CTE release version. Ex: 7.5.0
- **TARGET_RPM_NAME**: Target `rpm` filename to download. Ex: vee-fs-7.5.0-68-rh8-x86_64.rpm

2. Import the public key from the Thales package repository:

```
sudo rpm --import https://packages.vormetric.com/pub/PKG-GPG-KEY-v
ormetric
sudo rpm -qa gpg-pub*
gpg-pubkey-628536b7-56f9887b    : Imported CTE GPG public key.
gpg-pubkey-fd431d51-4ae0493b
```

3. Verify the signature of the package.

```
sudo rpm -Kv vee-fs-<release>-<build>-<distro>-<architecture>.rpm
```

Example

```
sudo rpm -Kv vee-fs-7.5.0-68-rh8-x86_64.rpm
```

4. To start the installation using the RPM file, use the following command:

```
sudo rpm -ivh vee-fs-<release>-<distro>-x86_64.rpm
```

5. Follow the prompts to register CTE.

   For details about the installation and registration process, see the appropriate installation procedure.

   - **If you are going to register the system with a CipherTrust Manager, see** Configuring CTE for Linux with CipherTrust Manager**.**

Linux Package Installation for SLES

This section describes how to access the Linux RPM installation package so that the CTE Agent installation integrates with the distribution software. To access the Linux RPM file, you can:

- Extract the RPM file from the CTE Agent install bin file. This is the easiest method, but the files in the package are not signed and therefore cannot be verified. For details, see Installing the Unsigned RPM Package.
- Download the package from the RPM repository. If you use Zypper Package Manager, the files in the package are signed and the signatures are automatically verified when the package is installed. For details, see Installing the Signed RPM Package with Zypper.
- Manually download the RPM package outside of Zypper and manually verify the package signatures. For details, see Installing the Signed RPM Package Manually.

Prerequisites

Note

Before you can download and install the package using the Zypper repository, you must contact Thales Supportto get the username and password for the package repository on which the package resides.

Installing the Unsigned RPM Package

The CTE installation `bin` files contain the unsigned native packages. Extract them by running the `bin` file with the `-e` flag.

1. Log on to the host system as root and copy or mount the installation file to the host system.

2. Extract the RPM file using the following command:

```
./vee-fs-<release>-<build>-<distro>-<architecture>.bin -e
```

Example

```
./vee-fs-7.5.0-68-sles15-x86_64.bin -e
```

Response

```
Contents extracted.
```

3. Verify that the package extracted correctly:

```
ls *rpm
```

Example Response

```
vee-fs-7.5.0-68-sles15-x86_64.rpm
```

4. To start the installation using the RPM file, use the following command:

```
sudo rpm -ivh vee-fs-<release>-<distro>-x86_64.rpm
```

5. Follow the prompts to install and register CTE.
   For details about the installation and registration process, see the appropriate installation procedure.
   - **If you are going to register the system with a CipherTrust Manager, see** Configuring CTE for Linux with CipherTrust Manager**.**

Installing the Signed RPM Package with Zypper Package Manager

1. Create the repository file `/etc/zypp/repos.d/Vormetric-VTE.repo`, with the following contents:

```
/etc/zypp/repos.d/Vormetric-VTE.repo
```

2. Add with the following contents:

```
[vormetric-vte]

name=Vormetric VTE Packages for SLES

enabled=1

autorefresh=0

baseurl=https://USER:PASSWORD@packages.vormetric.com/vte/VERSION/SLES_VER

type=rpm-md
```

Where:

- **USER/PASSWD**: contact Thales Support to obtain the credential
- **VERSION**: CTE release version. Ex: 7.3.0, 7.4.0
- **SLES_VER**: SLES release version. Ex: sles12, sles15

3. Import public key from the repository:

```
sudo rpm --import https://packages.vormetric.com/pub/PKG-GPG-KEY-vormetric
```

4. Clean up cached data:

```
$ sudo zypper clean

$ sudo zypper refresh
```

5. List available versions of the CTE.

```
$ zypper se --match-exact -t package -s vee-fs
```

6. Install CTE binary:

```
$ zypper in vee-fs                          : download the latest

$ zypper in vee-fs=<CTE_VERSION>            : download the specific ver
sion
```

7. Follow the prompts to install and register CTE.
   For details about the installation and registration process, see the appropriate installation procedure.
   - **If you are going to register the system with a CipherTrust Manager, see** Configuring CTE for Linux with CipherTrust Manager**.**

Installing the Signed RPM Package Manually

If you want to manually verify the signed version of the CTE RPM package, you can download the public key from the Thales package repository and manually verify the rpm signature.

1. Download the `rpm` file, type:

```
$  wget -c --user USER --password PASSWORD  https://
packages.vormetric.com/vte/VERSION/REL_VERSION/TARGET_RPM_NAME
```

   Example

```
$  wget -c --user USER --password PASSWORD  https://
packages.vormetric.com/vte/7.4.0/sles15/vee-fs-7.5.0-68-sles15-
x86_64.rpm
```

   where:
   - **REL_VERSION**: RHEL release version. Ex: sles12, sles15,
   - **USER/PASSWORD**: Username/password obtained from Thales Support
   - **VERSION**: CTE release version. Ex: 7.5.0

- **TARGET_RPM_NAME**: Target `rpm` filename to download. Ex: vee-fs-7.5.0-68-sles15-x86_64.rpm

2. Import the public key from the Thales package repository:

```
    sudo rpm --import https://packages.vormetric.com/pub/PKG-GPG-KEY-vormetric
    sudo rpm -qa gpg-pub*
    gpg-pubkey-628536b7-56f9887b    : Imported CTE GPG public key.
    gpg-pubkey-fd431d51-4ae0493b
```

3. Verify the signature of the package using the `rpm -Kv` command.

```
        sudo rpm -Kv vee-fs-<release>-<build>-<distro>-<architecture>.rpm
```

Example

```
        sudo rpm -Kv vee-fs-7.5.0-68-sles15-x86_64.rpm
```

Response

```
        Header V4 DSA/SHA1 Signature, key ID 628536b7: OK

        Header SHA1 digest: OK (ed3d33dca580c66c70961cfee143e9877a09544c)

        MD5 digest: OK (95273b36ef1c205a7cea444e14bef15f)

        V4 DSA/SHA1 Signature, key ID 628536b7: OK

        The output should show that the keys match is OK.
```

4. To start the installation using the RPM file, use the following command:

```
sudo rpm -ivh vee-fs-<release>-<build>-<distro>-<architecture>.rpm
```

5. Follow the prompts to register CTE.

For details about the installation and registration process, see the appropriate installation procedure.

- **If you are going to register the system with a CipherTrust Manager, see** Configuring CTE for Linux with CipherTrust Manager**.**

Linux Package Installation for Ubuntu

Prerequisites

> **Note**
>
> Before you can download and install the package using the Apt repository, you must contact Thales Supportto get the username and password for the package repository on which the package resides.

Installing the Unsigned DEB Package

The CTE installation `bin` files contain the unsigned native packages. Extract them by running the `bin` file with the `-e` flag.

1. Log on to the host system as root and copy or mount the installation file to the host system.
2. Extract the DEB file using the following command:

   ```
   ./vee-fs-<release>-<build>-<distro>-<architecture>.bin -e
   ```

   Example

   ```
   ./vee-fs-7.5.0-68-ubuntu22-x86_64.bin -e
   ```

   Response

   ```
   Contents extracted.
   ```

3. Verify that the package extracted correctly:

```
ls *deb
```

Example Response

```
vee-fs-7.5.0-68-ubuntu22-x86_64.deb
```

4. To start the installation using the deb file, use the following command:

```
dpkg -i vee-fs-7.5.0-68-ubuntu22-x86_64.deb
```

5. Follow the prompts to install and register CTE.
   For details about the installation and registration process, see the appropriate installation procedure.
   - **If you are going to register the system with a CipherTrust Manager, see** Configuring CTE for Linux with CipherTrust Manager**.**

Installing the Signed DEB Package

1. Create the source list file, `/etc/apt/sources.list.d/Vormetric-VTE.list`, with the following contents:

Ubuntu 18 or subsequent versions

a. Add the following contents from the Vormetric Source repository.

deb https://packages.vormetric.com/vte/VERSION/UBUNTU_VER /

Where:

- **VERSION**: CTE release version. Ex: 7.5.0
- **UBUNTU_VER**: Ubuntu release version. Ex: ubuntu20, ubuntu22

b. Create the login configuration file, type:

```
vi /etc/apt/auth.conf
```

c. Enter the login information, type:

```
    machine packages.vormetric.com/vte/VERSION/UBUNTU_VER login U
SER password PASSWD
```

Where:

- **USER PASSWORD**: Username/password obtained from Thales Support
- **VERSION**: CTE release version. Ex: 7.5.0
- **UBUNTU_VER**: Ubuntu release version. Ex: ubuntu22, ubuntu20

Ubuntu 16

a. Add the following contents from the Vormetric Source repository.

deb https://USER:PASSWD@packages.vormetric.com/vte/VERSION/UBUNTU_VER / `

Where:

- **USER PASSWORD**: Username/password obtained from Thales Support
- **VERSION**: CTE release version. Ex: 7.5.0
- **UBUNTU_VER**: Ubuntu release version. Ex: ubuntu16

2. Import public key from the repository

   $ sudo curl -O https://packages.vormetric.com/pub/PKG-GPG-KEY-vormetric $ sudo gpg --import PKG-GPG-KEY-vormetric $ sudo apt-key add PKG-GPG-KEY-vormetric //add an OpenPGP key to your repository. apt-key will deprecated with Ubuntu 22. $ sudo apt-key update

3. Retrieve the new list of packages.

```
$ sudo apt-get update
```

4. List all versions of the CTE binary:

```
$ apt-cache madison vee-fs
```

5. Install CipherTrust Transparent Encryption.

```
$ sudo apt-get install vee-fs : installs the latest version of CTE
$ sudo apt-get install vee-fs=<release>-<build> : installs the sp
ecific version of CTE
```

6. Follow the prompts to install and register CTE.
   For details about the installation and registration process, see the appropriate installation procedure.
   - **If you are going to register the system with a CipherTrust Manager, see** Configuring CTE for Linux with CipherTrust Manager**.**

Installing the Signed DEB Package Manually

If you want to manually verify the signed version of the CTE DEB package, you can download the public key from the Thales package repository and manually verify the DEB signature.

1. Download the `deb` file, type:

```
$  wget -c --user USER --password PASSWORD  https://
packages.vormetric.com/vte/VERSION/REL_VERSION/TARGET_PKG_NAME
```

   Example

```
$  wget -c --user USER --password PASSWORD  https://
packages.vormetric.com/vte/7.5.0/ubuntu22/vee-fs-7.5.0-68-
ubuntu22-x86_64.deb
```

   where:
   - **REL_VERSION**: RHEL release version. Ex: ubuntu20, ubuntu22
   - **USER/PASSWORD**: Username/password obtained from Thales Support
   - **VERSION**: CTE release version. Ex: 7.5.0
   - **TARGET_PKG_NAME**: Target package filename to download. Ex: vee-fs-7.5.0-68-ubuntu22-x86_64.deb

2. Import the public key from the Thales package repository:

$ sudo gpg --import PKG-GPG-KEY-vormetric $ sudo apt-key add PKG-GPG-KEY-vormetric $ sudo apt-key update

3. Verify the signature of the package.

```
sudo dpkg-sig --verify vee-fs-<release>-<build>-<distro>-<architecture>.deb
```

Example

```
sudo dpkg-sig --verify vee-fs-7.5.0-68-ubuntu22-x86_64.deb
```

Response

```
Processing vee-fs-7.4.0-119-ubuntu20-x86_64.deb...
GOODSIG _gpgvormetric 7CB54F5540D41B63BF91C896F00A13B0628536B7 1698960263
```

4. To start the installation using the deb file, use the following command:

```
sudo dpkg -i vee-fs-7.5.0-68-ubuntu22-x86_64.deb
```

5. Follow the prompts to register CTE.
   For details about the installation and registration process, see the appropriate installation procedure.
   - **If you are going to register the system with a CipherTrust Manager, see** Configuring CTE for Linux with CipherTrust Manager**.**

Verifying Kernel Compatibility

Thales maintains a compatibility matrix in a JSON file that maps all CTE Agent releases to the Operating Systems and kernels that support those releases. The information in this file allows you to verify the compatibility between any Linux host and the version of the CTE Agent that you want to install on that host.

You can view the current compatibility information in CipherTrust Manager by downloading the most recent compatibility JSON file from Thales and then uploading it to CipherTrust Manager. After the JSON file has been uploaded, the CipherTrust

Manager displays the compatibility of all Linux hosts registered with the CipherTrust Manager in the **Compatibility View** on the **Hosts** page.

For details about how to do this in CipherTrust Manager refer to: Kernel Compatibility Matrix.

The following procedure describes how to download the compatibility JSON file and verify its authenticity.

1. Download the `cte_compatibility_matrix.tgz` file from https://packages.vormetric.com/pub/ or from the Thales Customer Support Portal.
2. Extract the files from the TGZ file. The resulting files are:
   - `CTE_Compatibility_Matrix_Cert_mm-dd-yyyy.pem` — The X.509 Public Key Certificate.
   - `cte_compatibility_matrix.json` — The compatibility JSON file.
   - `cte_compatibility_matrix.sign.sha256` — The SHA256 signature for the JSON file.

   For example:

   ```
   tar -xvzf cte_compatibility_matrix.tgz
   ```

   CTE_Compatibility_Matrix_Cert_12-17-2020.pem
   cte_compatibility_matrix.json
   cte_compatibility_matrix.sign.sha256
3. Extract the Public Key from the X.509 Public Key Certificate. For example:

   ```
   openssl x509 -in CTE_Compatibility_Matrix_Cert_12-17-2020.pem -pubkey \
   -noout > cte_compatibility_matrix_public_key.pem
   ```

   The Public Key is in PEM format in the file `cte_compatibility_matrix_public_key.pem`.
4. Verify the SHA signature using the Public Key. For example:

   ```
   openssl dgst -sha256 -verify
   cte_compatibility_matrix_public_key.pem \
   -signature cte_compatibility_matrix.sign.sha256
   ```

```
cte_compatibility_matrix.json
Verified OK
```

Installing CTE with No Key Manager Registration

The following procedure installs the CTE Agent on the host but does not register it with a key manager. You cannot protect any data on the host until the CTE Agent is registered with one of the supported key managers. For a comparison of the available key managers, see CTE Components.

1. Log on to the host where you will install the CTE Agent as `root`. You cannot install the CTE Agent without `root` access.
2. Copy or mount the installation file to the host system. If necessary, make the file executable with the `chmod` command.

3. Install the CTE Agent. A typical installation uses the following syntax:

```
./vee-fs-<release>-<build>-<system>.bin
```

For example:

```
./vee-fs-7.3.0-135-rh8-x86_64.bin
```

To install the CTE Agent in a custom directory, use the `-d <custom-dir>` option. For example:

```
./vee-fs-7.3.0-135-rh8-x86_64.bin -d /home/my-cte-dir/
```

> **Note**
>
> If possible, Thales recommends that you use the default directory `/opt/vormetric`.}

To view all installer options, use the `-h` parameter. For example:

```
./vee-fs-7.3.0-135-rh8-x86_64.bin -h
```

4. The Thales License Agreement displays. When prompted, type **Y** and press Enter to accept.
The install script installs the CTE Agent software in either `/opt/vormetric` or your custom installation directory and then prompts you about registering the CTE Agent with a key manager.

Welcome to the CipherTrust Transparent Encryption File System Agent Registration Program.

```
Agent Type: CipherTrust Transparent Encryption File System Agent
Agent Version: <Release.build-number>

In order to register with a CipherTrust Manager you need a valid
registration token from the CM.

Do you want to continue with agent registration? (Y/N) [Y]:
```

5. Type **N** and press Enter to end the installation procedure without registering the CTE Agent with either key manager.

   When you are ready to register the CTE Agent with a key manager, see Configuring CTE for Linux with CipherTrust Manager.

Configuring CTE for Linux with CipherTrust Manager

This section describes how to install and configure CTE on Linux systems that you plan to register with a CipherTrust Manager.

Installation Overview

The installation and configuration process when you are using CTE with a CipherTrust Manager consists of the following steps:

1. Installation Prerequisites

   Gather the information needed for the installation and set up your network.

2. Interactive Installation on Linux

   Install CTE interactively on a protected host and register the protected host with CipherTrust Manager.

   Silent Installation on Linux

   Install CTE silently (non-interactive) on a protected host and register the protected host with CipherTrust Manager.

3. External Certificates

   Use for communication between CTE and CM. Install the external certificate before registering CipherTrust Transparent Encryption with CipherTrust Manager.

4. Validating CipherTrust Manager and CipherTrust Transparent Encryption with a Local CA Certificate

   Ensure that registration by the CTE agent is serviced only by the expected key manager by providing a copy of the CA certificate that will be used to authenticate the TLS communications with the key manager.

5. Registration

   Register the protected host with CipherTrust Manager and make sure that CipherTrust Manager and CipherTrust Transparent Encryption can communicate

with each other. This can be done as part of the initial installation or at any point after the CTE Agent has been installed.

6. Guard Devices
   Create GuardPoint to protect sensitive data.

Interactive Installation on Linux

The Linux typical install is an interactive script that asks you a series of questions during the installation. You can also install CTE using a silent installer which pre-packages the install information. This allows you to install CTE on a large number of hosts. (For more information, see Silent Installation on Linux.

After you install CTE, you are prompted to register it immediately with a key manager. CTE must be registered with a key manager before you can protect any of the devices on the host. However, you may postpone the registration if you plan to register CTE later.

> **Note**
>
> Do not install CTE on network-mounted volumes like NFS.

Prerequisites

The following prerequisites must be met for CTE/CTE-U to install and register to CipherTrust Manager properly:

- CipherTrust Manager installed and configured. See CipherTrust Manager Documentation for more information.
- CipherTrust Manager must contain a Client Profile. See Changing the Profile for more information.
- CipherTrust Manager must contain a registration token. See Creating a Registration Token.
- Optionally, the name of the host group you want this client to be a part of.
- CipherTrust Manager must contain an LDT Communication Group if you will use CTE to guard data over CIFS/NFS shares using LDT policies. See Managing LDT Communication Groupsfor more information.

Procedure

1. Log on to the host where you will install the CTE Agent as `root`. You cannot install the CTE Agent without `root` access.
2. Copy or mount the installation file to the host system. If necessary, make the file executable with the `chmod` command.
3. Install the CTE Agent. A typical installation uses the following syntax:

   ```
   ./vee-fs-<release>-<build>-<system>.bin
   ```

   For example:

   ```
   ./vee-fs-7.3.0-135-rh8-x86_64.bin
   ```

   To install the CTE Agent in a custom directory, use the `-d <custom-dir>` option. For example:

   ```
   ./vee-fs-7.3.0-135-rh8-x86_64.bin -d /home/my-cte-dir/
   ```

   > **Note**
   >
   > If possible, Thales recommends that you use the default directory `/opt/vormetric`.

   To view all installer options, use the `-h` parameter. For example:

   ```
   ./vee-fs-7.3.0-135-rh8-x86_64.bin -h
   ```

4. The Thales License Agreement displays. When prompted, type **Y** and press Enter to accept.
   The install script installs the CTE Agent software in either `/opt/vormetric` or your custom installation directory and then prompts you about registering the CTE Agent with a key manager.

   Welcome to the CipherTrust Transparent Encryption File System Agent Registration Program.

```
Agent Type: CipherTrust Transparent Encryption File System Agent
Agent Version: <Release.build-number>

In order to register with a CipherTrust Manager you need a valid
registration token from the CM.

Do you want to continue with agent registration? (Y/N) [Y]:
```

5. Type **N** and press Enter to end the installation procedure without registering the CTE Agent with either key manager.
6. Enter **Y** to continue with the registration process. The install script prompts you to enter the host name or IP address of the CipherTrust Manager with which you want to register CTE. For example:

   Do you want to continue with agent registration? (Y/N) [Y]: Y

```
Please enter the primary key manager host name: 10.3.200.141
```

> **Note**
>
> The default communication port is 443. If you want to specify a different communication port, enter it with the primary key manager host name in the format: *<hostName>:<port#>*

   You entered the host name 10.3.200.141
   Is this host name correct? (Y/N) [Y]: Y

7. Enter the client host name when prompted.

```
Please enter the host name of this machine, or select from the fo
llowing list.

[1] sys31186.qa.com
[2] 10.3.31.186

Enter a number, or type a different host name or IP address in ma
nually:
```

```
What is the name of this machine? [1]: 2
You selected "10.3.31.186".
```

8. Enter the CipherTrust Manager registration token, profile name, host group and host description. If you omit the profile name, CipherTrust Manager associates the default client profile with this client.

```
Please enter the registration token: 12345
Please enter the profile name for this host: My-Profile
Please enter the host group name for this host, if any:
Please enter a description for this host: RHEL7 system West Coast
Datacenter

Token            : 12345
Profile name     : My-Profile
Host Group       : (none)
Host description : RHEL7 system West Coast Datacenter
Are the above values correct?  (Y/N) [Y]: Y
```

9. At the hardware association prompt, select whether you want to enable the hardware association feature to prevent cloning. The default is Y (enabled):

   It is possible to associate this installation with the hardware of this machine. If selected, the agent will not contact the key manager or use any cryptographic keys if any of this machine's hardware is changed. This can be rectified by running this registration program again. Do you want to enable this functionality? (Y/N) [Y]: Y

10. At the LDT prompt, specify that you want this client to use CTE-LDT by typing **Y** and pressing Enter:

```
Do you want this host to have LDT support enabled on the server?
(Y/N) [N]: Y
```

11. If you are planning to create GuardPoints on NFS shares, enter the name of the LDT Communication Group that this node will join.

```
Enter the LDT Communication Group name: LCG1
```

Warning

The registration token, profile name, client group name and LDT Communication Group name are case-sensitive. If any of these are entered incorrectly, the client registration will not succeed. If the registration fails, click Back in the installer and verify that the case is correct for all entries on this page.

12. At the Cloud Object Storage (COS) prompt, specify whether you want this client to use CTE COS.

```
Do you want to configure this host for Cloud Object Storage? (Y/N)
[N]:
```

13. CTE finishes the installation and registration process.

```
Generating key pair for the kernel component...done.
Extracting SECFS key
Generating EC certificate signing request for the vmd...done.
Signing certificate...done.
Enrolling agent with service on 10.3.200.141...done.
Successfully registered the CipherTrust Transparent Encryption CTE
Agent with the CipherTrust Manager on 10.3.200.141.

Installation success.
```

14. In CipherTrust Manager, change the client password using the manual password creation method. This password allows users to access encrypted data if the client is ever disconnected from the CipherTrust Manager. For details on changing the password, see the CipherTrust Manager documentation.

Silent Installation for CTE or CTE-U on Linux

This section describes how to perform a silent (unattended) installation on CTE or CTE-U on a single host. The silent installation automates the installation process by storing the answers to installation and registration questions in a separate file that you create. It installs CTE/CTE-U on the host, and registers the host with the CipherTrust Manager you specify in the silent installation file. You can also use the silent installation to install CTE/CTE-U, on multiple hosts simultaneously.

Prerequisites

The following prerequisites must be met for CTE/CTE-U to install and register to CipherTrust Manager properly:

- CipherTrust Manager installed and configured. See CipherTrust Manager Documentation for more information.
- CipherTrust Manager must contain a Client Profile. See Changing the Profile for more information.
- CipherTrust Manager must contain a registration token. See Creating a Registration Token.
- Optionally, the name of the host group you want this client to be a part of.

Procedure

1. Log on as an administrator to the host where you will install CTE/CTE-U.
2. Create a parameter file and store it on your system, or copy an existing file from another location. The file can contain any of the following parameters:

   SERVER_HOSTNAME

   Required if you want to register CTE with a CipherTrust Manager.

   SERVER_IP

   Alternative for hostname when registering.

   REG_TOKEN

   The registration token for the CipherTrust Manager with which you plan to register this client. Required for registration.

   HOST_PROFILE

Specifies the client profile in the CipherTrust Manager that will be associated with this client. If this value is omitted, the CipherTrust Manager uses the default client profile.

TMPDIR

Specifies a custom temporary directory that the installer can use during the installation process. If this value is omitted, the installer uses the default temporary directory.

AGENT_HOST_NAME

FQDN of the host on which the CTE Agent is being installed. If this value is not specified, the installer uses the host's IP address.

AGENT_USEIP

Use the IP address of the protected host instead of host name. Used when hostname is not supplied.

AGENT_HOST_PORT

Specifies the port number for this CTE Agent to use.

HOST_GROUP

Specifies the optional host/client group with which this host/client will be associated.

HOST_DESC

Specifies a description for the host. This description is displayed in the CipherTrust Manager. If an entry for this host already exists, and the host already has a description, CipherTrust Manager **does not** overwrite the existing description, even if this option is specified.

USEHWSIG

Set this value to 1 when you want to associate this installation with the machine hardware for cloning prevention.

CTE

CA_CERT

Set to provide CA certificate data to CipherTrust Manager.

CA_FILE

Set to provide a CA certificate file to CipherTrust Manager.

ENABLE_CLOUD

Set to enable cloud in the Key Manager.

ENABLE_LDT

Set this value to **1** to automatically enable and register CTE-LDT (Live Data Transformation) for this host on your key manager during the silent install.

LDTGROUP_NAME

Set the LDT Communication Group for LDT over NFS/CIFS.

CERT_FIELD_PARAM

Example 1: Registering with CipherTrust Manager

The following example contains just the required information for registration with CipherTrust Manager. In this case, the client will be registered with the CipherTrust Manager using its IP address instead of its host name:

```
SERVER_HOSTNAME=Key-Mgmt-Server.example.com
REG_TOKEN=12345
AGENT_HOST_NAME=10.192.80.86
```

Example 2: Registering with CipherTrust Manager

The following example specifies the required registration information, adds a host name and description, enables hardware association, and CTE-LDT. In this case, the client will be registered with the CipherTrust Manager using its host name instead of the IP address:

```
SERVER_HOSTNAME=Key-Mgmt-Server.example.com
REG_TOKEN=12345
AGENT_HOST_NAME=myagent.example.com
HOST_DESC="West Coast Server 12"
USEHWSIG=1
CERT_FIELD_PARAM="/C=US/ST=California/L=San Jose/O=Thales
eSecurity/OU=Vormetrics/CN=localhost/
emailAddress=admin@thalegroup.com"
SUBJECT_ALT_NAME_PARAM="DNS:www.thalesgroup.com,email:admin@thale
sgroup.com"
```

CTE-U

CERT_FIELD_PARAM

Example: Registering with CipherTrust Manager

The following example specifies the required registration information, adds a host name and description, enables hardware association, CTE-LDT. In this case, the client will be registered with the CipherTrust Manager using its host name instead of the IP address:

```
    SERVER_HOSTNAME=Key-Mgmt-Server.example.com
    REG_TOKEN=12345
    AGENT_HOST_NAME=myagent.example.com
    HOST_DESC="West Coast Server 12"
    USEHWSIG=1
    CERT_FIELD_PARAM="/C=US/ST=California/L=San Jose/O=Thales
eSecurity/OU=Vormetrics/CN=localhost/
emailAddress=admin@thalegroup.com"
    SUBJECT_ALT_NAME_PARAM="DNS:www.thalesgroup.com,email:admin@t
halesgroup.com"
```

3. Copy or mount the installation file to the host system. The installation file is in the format:

CTE

```
vee-fs-<release>-<build>-<system>.bin
```

CTE-U

```
cte-fuse_<version>.<build>_<processor>.rpm
```

4. Run the installer using the following syntax:

CTE

```
./vee-fs-<release>-<build>-<system>.bin [-d <custom-dir>] -s
<install-file>
```

where:

- `-d <custom-dir>` is an optional parameter that specifies the installation directory for CTE. If you omit this parameter, CTE is installed in: `/opt/vormetric/DataSecurityExpert/agent/`
- `-s <install-file>` indicates that you want to install CTE silently using the installation options file `<install-file>`

For example, if the installation options file is called `/tmp/unattended.txt`, you would enter:

```
./vee-fs-7.3.0-135-rh8-x86_64.bin -s /tmp/unattended.txt
```

CTE-U

```
rpm -i cte-fuse_<version>.<build>_<processor>.rpm
```

- Invoke Registration for CTE-U:

```
/opt/vormetric/DataSecurityExpert/agent/vmd/bin/
register_host silent <silent_reg_file>
```

Note

You can also invoke registration by replacing 'register_host' with the following two options:

- **`REG_HOST_CLEAN`: Register and remove certificates used for communication**
- **`REG_HOST_SILENT_<fileName>`: Use silent/non-interactive mode; get name=value input from**

5. Verify the installation by checking the CTE/CTE-U processes on the host:
    - Run `vmd -v` to check that the version of CTE/CTE-U matches that just installed.
    - Run `vmsec status` to display CTE kernel status. (CTE only)
    - Look at the log files in `/var/log/vormetric`, especially `install.fs.log.<date>` and `vorvmd_root.log`.
6. In CipherTrust Manager, change the client password using the manual password creation method. This password allows users to access encrypted data if the client is ever disconnected from the CipherTrust Manager. For details on changing the password, see the CipherTrust Manager documentation.

Using external certificates for communication between CTE Agent and CipherTrust Manager

Overview

CipherTrust Transparent Encryption can now use an external certificate, available at a user-defined path, to communicate with CipherTrust Manager.

Prerequisites

The external certificate must be:

- On the file system
- In PEM format

A key pair must already exist for the client:

- Must have Encryption type of either:
    - sha256WithRSAEncryption
    - ecdsa-with-SHA384
- Must be Encrypted with a pass phrase

Initial setup

1. Obtain your external CA certificate.
2. Create a certificate using the external CA certificate and key.

CipherTrust Manager Setup

To setup CipherTrust Manager to communicate through an external certificate:

1. Import the CA certificate into the CipherTrust Manager, click **CA** > **External** > **Add External CA**.

   Note

   In the Add External CA dialog, copy and paste the `<ca_certificate_name>`.pem file content from the UI page and provide a user-friendly name.

   For more information, see Using an Externally Generated Server Certificate for an Interface
2. Add the CA certificate to the list of trusted sources for the web interface, click **Admin Settings** > **Interfaces** > **web** > **Edit** > **External Trusted CAs**.
3. Restart the web server, click **Admin Settings** > **Services** > **web** > **Restart**.
4. Create a Registration Token for the CTE agent.

CTE Agent setup

1. Create a directory on the system to hold the required files, for example:
   - `/root/cert_files` (**Linux/AIX**)
   - `c:\temp\cert_files` (**Windows**)
2. Copy or create the following files in this directory:
   - **client_cert.pem**
   - **client_key.pem**
   - **passphrase** - this is currently expected as plain text
3. For **Linux/AIX** systems, to add the directory path to the environment, type:

```
$ export EXTERNAL_CERT_DIR=/root/cert_files
```

4. For **Windows** system, invoke registerhost.exe from the command line and add this argument:

```
c:\> register_host.exe -extcertdir=c:\temp\cert_files
```

5. Register the CTE client with the CM server as normal. If this is being done as part of an installation, then the above steps should be done before the installation, or, on windows, added to the registration parameters passed to the installer.

Post Registration

During registration, the certificate file is uploaded to the CipherTrust Manager, and the certificate and key files are imported into the CTE pem store. The key is decoded using the provided passphrase, then re-encoded using a random key using the normal CTE key security mechanisms for TLS keys. There is no need to keep the input files after registration is successful, so for security reasons they should be removed / shredded.

Certificate Renewal

The location of the external certificate files (i.e. the `EXTERNAL_CERT_DIR` or `-extcertdir` parameters) will be recorded in the CTE agent configuration file, `agent.conf`. When the current certificate is approaching expiration date (i.e. approx. 60 days prior to expiration) the CTE agent will look in this directory for an updated set of files.

If a new certificate file is present, then the file will be read and pushed to the CM, and if accepted, then the certificate and key will be imported into the CTE pem store, and the VMD process restarted to use the new certificate.

If no new certificate is present, a WARNING level message will be written to the logs and/or uploaded to the CM as per the logging settings, and the CTE agent will check again after 24 hours.

If the user wishes to change the directory path to store the new certificates, then the entry in the `agent.conf` file should be updated and the vmd service restarted. Alternatively, the user can update the external certificate set using the following command (this will not update the saved path):

```
# vmutil -a vmd -d <ext_cert_Dir> updatecerts
```

If the user fails to update the certificate set prior to expiration then communication with the CM may be blocked, and re-registration will be required.

Note

Any renewed certificates must have exactly the same common name field as the original certificate, or the CipherTrust Manager will reject the update.

Validating CM and CTE with a Local CA Certificate

To ensure that registration by the CTE agent is serviced only by the expected key manager, you can provide a copy of the root CA certificate that will be used to authenticate the TLS communications with the key manager, during the registration process.

Note

You can only download the CA certificate when you are a root user in the root domain. You cannot download the certificate from a subdomain. It will not work.

Prerequisite

Make sure that you have previously created the client in CipherTrust Manager.

Using a Local CA Certificate

1. Extract the root CA certificate from the CipherTrust Manager.
    a. Log on to CipherTrust Manager as an administrator.
    b. In the left navigation pane, click **CA > Local**. The list of available CAs displays.
    c. Click the ellipsis icon corresponding to the CA.
    d. Click **Download** to download the CA.

e. Copy the certificate to a directory on the agent system.

2. Present the root certificate data to CTE in one of two ways:

   a. Use a file:

   When written to a file, it must be in PEM file format, starting and ending with:

   ```
   -----BEGIN CERTIFICATE-----
   -----END CERTIFICATE-------
   ```

   b. Use a a string parameter:

   If you are providing the information in a single string, it must contain the same data as in the preceding case, except that all new lines are replaced by `\n` escape sequences. For example:

   CA_CERT=-----BEGIN CERTIFICATE-----\n -----END CERTIFICATE-----\n

3. To install the root certificate into the CTE client:

```
    /opt/vormetric/DataSecurityExpert/agent/vmd/bin/
register_host.exe -s <Path-to-install-file> AGENT_HOST_NAME=<Host
name-or-IP-of-agent> REG_TOKEN=<CM registration token> CA_FILE=<P
ath-to-root-ca-cert>
```

Example

```
    /opt/vormetric/DataSecurityExpert/agent/vmd/bin/
register_host.exe -s /opt/silent/vte_reg_log.txt AGENT_HOST_NAME=a
ni-vm-217-35190.sjcicd.com REG_TOKEN=mMEz3Y6Ob9D4L7QuvK5SOmhulRm8
DYI8odV5j3OdvuHqk6LhZqE0FeIZHILYTmDiE9 CA_FILE=/cert_files/Austin1
75.pem
```

4. Confirm in CipherTrust Manager that the client is registered and healthy.

Installation and Registration Options

CTE provides the following installation and registration options. The options you choose determine the information you need to supply during the actual install procedure.

Installation Method Options

There are two methods for installing CTE on Linux platforms:

- **Typical**: Most common and recommended type of installation. Use this method for installing the CTE Agent on one host at a time. See Interactive Installation on Linux.
- **Silent**: Create pre-packaged installations by providing information and answers to a set of installation questions. Use silent installations when installing on a large number of hosts. See Silent Installation on Linux.

Hardware Association (Cloning Prevention) Option

CTE's hardware association feature associates the installation of CTE with the machine's hardware. When enabled, hardware association prohibits cloned or copied versions of CTE from contacting the key manager and acquiring cryptographic keys. Hardware association works on both virtual machines and hardware clients.

You can enable hardware association during CTE registration process. You can disable hardware association by re-running the registration program.

To verify if hardware association (cloning prevention) is enabled on a Linux host, on the command line enter the following:

```
cat /opt/vormetric/DataSecurityExpert/agent/vmd/etc/access
```

If you see `usehw:true`, then hardware association is enabled. If you see `usehw:false`, it's disabled.

Guarding a Device with CipherTrust Manager

After you register a client with a CipherTrust Manager, you can create as many standard GuardPoints on the client as you need. These GuardPoints can protect an entire device or individual directories.

> **Note**
>
> For guarding using LDT on a local drive, or on a CIFS/Share drive, refer to CTE-Live Data Transformation with CipherTrust Manager guide.

In order to guard a device or directory, you need to use the CipherTrust Manager Console to:

1. Access the CipherTrust Manager domain in which the client is registered.
2. Identify or create an encryption key that CTE will use to encrypt the data on the device or directory.
3. Identify or create a policy for the device or directory that specifies the access controls and the encryption keys to use for the device or directory.
4. Assign a GuardPoint to the device or directory.

The following example creates a simple policy and uses it to guard a directory on a registered client. For all of the following procedures, you must be logged into the CipherTrust Manager Console as a CipherTrust Manager Administrator, and you must be in the domain with which the client is registered.

For details about any of these procedures or the options for domains, encryption keys, policies, and GuardPoints, see the CipherTrust Manager documentation.

Access the CipherTrust Manager Domain

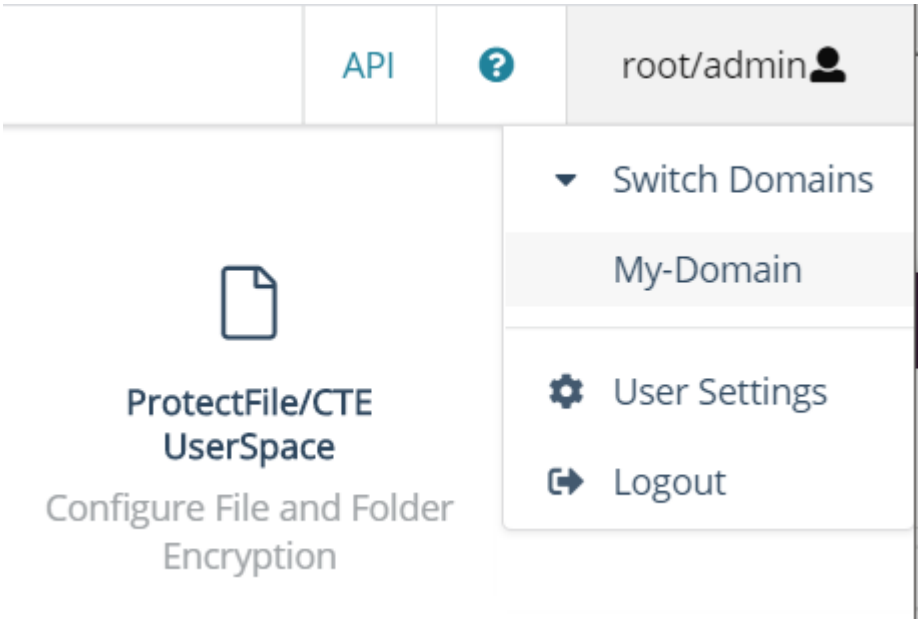
1. In a web browser, navigate to the URL of the CipherTrust Manager Console you want to use and log in with CipherTrust Manager Administrator credentials.

2. If the client you want to protect is registered to the default domain (root), proceed to Create an Encryption Key. If you need to change to a different domain, do the following:

   a. In the top menu bar, click the user name **root/admin** on the right-hand side.

   b. Select **Switch Domains**, then select the domain in which the client is registered.

   c. The logged in user now shows the new domain name/user name.



Create an Encryption Key

Note

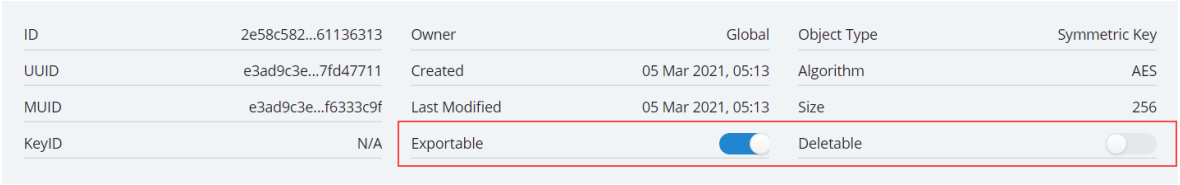
The following procedure is based on CipherTrust Manager version 2.2. If you are using a different version, see the CipherTrust Manager documentation for the version that you are using.

1. From the Products page in the CipherTrust Manager Console, click **Keys** in the left hand pane.

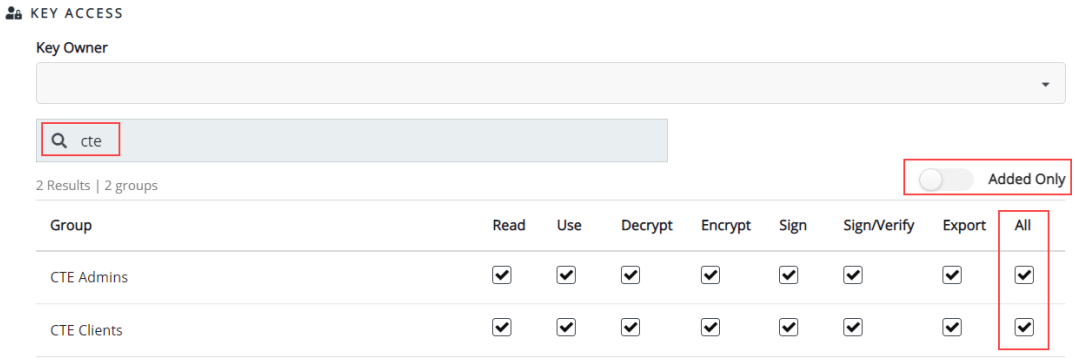
Tip

To navigate to the Products page from anywhere in the CipherTrust Manager Console, click the App Switcher icon in the top left corner.

2. Above the Key table, click **Create a New Key**.
3. In the **Key Name** field, add a name for the key. This name must be unique. For example, Simple-Key.
4. In the **Key Usage** section, make sure **Encrypt** and **Decrypt** are selected.
5. Click **Create**. CipherTrust Manager displays the properties for the new key.
6. In the general options area, enable the **Exportable** option.
   You can also enable the **Deletable** option in this section if you want a CipherTrust Manager Administrator to be able to delete the key.



7. In the **Key Access** section, do the following:
   a. In the Search Groups box, type "cte".
      If no groups are displayed, make sure the **Added Only** option is *disabled*.
   b. Click the **All** check box for both the CTE Admins and CTE Clients groups.



   c. When you are done, click **Update**.
8. Click the **CTE** tab and set the following properties:
   - **CTE Versioned**: Specify whether the key is versioned. By default, the key is set as versioned.
     For a standard policy, you should clear this check box. If you do not, the key will *not* appear in the keys list when you add the key rule to the standard policy.
   - **Persistent on Client**: Specify whether the key is stored in persistent memory on the client.

When the check box is selected, the key is downloaded and stored (in an encrypted form) in persistent memory on the client.

When the check box is left clear, the key is downloaded to non-persistent memory on the client. Every time the key is needed, the client retrieves it from the CipherTrust Manager. This is the default setting.

- **Encryption Mode**: Encryption mode of the key. The options are:
  - CBC
  - CBC-CS1
  - XTS

Encryption using the XTS and CBC-CS1 keys is known as enhanced encryption.

When you are done, click **Update**.

Create a Standard Policy

1. In the Applications page of the CipherTrust Manager Console, select the **Transparent Encryption** application.
2. In the sidebar on the Clients page, click **Policies**.
3. Click **Create Policy**. CipherTrust Manager displays the Create Policy Wizard.
4. On the General Info page, set the following options:

Field	Description
Name	A unique name for the policy. Make sure you use a name that is descriptive and easy to remember so that you can find it quickly when you want to associate it with a GuardPoint. This example uses "Simple-Policy".
Policy Type	The type of policy you want to create. In this example, we will create a **Standard** policy.
Description	A user-defined description to help you identify the policy later. For example: Standard policy for new GuardPoints.
Learn Mode	Learn Mode provides a temporary method for disabling the blocking behavior of CTE/CTE-LDT policies. While useful for quality assurance, troubleshooting, and mitigating deployment risk, Learn Mode is not intended to be enabled permanently for a policy in production. This prevents the

Field	Description
	policy Deny rules from functioning as designed in the policy rule set. Ensure that the policy is properly configured for use in Learn Mode. Any Security Rule that contains a Deny effect must have Apply Key applied as well. This is to prevent data from being written in mixed states, resulting in the loss of access or data corruption. Apply Key will have no effect when combined with a Deny rule unless the policy is in Learn Mode.
Data Transformation	If you select **Standard** as the policy type, also select the the **Data Transformation** option to tell CTE that you want to change the current encryption key used on the data in the GuardPoint, or that you want to encrypt clear-text data for the first time. This option is only displayed for Standard policies.

When you are done, click **Next**.

5. On the Security Rules page, define the security rules that you want to use.
   CipherTrust Manager automatically adds a default security access rule with an action of `key_op` and the effects `Permit` and `Apply Key`. This rule permits key operations on all resources, without denying user or application access to resources. This allows it to perform a rekey operation whenever the encryption key rotates to a new version.
   To add additional security rules, click **Create Security Rule** and enter the requested information. For details about adding security rules, see the CipherTrust Manager documentation.
   When you are done, click **Next**.

6. On the Create Key Rule page, click **Create Key Rule** and enter the following information:

Field	Description
Resource Set	If you want to select a resource set for this key rule, click Select and either choose an existing resource set or create a new one. Resource sets let you specify which directories or files will either be encrypted with the key or will be excluded from encryption with this key.

Field	Description
Current Key Name	Click **Select** to choose an existing key or create a new one. If the data has not yet been encrypted, select **clear_key**. Otherwise select the name of the non-versioned key that is currently being used to encrypt the data. In this example, select **clear_key**.
Transformation Key Name	Click **Select** to choose an existing versioned key or to create a new one. CTE uses the versioned key specified in this field to encrypt the data in the GuardPoint. If the data is currently encrypted, CTE decrypts it using the key specified in the **Current Key Name** field and re-encrypts it using the key specified in this field.

When you are done, click **Next**.

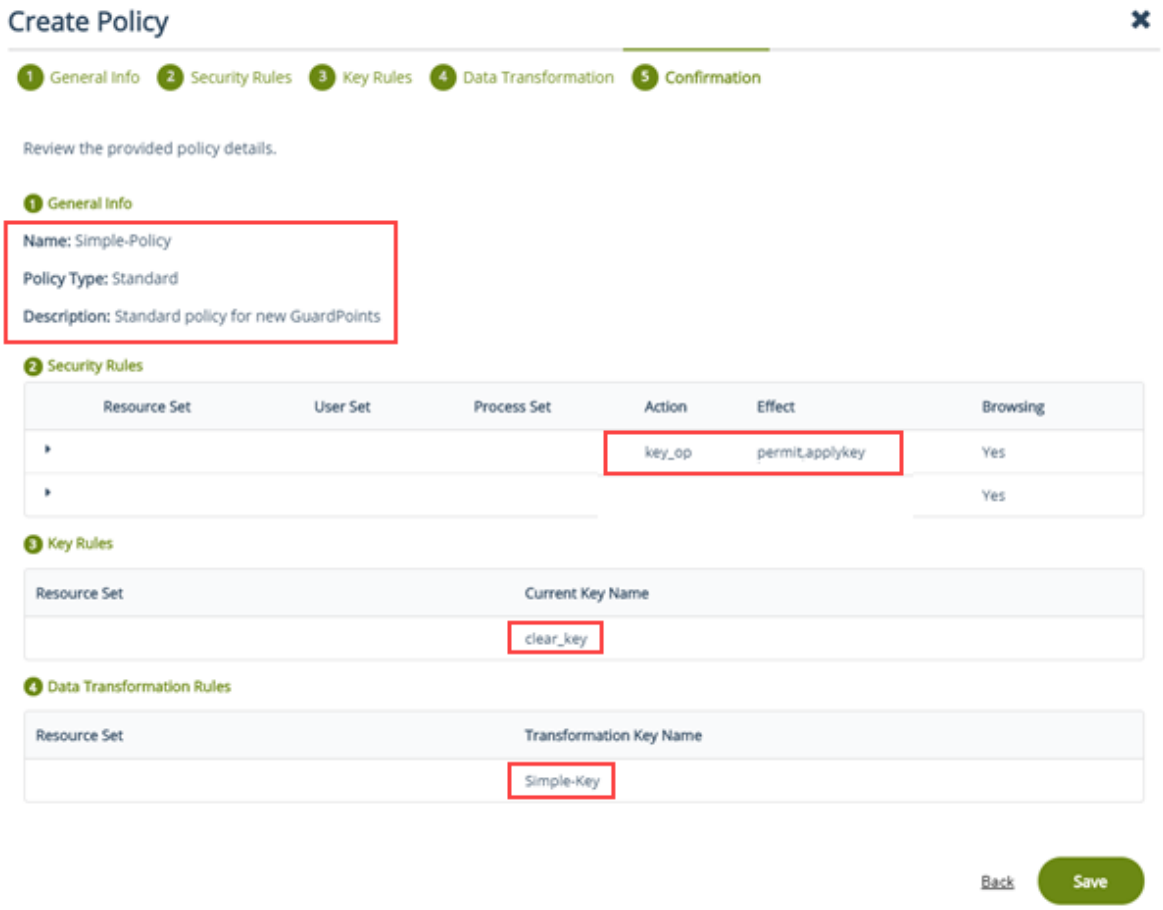
7. On the Data Transformation page, click **Create Data Transformation Rule** and enter the following information:

Field	Description
Resource Set	If you want to select a resource set for this key rule, click Select and either choose an existing resource set or create a new one. Resource sets let you specify which directories or files will either be encrypted with the key or will be excluded from encryption with this key.
Transformation Key Name	Click **Select** to choose an existing key or to create a new one. CTE uses the key specified in this field to encrypt the data in the GuardPoint. If the data is currently encrypted, CTE decrypts it using the key specified in the **Current Key Name** field and re-encrypts it using the key specified in this field. For this example, select the key Simple-Key you created in Create an Encryption Key.

When you are done, click **Next**.

8. Click **Next**.

9. On the confirmation page, review the information for the policy and click **Save**.

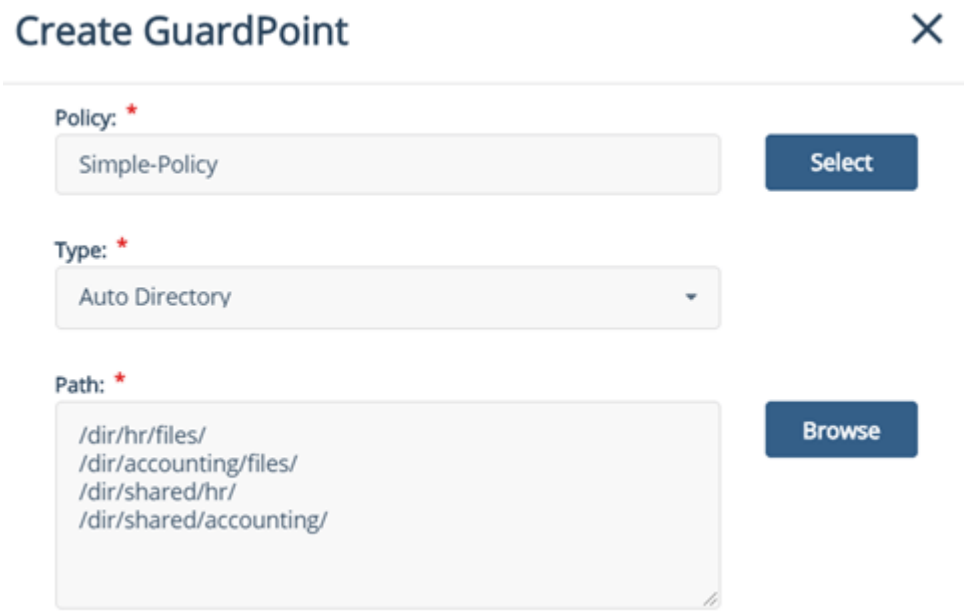


Create a GuardPoint

1. Stop all applications that are accessing the device you want to protect. In this example, we are going to protect the following directories with the same policy and encryption key.
    - /dir/hr/files
    - /dir/accounting/files
    - /dir/shared/hr
    - /dir/shared/accounting

> **Tip**
>
> If you want to encrypt data without taking the device offline, you must use CipherTrust Transparent Encryption - Live Data Transformation.

2. In the Applications page of the CipherTrust Manager Console, select the **CTE** application.
3. In the Clients table, click on the name of the client you want to protect.
4. Above the GuardPoints table, click **Create GuardPoint**.
5. In the Create GuardPoint page:
    a. In the **Policy** field, select the policy you created earlier.
    b. In the Type field, select the type of device. You can guard a directory or a raw/block device. For this example, select **Auto Directory**.
    c. In the **Path** field, enter the directories you want to protect with this policy or click **Browse** to select them from a explorer window.
    If you want to enter multiple paths, put each path on its own line. For example:



    d. Click **Create**.
    e. If you want to use the same policy and GuardPoint type on another path, click **Yes** when prompted. Otherwise, click **No**. For this example, click No.

    The CTE clients pull the GuardPoint configuration information from the CipherTrust Manager.
6. Type the following to transform the data:

```
dataxform --rekey --print_stat --preserve_modified_time --gp
<pathToGP>
```

When the data transformation has finished, applications can resume accessing the now-protected data. (See the CTE Data Transformation Guide for more information.)

Multifactor Authentication for CTE GuardPoints

CipherTrust Transparent Encryption is supporting Multifactor Authentication through integration with an MFA provider. CipherTrust Transparent Encryption will continue to integrate with additional providers and release that information in the future.

Note

Multifactor Authentication for CipherTrust Transparent Encryption Linux is only supported with CipherTrust Manager v2.14 and subsequent versions.

- Introduction to Multifactor Authentication
- Set up your account with KeyCloak
- Use Cases for Multifactor Authentication
- Exempting some users from authentication with a Whitelist
- Setting up Multifactor Authentication with a One-Time-Password
- Setting up Multifactor Authentication with a Time Out
- Administration for Multifactor Authentication
- Troubleshooting Multifactor Authentication

Introduction to Multifactor Authentication

Why do companies need Multifactor Authentication

Every day, the threat of ransomware attacks increase in frequency, sophistication, and effectiveness. Victims of ransomware attacks can be blocked from data, applications, and systems – making an organization unable to function.

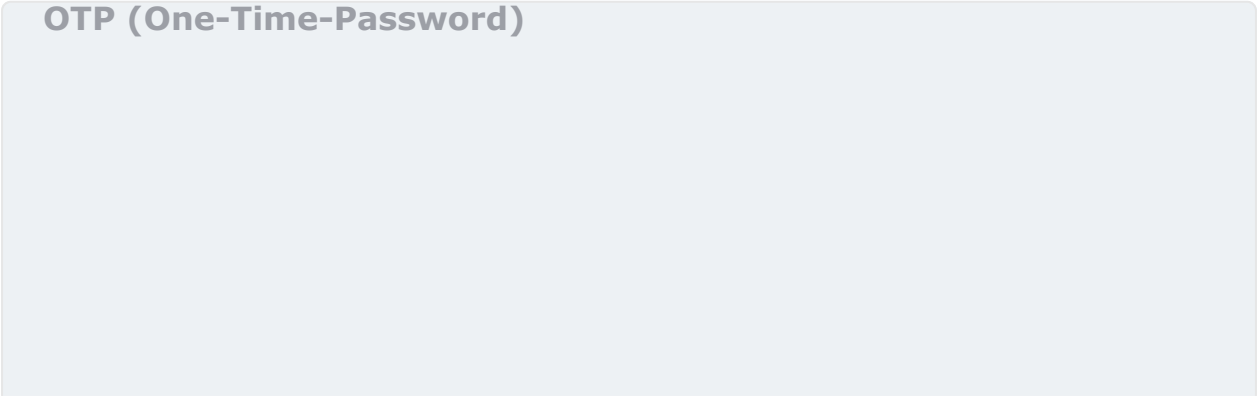
Credential compromise is the leading cause of ransomware attacks, because credentials give hackers the access they need to hold your systems hostage. Unfortunately, credentials can be stolen, shared, bought or hacked. Once the hackers gain entry, the threat actors will often look to compromise privileged access credentials to further infiltrate your network and steal sensitive data.

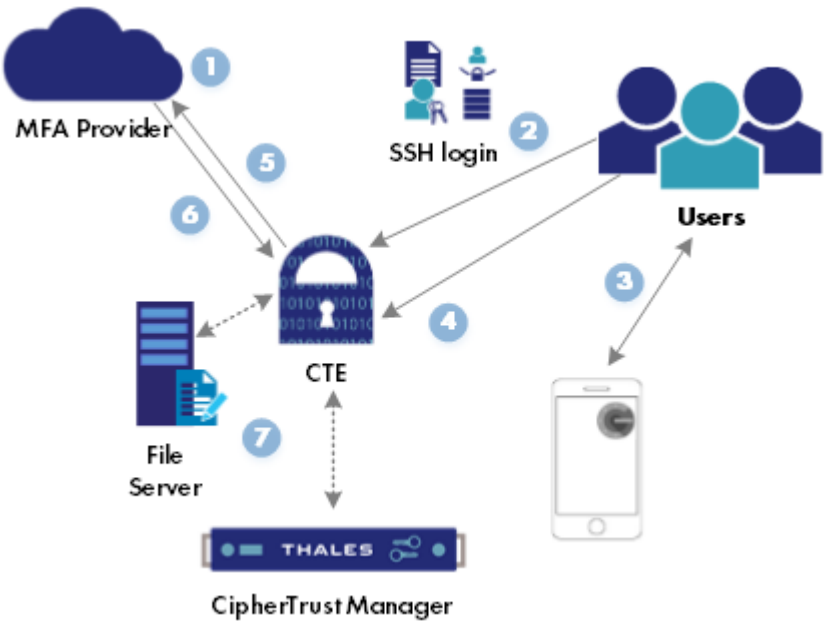
What is Multifactor Authentication

Multifactor authorization ensures that the access credentials presented belong to the actual person. After logging in to the system, when a user tries to access a CipherTrust Transparent Encryption GuardPoint, it triggers a second factor authorization to verify the user with a second form of authentication, like sending a passcode to the users's registered cell phone, that they then have to input into the application.

How does Multifactor Authentication work

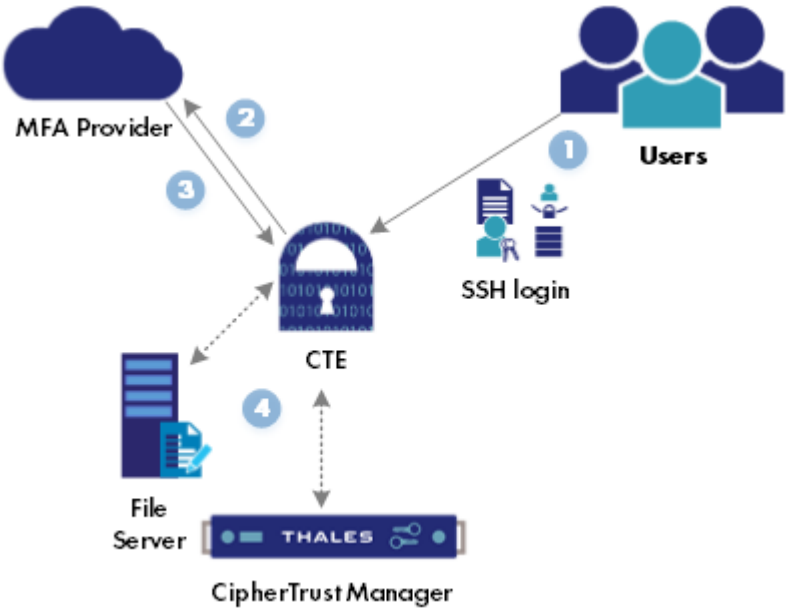
The following diagram explains how Multifactor Authentication operates in a CipherTrust Transparent Encryption environment.





Step	Description
1	A Multifactor Authentication administrator configures the MFA provider to use OTP (one-time password) for CTE multifactor authentication.
2	Multifactor Authentication is performed when either an SSH connection with MFA has already been enabled by a root user using the command `voradmin mfa ssh_enable`, and a user starts an SSH connection; or after an SSH connection is already established and a user uses the command: `voradmin mfa login`.
3	A CTE user who has previously registered the CTE site URL with an authenticator application (such as Google Authenticator), running on his/her registered mobile device, uses the authenticator to generate a one time password.
4	CTE user enters the OTP from their mobile phone into CTE.
5	CipherTrust Transparent Encryption sends a message to the Multifactor Authentication provider to verify the user.
6	Multifactor Authentication provider confirms/denies user access.
7	If authenticated using voradmin `voradmin mfa login`, the user's current shell, and all of the programs running inside that same shell, are authenticated. If authenticated during an ssh login, the entire ssh session is authenticated for the user.

Password



Step	Description
1	Multifactor Authentication is performed when either an SSH connection with MFA has already been enabled by a root user using the command `voradmin mfa ssh_enable` and a user starts an SSH connection, or after an SSH connection is already established and a user uses the command: `voradmin mfa login` and enters their Multifactor Authentication password.
2	CTE sends a verification request to the Multifactor Authentication.
3	Multifactor Authentication provider confirms/denies user access.
4	If authenticated using voradmin `voradmin mfa login`, the user's current shell, and all of the programs running inside that same shell, are authenticated. If authenticated during an ssh login, the entire ssh session is authenticated for the user.

Note

Multifactor Authentication is **NOT** supported on RHEL 7 or SLES 12.

For more information on Multifactor Authentication, see Multifactor Authentication.

Using Keycloak for Multifactor Authentication for CTE GuardPoints

Integration with Keycloak requires creating an OIDC connection in CipherTrust Manager, after you create an OIDC template in Keycloak.

Prerequisites

- Have a CipherTrust Manager set up with:
  - Client Profile enabled for Multifactor Authentication
  - Registration token

On the Keycloak platform

The following is the minimum setup for a Keycloak platform. Refer to Keycloak documentation for more information.

1. Create an admin user.
2. Login to the realm and create one or more users.
3. Create a password for the user.
4. Create an OIDC client in realm with the following settings enabled:
   - **General Settings**:
     - **Client Type**: OpenID Connect
     - **Client ID**: Client name
   - **Capability Config**:
     - **Client Authentication: On**
   - **Authentication flow**:
     - **Select: Standard flow and Direct access grants**
5. Note three OIDC parameters:
   - **OIDC** Provider: `https://<keycloak-name>:<keycloak-port>/realms/<realm-name>/.well-known/openid-configuration`

- Client-ID as configured for the OIDC client
- Client-Secret as shown for the OIDC client

Create an OIDC connection on CipherTrust Manager

1. Log on to the CipherTrust Manager GUI as an administrator.
2. In the left pane, click **Access Management** > **Connections**.
3. In the Connections, click **Add Connection**.
4. Click **OIDC** and then click **Next**.
5. Provide a name for the connection and click **Next**.
6. Enter values for the configuration information.

> **Note**
>
> Refer to your Multifactor Authentication provider profile for the values:

- **URL of OIDC provider:**

Linux

- **For KeyCloak, select the URL of the OIDC provider**

Windows

- For Thales Safenet Trusted Access, select **Well Known Configuration URL**
- For all other providers, select the URL of the OIDC provider

- Client-ID as configured for the OIDC client
- Client-Secret as shown for the OIDC client

7. Click **Next** and in the Add Products window, select **CTE** for product.
8. Click **Add Connection**.

Use Cases for MFA on CTE

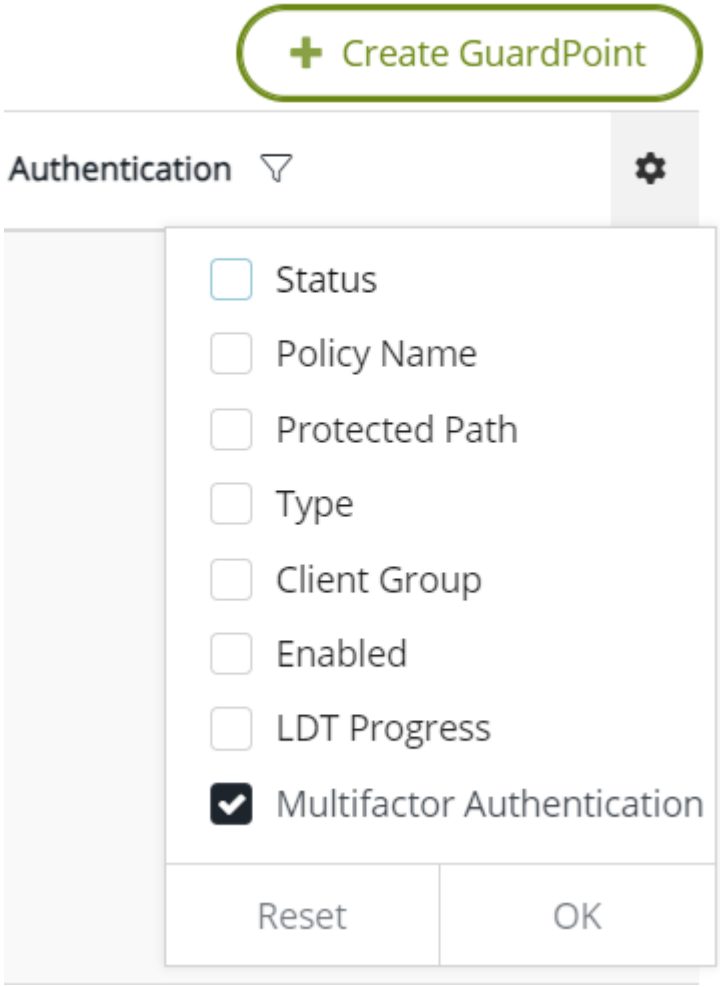
When using Multifactor Authentication with CipherTrust Transparent Encryption, you can:

- Enable Multifactor Authentication on a single GuardPoint
- Enable Multifactor Authentication on the entire Client
- Perform Multifactor Authentication
- Enable Multifactor Authentication for Client Groups

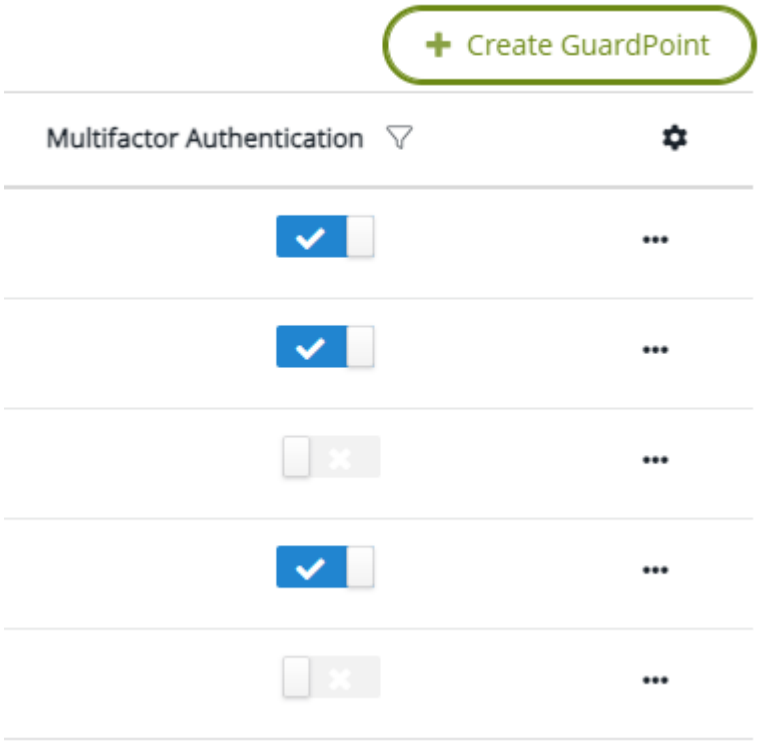
Enable Multifactor Authentication on a Single GuardPoint

You can enable Multifactor Authentication for individual GuardPoints on clients.

1. Open **CipherTrust Manager > Transparent Encryption** application.
2. Select the relevant client.
3. Select the **GuardPoints** tab.
4. Click the settings icon.
5. Select **Multifactor Authentication** to enable Multifactor Authentication for the GuardPoints.



6. Click **OK**. The Multifactor Authentication column displays.



7. Toggle the Multifactor Authentication switch to enable Multifactor Authentication for the selected GuardPoints.

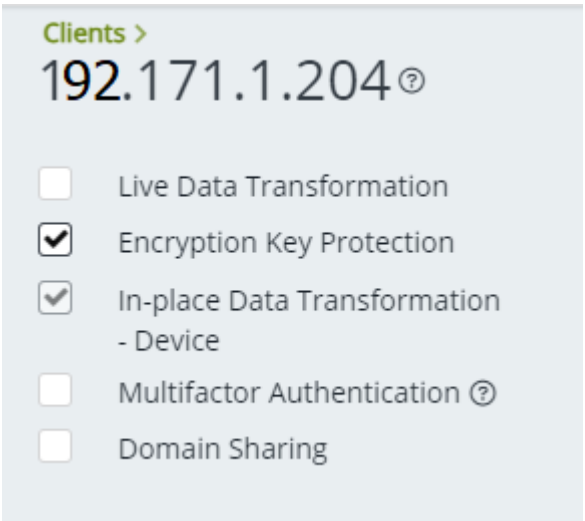
Note

To disable Multifactor Authentication on a GuardPoint, deselect the Multifactor Authentication toggle switches.

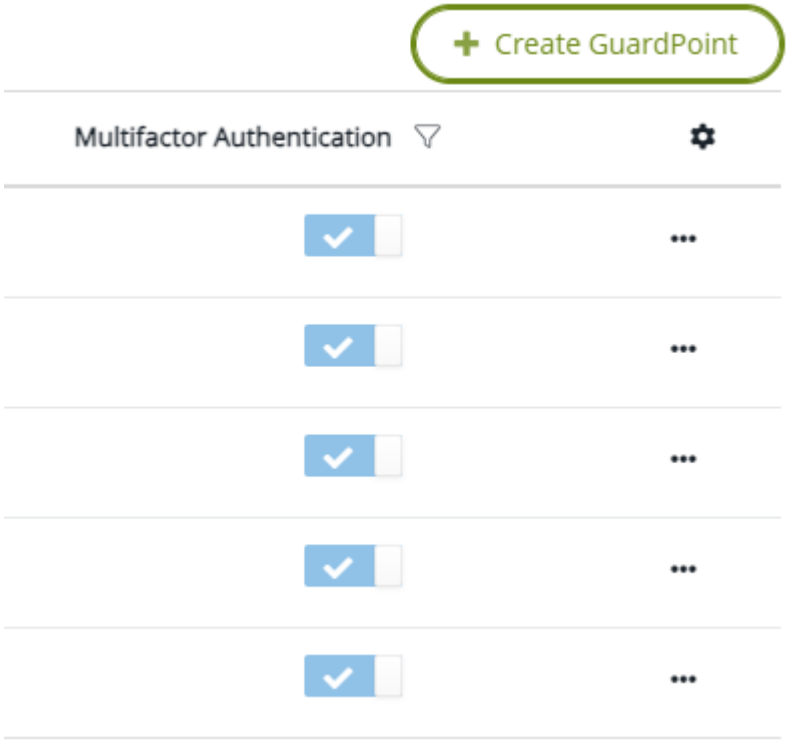
Enable Multifactor Authentication on the Entire Client

You can enable Multifactor Authentication for all of the GuardPoints on a client. When Multifactor Authentication is enabled at the client level, CTE enforces the configuration for all GuardPoints configured on the client. It overrides any MFA configuration set for individual GuardPoints.

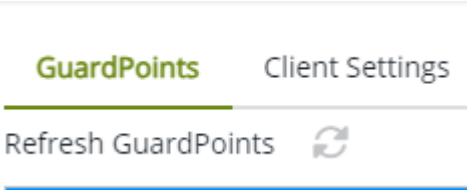
1. Open **CipherTrust Manager > Transparent Encryption** application.
2. Select the relevant client.
3. In the upper pane, select **Multifactor Authentication**.



4. Select **Apply**. All of the Multifactor Authentication switches are toggled to the **on** position.



5. If the MFA column doesn't display with all switches set to on, click **Refresh GuardPoints** to display the Multifactor Authentication column.



Note

To disable Multifactor Authentication on a GuardPoint, deselect Multifactor Authentication in the upper pane and click **Apply**.

Perform Multifactor Authentication

After you enable Multifactor Authentication on a single or multiple GuardPoints, there are multiple scenarios in which Multifactor Authentication can be authenticated for a user accessing the MFA-enabled GuardPoints on a client.

If Multifactor Authentication is enabled for a user, the user's current shell, and the following commands or programs running in the same shell, are allowed to access MFA-enabled GuardPoints until the user exits the current shell.

Note

If you only enable Multifactor Authentication on one GuardPoint, then you can only authenticate for that one GuardPoint. If you enable Multifactor Authentication on all of the GuardPoints on a client, you can then authenticate for all of the GuardPoints.

Note

- The CTE MFA username and password are created in Keycloak.
- Thales recommends using OTP (one-time password) to replace the static password. The OTP configuration on KeyCloak refers to KeyCloak configuration section. On CTE client side, the security administrator can run `voradmin mfa set_auth` to choose either OTP, or password, as the multifactor authentication method. If authentication method is not set up, users need to choose a method during ssh login or voradmin mfa login.

Selecting to run Multifactor Authentication at SSH connection

1. A sys-admin runs the `voradmin mfa ssh_enable` command and enables Multifactor Authentication for all SSH connections.

   ```
   voradmin mfa ssh_enable
   ```

2. Another user establishes an SSH connection to the client, and is asked if they want to run Multifactor Authentication. They choose `yes`. Multifactor Authentication runs.

   ```
   Do you need CTE MFA (y/n): y
   ```

   Response

   ```
   CTE MFA username: <username>
   CTE MFA password: <password>
   You passed CTE MFA. Can access CTE MFA-enabled GuardPoints
   ```

Note

If the security administrator has not setup an authentication method for the user, the login command prompt displays the following:

```
Choose authentication method (type p for password, or o for otp, default o):
```

If the user chooses **o**, the following prompt displays:

```
CTE MFA one-time code:
```

This change also applies when running `voradmin mfa login`.

Selecting to run Multifactor Authentication post login

1. A sys-admin runs the `voradmin mfa ssh_enable` command and enables Multifactor Authentication for all SSH connections.

   ```
   voradmin mfa ssh_enable
   ```

2. Another user establishes an SSH connection to the client, and is asked if they want to run Multifactor Authentication. They choose `no`. Multifactor Authentication does not run.

   ```
   Do you need CTA MFA (y/n): n
   ```

   Response

   ```
   You skipped CTE MFA. You won't be able to access CTE MFA-enabled GuardPoints.
   ```

3. When that user wants to run Multifactor Authentication, they use the command: `voradmin mfa login`. Multifactor Authentication runs.

```
voradmin mfa login
```

Response

```
CTE MFA username: <username>
CTE MFA password: <password>
CTE MFA authentication successful
You passed CTE MFA. Can access CTE MFA-enabled GuardPoints
```

Disabling Multifactor Authentication at SSH connection

1. A sys-admin runs the `voradmin mfa ssh_disable` command and disables Multifactor Authentication for all SSH connections.

```
voradmin mfa ssh_disable
```

2. When a non-root user wants to run Multifactor Authentication, they use the command: `voradmin mfa login`. Multifactor Authentication runs.

```
voradmin mfa login
```

Response

```
CTE MFA username: <username>
CTE MFA password: <password>
CTE MFA authentication successful
You passed CTE MFA. Can access CTE MFA-enabled GuardPoints
```

Enable Multifactor Authentication for Client Groups

Multifactor authentication cannot be enabled at the client group level. However, you can enable Multifactor Authentication for individual GuardPoints on client groups.

While propagating the Multifactor Authentication-enabled GuardPoints to the member clients, CipherTrust Transparent Encryption checks the Multifactor Authentication capability of the member clients. If a client is Multifactor Authentication-capable, the GuardPoints are added to the client. If a client is not Multifactor Authentication-capable, the GuardPoints are skipped.

Exempting some users from authentication with a Whitelist

When you activate CTE Multifactor Authentication, all users attempting to log in to the CTE client must successfully pass through CTE Multifactor Authentication to gain entry to the MFA-enabled GuardPoints. However, there are specific exemptions to this rule. Users whose system services, or applications, initiate automatically during system boot-up cannot undergo authentication though CTE Multifactor Authentication, as Multifactor Authentication necessitates user interaction. To accommodate such cases, exempt these users from Multifactor Authentication enforcement within the Client Profile by including them in the user set exemption list. Typically, only system users responsible for running system services are included in this list. However, since many system applications operate under the root user, or even under normal application user accounts, you must also add these system administrator/application users to the list. These users form part of the user set, commonly referred to as a whitelist.

Caution

Be careful when adding a user to the whitelist. The exemption applies for the entire client. Users on this list can bypass Multifactor Authentication and access all MFA-enabled GuardPoints. If a system service, or application, accesses only one GuardPoint among multiple GuardPoints, Thales advises you to leave that specific GuardPoint without Multifactor Authentication enforcement.

Note

You **cannot** share the whitelist between Windows and Linux operating systems. Each client profile must contain a unique Multifactor Authentication whitelist. Therefore, the User Set contains different users in Windows and Linux platforms. A CTE Windows client and a CTE Linux client **cannot** share the same client profile when Multifactor Authentication is enabled.

Creating a User Set

See Creating User Sets for information on creating a User Set in a Policy Element.

Adding the User Set to the Client Profile

To add an Multifactor Authentication whitelist to the client profile:

1. Create your Client Profile if it is not already created.
2. Click on your client profile to open it.
3. Click **Multifactor Authentication**.
4. In the **Select OIDC connection** field, select the OIDC connection that you created.
5. In the **Select the MFA exempted User Set** field, select the User Set that contains the people/applications that are exempted from authorization.
6. Click **Update**.

Setting up Multifactor Authentication with a One-Time-Password

CipherTrust Transparent Encryption Multifactor Authentication supports KeyCloak OTP through direct grant flow. This topic explains how to configure OTP support in KeyCloak.

Prerequisites

- Setup Keycloak

Enabling OTP Authentication in KeyCloak

1. Log in to the KeyCloak Admin Console. See KeyCloak documentation for more information.
2. Select **Authentication** from the menu for your CTE realm, e.g. cte-linux. This is the area where you can configure the different credential types.
3. Select the Browser Flow:
   a. Conditional OTP: **Required**
   b. Condition: User configured: **Required**
   c. OTP Form: **Required**
4. Modify, clone, or create a new Direct Grant flow:
   Option 1:
   Modify the built-in direct grant flow, or clone a direct grant flow, by clicking **Action** > **Duplicate**.
   Update the flow as the follows:
   a. Username Validation: **Required**
   b. Password: **Disabled**
   c. Direct Grant: Conditional OTP: **Required**
   d. Condition: User configured: **Required**
   e. OTP: **Required**

Option 2:

Create a new direct grant flow:

a. Select **Create Flow**

b. Fill out the Name and Description

c. Select flow type: **Basic Flow**

d. Select **Add step**

e. Select Username Validation: **Required**

f. Select **Add step**

g. Select OTP: **Required**

5. Bind the generated direct grant flow to the client defined for CTE Linux:
   a. Choose the client setup for CTE Linux, and select **Advanced**
   b. Select Browser Flow: **Browser**
   c. Select Direct Grant Flow: `<the new direct grant>`

Configuring OTP Policy

CTE Linux Multifactor Authentication only supports time-based OTP, which is the default KeyCloak OTP policy. To verify the policy configuration:

1. Navigate to the **CTE realm** > **Authentication**.
2. Select **Policies** > **OTP Policy**.
3. Ensure that Time-based is selected.

> **Note**
>
> Counter-based is **NOT** supported with CipherTrust Transparent Encryption.

4. Change other configurations as needed.

> **Note**
>
> Google authenticator only supports the algorithm: **SHA1**.

Setting Up User's OTP Authenticator

1. Add a user that has permissions to access CTE clients.

2. The user must install an authenticator that is able to provide an OTP token on their mobile phone.

> **Note**
>
> Thales recommends using Google Authenticator.

3. Instruct the user to login through a web browser to the CTE realm account. See KeyCloak documentation for more information.
4. Once authenticated, the OTP token form displays.
5. The user needs to finish the setup of OTP authentication in the OTP token form.

Conclusion

After a successful setup, the local users of the CTE Linux hosts can perform OTP authentication through `voradmin mfa login` or `SSH login`.

Setting up Multifactor Authentication with a Time Out

You can set Multifactor Authentication so that it times out after a specified period. The timeout value for an MFA session is set in minutes. The default is 0, meaning no timeout.

> **Note**
>
> MFA timeout only applies to new processes. Once a process, like Bash, for example, is MFA-authenticated, it does not time out. Additionally, when a timeout value changes, all MFA-authenticated processes remain authenticated. The timeout change does not affect them.

Administrator Tasks for Multifactor Authentication

Voradmin Commands

The Multifactor Authentication command group contains the following commands:

Syntax

```
    # voradmin mfa [ config | login | ssh_enable | ssh_disable |
set_auth | set_timeout]
```

voradmin mfa config

Displays configuration information.

Syntax

```
    # voradmin mfa config
```

Response

```
    Host Mfa Enable is not set.
    MFA enabled guardpath(s) (Number of paths: 2):
            /home/work1/gp_mfa
            /home/work1/gp_mfa
    MFA Exempt-List: (Number of entries: 1)
            uid:1001,       gid:1001,       user: "malfoy", group:
"wizards", domain(s): ""
    OIDC configuration:
            client-id : cte-client-malfoy-1
                    url : https://www.keycloak-sj1.org:8443/realms/cte-
linux/.well-known/openid-configuration
    ssh with MFA : enabled
```

> **Note**
>
> Root user privilege required.

voradmin mfa login

Enables Multifactor Authentication for the current shell, and commands and programs running in that shell, so that it can access MFA-enabled GuardPoints. You must provide a login name and password for Multifactor Authentication enablement.

Syntax

```
# voradmin mfa login
```

Response

```
CTE MFA username: ######
CTE MFA password: ######
CTE MFA authentication successful
```

voradmin mfa ssh enable

Enables the option to use Multifactor Authentication once a user has established an ssh login to the current host. After running this command, user is asked if they want to use Multifactor Authentication. Once logged in successfully, user is asked if they want to enable Multifactor Authentication. Once you enable MFA successfully, CTE Agent can access Multifactor Authentication-enabled GuardPoints.

Syntax

```
# voradmin mfa ssh_enable
```

> **Note**
>
> Root user privilege required.

voradmin mfa ssh_disable

Disables the option to use Multifactor Authentication once a user has established an ssh login to the current host.

Syntax

```
# voradmin mfa ssh_disable
```

Note

Root user privilege required.

voradmin mfa set_auth

Set up the authentication method (either OTP or password) to use during CTE multi-factor authentication. If not setup, users have to choose a method during ssh login or voradmin mfa login.

Syntax

```
# voradmin mfa set_auth
```

Response

Choose authentication method (type p for password, or o for OTP, default o):: o

MFA authentication method is set.

Note

Root user privilege required.

voradmin mfa set_timeout

Set a timeout value for an MFA session. The timeout value is set in minutes. The default is 0, meaning no timeout.

Syntax

```
# voradmin mfa set_timeout <minutes>
```

Example 1

```
# voradmin mfa set_timeout 5
```

Response

```
MFA session will expire after 5 minute(s).
```

Example 2

```
# voradmin mfa set_timeout 0
```

Response

```
Disabled MFA session timeout.
```

> **Note**
>
> Root user privilege required.

Logging

This chapter contains the following sections:

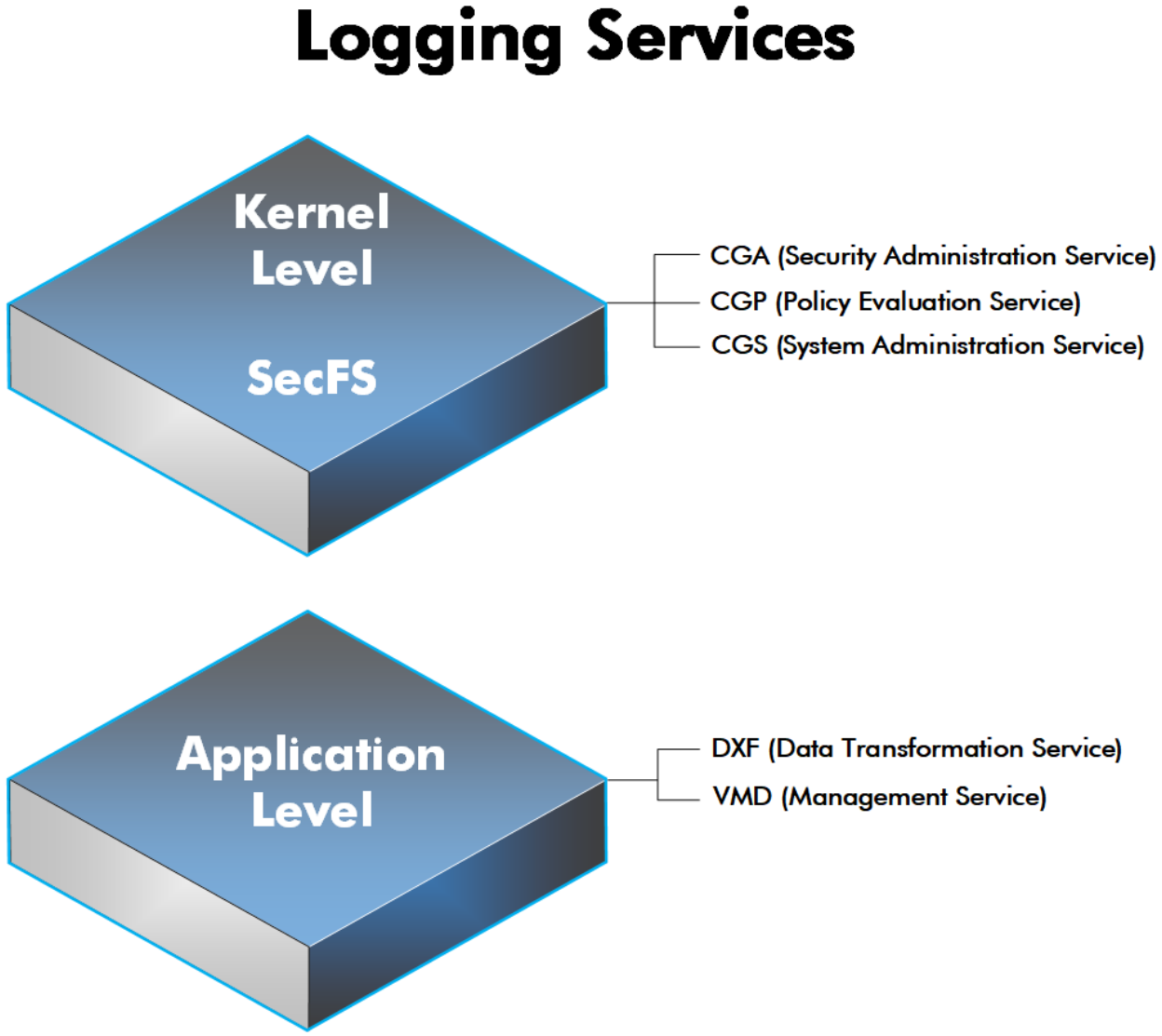
- Setting CTE Agent Logging Preferences
- Audit Logs
- Analyzing Audit log entries
- File System Audit Log Effects Codes
- Concise Logging

Setting CTE Agent Logging Preferences

You can configure the Agent process information that is entered into the Message Log. You can configure the process information globally, in which all the Agents that are added after the configuration change inherit the log attributes, while all current file system configurations remain intact. Alternatively, you can configure log attributes for individual Agent installations.

Always monitor log generation on new server and agent installations, and after changing logging preferences and options.

A variety of logging services are available and configured in the Log tab.



CTE log data may be sent to various different files such as:

- Sys log files, such as:

```
/var/log/messages

/var/log/syslog
```

  Event log on

> **Note**
>
> The CM domain name can include spaces. However, Syslog does not allow spaces in header fields. Therefore, for Syslog purposes, the CTE client replaces the spaces with an underscore. For example: My_Domain instead of My Domain.

- CTE log files local to the agent, such as:

```
/var/log/vormetric/vorvmd_root.log

C:\ProgramData\Vormetric\DataSecurityExpert\agent\log\vorvmd.log
(Windows)
```

- Uploaded to the Key Manager
- Uploaded to a Syslog server

Data Transformation log files are sent to:

```
/var/log/vormetric/vordxf_path_usr.log
```

Audit Logs

```
Example audit log:
CGP2601I: [SecFS, 0] [AUDIT] Policy[allowAllOps_fs]
User[root,uid=0,gid=0\root,bin,daemon,sys,adm,disk,wheel\] Process[/
bin/cat] Action[write_app] Res[/opt/apps/apps1/doc/file2.txt]
Key[aes128] Effect[PERMIT Code (1U,2U,3R,4M)]
```

Analyzing Audit log entries

The format of a File System Audit log entry is:

```
CGP2602I: [SecFS, 0] Level: Policy[policyName?] User[userID?]
Process[command?] Access[whatIsItDoing?] Res[whatIsItDoingItTo?]
Effect[allowOrDeny? Code (whatMatched?)]
```

Parameter	Description
Identifier	The TLA for the error message.
SECFS	Indicates that the message was generated by an Agent. You can enter `secfs` in the Search Message field in the Logs window to display the Agent policy evaluation and GuardPoint activity for all configured hosts.
Level	Indicates the importance of the message. For example, AUDIT indicates an informational message, whereas ALARM indicates a critical failure that you should not ignore.
Policy	Indicates the name of the policy that is being used to evaluate the access attempt.
User	Identifies the system user attempting to access data in the GuardPoint. It typically displays the user name, user ID, and group ID.
Process	Indicates the command, script, or utility being executed.
Access	Indicates what access is being attempted. Access may be read_dir, remove_file, write_file_attr, write_app, create_file, etc. These correspond to the Access methods that you configure in the policy. Read_dir corresponds to d_rd. Remove_file corresponds to f_rm, etc.
Res	Indicates the object/resource being accessed by the Process[].
Effect	Indicates the rule that matched and, based upon that rule, whether or not theCipherTrust Manager grants access. Access states may be either PERMIT or DENIED.

File System Audit Log Effects Codes

Codes are provided in the audit logs that identify actions by the policy enforcement engine. The code follows the number of the rule being processed.

Code	Definition
A	The Action component of a security rule failed to match.
M	All security rule components match and, unless overridden, the Effect for that security rule is applied.

Code	Definition
P	The Process component of a security rule failed to match.
R	The Resource component of a security rule failed to match.
T	The time specified in the When component of a security rule failed to match.
U	The User component of a security rule failed to match.

Refer to the audit log example above:

- The first and second Security Rules fail because of a mismatch in the User component (1U, 2U).
- The third Security Rule fails because of a Resource component (3R) mismatch.
- All of the rules in the fourth Security Rule match (4M), and the actions defined in the policy, such as use an encryption key, are applied.

Concise Logging

Thales's standard operational logging sends audit messages for each file system operation each time a file is opened, read, updated, or written. Thales's standard logging can generate high volumes of log data. Most of these messages might not be useful or required by security administrators to monitor file system activity on the system.

Agent log data can be stored on the local host, sent to a syslog server, or uploaded to the Management Console. On an agent system, log entries can flood the local messages file or Event Log. Extreme logging can also affect network performance.

Concise Logging eliminates the following types of messages:

- Duplicate audit messages for each and every block read by the user or application. With Concise Logging, CTE only sends an audit message the *first* time a user or application performs a read/write activity. Subsequent read/write activity by that user or application is not logged.
- Audit messages that read the attributes, read the basic information of file-set attributes, and other event-based messages.
- Audit messages for directory open, read directory attributes, and directory close.

Using Concise Logging

You can enable and disable the Concise Logging option from the CipherTrust Manager for the following:

- All registered hosts in all domains
- A host that has registered with the CipherTrust Manager.

Considerations

- Concise Logging changes the set of log messages that are sent to Security Information and Event Management (SIEM) software systems. If this results in loss of data required for customer reports, then disable Concise Logging.
- Concise Logging is only supported by CTE `secfs`.
- Enable and disable Concise Logging on the client, in the Client Profile. CTE applies it for all users of that Client Profile. There is no finer-grained control, such as per GuardPoint, user, or message type.
- Do not use Learn mode with Concise Logging.

Configuring Concise Logging for CTE Clients or Client Groups with CipherTrust Manager

In CipherTrust Manager, when you create a Client Profile, you can select to Enable Concise Logging. Then, you can apply that Client Profile to a specific client, or to all clients in a Client Group. To enable Concise Logging, see the chapter *Managing Profiles in the CTE Administrator Guide*.

Special Cases for CTE Policies

This section describes some CTE-specific configuration tasks related to configuring policies in the key manager. It contains the following topics:

Compatibility

- Starting with VTE for Linux version 6.1.0, CTE is backward compatible with, and fully supports, the existing AES-CBC mode for both new and existing datasets.
- Starting with VTE for Linux version 6.1.0, CTE fully supports AES-CBC-CS1 encryption for LDT and offline data transformation on CTE Linux environments.

  Versions of VTE prior to version 6.1.0 are not backwards compatible with AES-CBC-CS1 encryption. On these earlier versions, attempting to guard a device using a policy containing an AES-CBC-CS1 key will fail.
- Protected hosts supporting AES-CBC-CS1 encryption can be added to host groups.

Difference between AES-CBC and AES-CBC-CS1

The two encryption modes are completely different from a file format standpoint.

- AES-CBC-CS1 encryption only applies to file system directories; AES-CBC encryption applies to both files and block devices.

Note

- If you attempt to use an AES-CBC-CS1 key to guard a block device or partition, the guarding fails with an error reported on the CipherTrust Manager, similar to: Raw or Block Device (Manual and Auto Guard) GuardPoints are incompatible with Policy “policy-xxx" that contains a key that uses the CBC-CS1 encryption mode.”
- While AES-CBC-CS1 encryption is supported on both Linux and Windows environments, the file formats are incompatible. An encrypted file created with a specific AES-CBC-CS1 key on Windows cannot be read on Linux, even if that specific key were to be used and vice versa.

- AES-CBC-CS1 uses cipher-text stealing to encrypt the last partial block of a file whose size is not aligned with 16 bytes.
- Each file encrypted with an AES-CBC-CS1 key is associated with a unique and random base IV.

- AES-CBC-CS1 implements a secure algorithm to tweak the IV used for each segment (512 bytes) of a file.

Disk Space

Files encrypted with AES-CBC-CS1 keys consume additional disk space in contrast to files encrypted with AES-CBC keys. This is because AES-CBC-CS1 encryption requires file IVs to be created and persistently stored in contrast to AES-CBC encryption which does not consume any additional disk storage.

Therefore, administrators need to plan and provision additional disk capacity prior to deploying AES-CBC-CS1 encryption.

	AES-CBC	AES-CBC-CS1
Local Linux (CTE)	No change to file size. No extended attribute allocation	Internal use of extended attribute per file. Extra 4KB increase in file size.
Local Linux (CTE-U)	Internal use of extended attribute per file. Extra 4KB increase in file size.	Internal use of extended attribute per file. Extra 4KB increase in file size.
Remote Linux (CTE)	No change to file size. No extended attribute allocation	Extra 4KB allocation in the form of an embedded header per file. With CTE guarding enabled, file size expansion is hidden.
Remote Linux (CTE-U)	Internal use of extended attribute per file. Extra 4KB increase in file size.	Internal use of extended attribute per file. Extra 4KB increase in file size.

Encryption Migration

You can use either LDT or offline dataxform to:

- Transform data encrypted by AES-CBC to AES-CBC-CS1 and vice versa.
- Transform AES-CBC-CS1 encrypted data to clear contents and vice versa.

File Systems Compatibility

On Linux, you can use AES-CBC-CS1 keys to guard currently supported file systems.

AES-CBC-CS1 encrypted files on Linux remote file systems like NFS and CIFS increase the file size compared to encrypted files on Linux local file systems which retain the original file size.

AES-CBC-CS1 encrypted files on Linux local file systems, in conjunction with LDT policies, can result in additional space consumption. Unlike the current AES-CBC encryption where encrypted files on all file systems, both remote or local, have the same file format, AES-CBC-CS1 encrypted file formats differ based on whether or not they were created on local or remote file systems.

AES-CBC-CS1 files on Linux remote file systems such as NFS and CIFS embed the IV in a 4K-byte header within the file. When these files are guarded, CTE masks the file header -- to applications and system utilities. The expanded file is only apparent when CTE guarding is disabled.

Note

The remote file system must have enough extra space to store the extra 4K bytes of the embedded header. You can run the following script in the GuardPoint to identify how much space to reserve before data transformation:

```
x=$(find . -type f | wc -l); y=$(echo "$x * 4 /1024" | bc); echo ${y}MB
```

File System Requirements

CTE Only

Unlike with AES-CBC encryption, files encrypted with AES-CBC-CS1 on remote file systems cannot be copied over to local file systems in the absence of CTE guarding. Similarly, AES-CBC-CS1 encrypted files on local file systems cannot be copied over to remote file systems in the absence of CTE guarding.

The fundamental reason for this incompatibility is the usage of extended attributes on local file systems to store the IV, in contrast to its storage as a part of the file metadata on remote file systems. This is why files cannot be transferred across these file system boundaries in the absence of CTE guarding.

Note

In RHEL 6, the `user_xattr` mount option needed to be provided manually if the files were encrypted with CTE AES-CBS-CS1 encryption mode. This is no longer required. Thales does not support RHEL 6 anymore. Therefore, the File Systems Compatibility table has been removed to make the document simpler and easier to understand.

Storing Metadata

AES-CBC-CS1 encrypted files on Linux store the base IV of a file in either the extended attributes or in the file metadata:

- On local FS (EXT3/EXT4/XFS), Linux saves it as an extended attribute associated with the file. The underlying file system requires that you mount it with the extended attribute mount option.
- On remote FS (NFS and CIFS), Linux saves it in the embedded header of the file.
- For CipherTrust Transparent Encryption UserSpace, all files have an embedded header.

To get the value of the base IV, type:

```
voradmin secfs iv get <file-name>
```

Note

The base IV of a file is protected. It cannot be set/modified/removed by commands and applications. However, if a GuardPoint is unguarded, the files in the GuardPoint are no longer protected. An adversary can then corrupt the content of the files, as well as the IVs.

AES-CBC-CS1 depends on the physical file system's support for extended attributes in a manner similar to the CipherTrust Transparent Encryption - Live Data Transformation feature.

Missing IV file

If the IV for a file is missing, or CTE is unable to read the IV, then CTE denies access to the file. This access denied message may trigger an application to display an error message. This message may vary from application to application.

	AES-CBC	AES-CBC-CS1
Local Linux FS	No change	Internal extended attribute for each file
Remote Linux FS	No change	4KB embedded header for each file

HDFS

CTE Only

The AES-CBC-CS1 key is compatible with the current Hadoop File System support.

Backups

Backups and other data protection utilities should take into account the extended attributes present in each AES-CBC-CS1 encrypted file on a Linux local file system to ensure that they are safely backed up. An AES-CBC-CS1 encrypted file whose IV is corrupted, renders the files to be corrupted and therefore unreadable. Hence all data protection software must preserve the file's extended attributes.

CTE Linux can inspect a file's IV using the following command:

```
voradmin secfs iv get file
```

CTE Only

On Linux, the backup utility specified in the guarding policy should automatically backup/restore extended attributes as well. For example, you must use the options to preserve extended attributes when running `cp` or `rsync.normal`.

Due to the different file formats, the backup/restore process across the local and remote file systems are not allowed. If you want to backup a GuardPoint from a local directory, you must restore it to a local directory. If a GuardPoint is backed up on a remote file system, you must restore it to a remote file system.

Backup/Restore Process	AES-CBC	AES-CBC-CS1
General backup utility requirement (all platforms)	Backup utility defined in guarding policy with clear view	Backup utility defined in guarding policy with clear view
Special requirement for backup/ restore local fs on Linux	No	Backup utility must be run with user extended attribute enabled
Special requirement for backup/ restore remote fs on Linux	No	No
On Linux, backup local fs and restore to remote fs	Allowed	Not allowed, the restored files cannot be accessed with I/O error
On Linux, backup remote fs and restore to local fs	Allowed	

Backup/Restore Process	AES-CBC	AES-CBC-CS1
		Not allowed, the restored files cannot be accessed with I/O error

Container Compatibility

The CBC-CS1 key is compatible with current Docker and OpenShift support.

Using the AES-CBC-CS1 Encryption Mode in CM

When you create a key in CTE, you enable Encryption Mode by selecting CTE Key Properties. See *Creating a New Key* in the Managing Policieschapter in the CTE Administrator Guide.

Exceptions and Caveats

Note the following when using AES-CBC-CS1 keys.

Ensure User Extended Attributes are Enabled on RHEL 6

On RHEL6, EXT3/EXT4 are not mounted with user extended attributes enabled, by default. If a GuardPoint is on EXT3/EXT4, then remount EXT3/EXT4 with `user_xattr` as an option. Otherwise, guarding fails with the error "Extended attribute not enabled for GuardPoint."

Guarding Existing Files Without Data Transformation

You must convert an existing file with clear text through offline data transformation or LDT. If you do not transform the file, then after you guard using an AES-CBC key, the file displays garbled characters.

If you use an AES-CBC-CS1 key, access to the file is blocked with an I/O error.

Best Practices for AES-CBC-CS1 Keys and Host Groups

In a host group, do not deploy policies associated with AES-CBC and AES-CBC-CS1 keys unless all hosts are running VTE for Linux version 6.1.0 or CTE version 7.0.0 or later.

CTE and systemd

CipherTrust Transparent Encryption (CTE) for Linux is integrated with the systemd framework. To ensure that applications start after the CTE agent starts at startup, you must modify `systemd`. This is also true when the CTE agent is started and stopped manually.

This section contains the following topics:

Overview of CTE and systemd

You can use `systemd` to configure dependent applications to start up or shut down in the proper order when CTE starts up or shuts down on a live system. If applications start before CTE starts, those applications may not be guarded. This applies to cases when you manually start or stop the system, such as when you upgrade CTE.

`systemd` replaces `init` in Linux as a system and service manager. Linux inherited init from the UNIX System V `init. systemd`. The file is a collection of daemons, libraries, and utilities to provide central management and configuration for certain Linux distributions (see Linux Distributions that Support CTE and systemd). In addition to providing enhanced features and performance, `systemd` is backwards compatible with System V and Linux Standard Base `init` scripts.

Linux Distributions that Support CTE and systemd

The entries in the following table are Linux distributions that are compatible with CTE. The table shows which distributions implement `systemd`, and those that do not. See the *Compatibility Matrix for CTE Agent with CipherTrust Manager* or the *Compatibility Matrix for CTE Agent with Data Security Manager* for the Linux distributions and kernels supported with your CTE version and key manager.

Distributions that support systemd	**Distributions that do not support systemd
RHEL 8	SLES
SLES 12	
SLES 15	
Ubuntu 16.04	
Ubuntu 18.04	

CTE Agent Control Changes on systemd

The commands to start, stop, restart, and check CTE status on `systemd` are shown in the following table:

Command	Command syntax for distributions that support systemd
Start	`/etc/vormetric/secfs start`
Restart	`/etc/vormetric/secfs restart`
Stop	`/etc/vormetric/secfs stop`
Check status	`/etc/vormetric/secfs status`

The normal states of CTE services on a system with one or more active GuardPoints is the following:

- `secfs-fuse`: active (exited) state

Example

To check the status of CTE, type:

```
/etc/vormetric/secfs status
```

Result secfs-fuse service: active (running) since Tue 2022-09-27 09:51:04 CDT; 1h 3min ago

CTE Configuration Changes Required on systemd

The following table is a high-level overview of the required changes in `systemd` unit configuration files to control the applications that must be in sync with CTE during system startup and shutdown.

Typical applications that require `systemd` changes include `postgres, httpd, mongodb, mysqld` and `mariadb`. For example, `mongodb` requires that CTE to be running before `mongodb` starts.

Caution

Configuring the dependencies as recommended here mean that whenever you stop CTE or if CTE stops unexpectedly, the dependent applications will automatically stop shortly before CTE stops. Be sure you understand the implications of this behavior on your production environment.

Task	For more information
Compile a list of your applications that use GuardPoints.	See the application examples above.
Shut down the applications that use GuardPoints.	Adding Dependencies to systemd Unit Configuration Files

Add the following lines to the `[Unit]` section of the `systemd` unit configuration file for each application:
`After=secfs-fs-barrier.service`
`BindsTo=secfs-fs-barrier.service`
	Adding Dependencies to systemd Unit Configuration Files

To the Before= line in the [Unit] section of the `secfs-fs-barrier.service` file, add the following: `saslauthd.service`	Adding Applications to the secfs-fs-barrier.service File

Task	For more information
An entry for each application you added to the `systemd` unit configuration file.	

Add the following lines to the `[Unit]` section of the `saslauthd.service` configuration file:
`BindsTo=secfs-fs-barrier.service`
`After=secfs-fs-barrier.service`
	Adding Dependencies to the saslauthd.service File

Force the system to read the changed systemd configuration files by typing `systemctl daemon-reload`.	Adding Dependencies to the saslauthd.service File
Restart the applications that you shut down to make these changes.	Adding Dependencies to the saslauthd.service File

About systemd Dependency Changes for Unit Configuration Files

The previous table describes the effect of the required unit configuration file changes that are described in the following sections. See the systemd documentation for more information about the `After=` and `BindsTo=` unit configuration options.

Caution

In some cases, the unit configuration file for an application other than CTE may be overwritten when the application is upgraded. After upgrading an application, verify that changes to that application's unit configuration file are still in place. Changes to the `secfs-fs-barrier.service` file are retained after you upgrade CTE.

The `secfs-fs-barrier.service` service ensures that CTE services and dependent applications start after CTE services and stop before CTE services as needed during system startup, shutdown, and when starting and stopping services on a live system. This special service manages the dependencies between applications and the `secfs-fs, secfs-init, secfsd`, and `vmd` services that make up the CTE agent:

File and Change	Purpose
File: Application-specific unit configuration file Add this line to `[Unit]` section: `After=secfs-fs-barrier.service`	In conjunction with the `BindsTo=` option, this option ensures that the application starts after the CTE agent on system startup.
File: Application-specific unit configuration file Add this line to `[Unit]` section: `BindsTo=secfs-fs-barrier.service`	If the CTE agent shuts down, the application also shuts down.
File: `secfs-fs-barrier.service` Add each application to the existing `Before=` line in the `[Unit]` section. Add `saslauthd.service` to the existing `Before=` line in the `[Unit]` section.	Ensure that the `secfs-fs-barrier.service` starts before the application services mentioned in the `Before=` l

Adding Dependencies to systemd Unit Configuration Files

If your system supports `systemd` (see Linux Distributions that Support CTE and systemd), before you can safely reboot your protected host or use the files in the GuardPoint, you must perform the following steps to set the proper CTE dependencies for your applications.

1. Compile a list of your applications that use GuardPoints.
2. Prepare users of those applications for the interruption in service required to make these changes.
3. Shut down each of the affected applications.
4. For each application, log in as root and open the unit configuration file for that application using a text editor such as `vi`. See the table above to determine the location of the `systemd` unit file.
   For example, the unit configuration file for a hypothetical `exampled` application would be located in `/usr/lib/systemd/system/exampled.service`.
5. Locate the `[Unit]` area in the file and add the following two lines at the end of the `[Unit]` section:

   ```
   After=secfs-fs-barrier.service
   BindsTo=secfs-fs-barrier.service
   ```

For example, the `exampled` application unit configuration file might have the following existing `[Unit]` section:

```
[Unit]
Description=Example server
After=syslog.target
After=network.target
```

In this case, you would add the two new lines after `After=network.target`.

6. Save and close the unit configuration file.
7. Repeat steps 3–6 for each application.

Continue to Adding Applications to the secfs-fs-barrier.service File to make necessary changes to `secfs-fs-barrier.service`.

Adding Applications to the secfs-fs-barrier.service File

After editing the `systemd` unit configuration file for each dependent application, you must also add each application to the `secfs-fs-barrier.service` file that is installed as part of CTE. The `secfs-fs-barrier.service` file ensures that CTE starts before dependent applications during boot and CTE stops before dependent applications when the system is shut down.

1. As a root user, open the `secfs-fs-barrier.service` file in a text editor such as `vi`.
   For example, the `secfs-fs-barrier.service` file is located in `/usr/lib/systemd/system/secfs-fs-barrier.service`.
2. Add `saslauthd.service` along with the names of the unit configuration files that you edited in the last section to the end of the "`Before=`" clause.
   For example, the `secfs-fs-barrier.service` file might contain the following before editing:

   ```
   Before=postgresql.service httpd.service mongodb.service mongod.service mysqld.service mariadb.service nails.service
   ```

   To add an entry for `saslauthd.service` and two custom application services called `example1.service` and `example2.service`, you would edit the `Before=` line to look like this:

`Before=postgresql.service httpd.service mongodb.service mongod.service mysqld.service mariadb.service nails.service saslauthd.service example1.service example2.service`

3. Save and close the `secfs-fs-barrier.service` file.

In the future, CTE will start before each application listed in the `Before=` line and those applications will stop before CTE stops.

Continue to the next section to make the necessary changes to `saslauthd.service`.

Adding Dependencies to the saslauthd.service File

The `saslauthd` service must start after the `secfs-fs-barrier` service.

1. As a root user, open the `saslauthd.service` file in a text editor such as `vi`.
   For example, the `saslauthd.service` file is located in `/usr/lib/systemd/system/saslauthd.service`.
2. Locate the `[Unit]` area in the file and add the following two lines at the end of the `[Unit]` section:
   `After=secfs-fs-barrier.service`
   `BindsTo=secfs-fs-barrier.service`
   For example, the `saslauthd.service` file might have the following existing `[Unit]` section:
   `[Unit]`
   `After=syslog.target`
   `After=network.target`
   In this case, you would add the two new lines after `After=network.target`.
3. Save and close the `saslauthd.service` file.
4. To force the system to re-read the `systemd` configuration files that you changed in this section and the previous section, type `systemctl daemon-reload`.
5. Start each application that you shut down to make the configuration changes.

Location of Application Unit Configuration Files

To configure the proper startup order of services, modify the unit configuration file for applications that require CTE to be running. You must also modify the `secfs-fs-barrier.service` file. The location of these files varies between Linux distributions as described in the following table.

Distribution	Location of systemd unit configuration files
RHEL 8 SLES 12 SLES 15	`/usr/lib/systemd/system/<application_name>.service`
Ubuntu 16.04 Ubuntu 18.04	`/lib/systemd/system/<application_name>.service`

Drop-in Files for systemd

As an alternative to adding applications to the barrier file, you can use drop-in files to configure the `secfs-fs-barrier.service` file for dependency management using `systemd`. This eliminates the need to modify the `secfs-fs-barrier.service` file for configuring dependencies.

All drop-in directory files with the suffix `.conf` are merged in alphanumeric order and parsed after the main unit file has been parsed. Each drop-in file must contain appropriate section headers. `Systemd` reads the `.conf` files in the following order.

1. The instance `.d/` subdirectory
2. The template `.d/` subdirectory

The drop-in `.d/` directories for system services can be placed in the following directories:

- `/etc/systemd/system`
- `/usr/lib/systemd/system` (Ubuntu only)
- `/run/systemd/system`

Drop-in files in `/etc/` take precedence over those in the `/run/` directory which, in turn, take precedence over those in the `/usr/lib/` directory. Drop-in files in any of those

directories take precedence over unit files wherever they are located. Multiple drop-in files with different names are applied in lexicographic order, regardless of which directories they reside in.

Following is an example of a drop-in file (`dependencies.conf`) for the `secfs-fs-barrier.service`:

```
cat /usr/lib/systemd/system/sec-fs-barrier.service.d/dependencies.conf
[Unit]
Before=mfetpd.service mfeespd.service
```

The above example displays a drop-in file which adds two McAfee related services in the **Before**= clause of the `secfs-fs-barrier.service` file without modifying the service file itself. Eventually, `systemd` detects the drop-in files and links them to the `secfs-fs-barrier.service` file.

Utilities for CTE Management

Thales provides a variety of utilities that augment the standard Linux utilities. This combination of tools helps administrators manage CTE. The following utilities are described in this section:

- Agent Health Return Codes
- Agentinfo Utility (Java version)
- Backup Utility
- Check_host Utility
- Displaying Information for Nested File Systems with the DF tool
- Fsfreeze and xfs_freeze (CTE only)
- Protecting Files with Client Settings (CTE only)
- Restricting Access Overrides with Client Settings
- Register_host Utility
- Secfsd Utility
- Using Advanced Encryption Set New Instructions (AES-NI)
- User Cache Lookup Improvements
- Vmutils
- VMD

- Vmsec

Agent Health Utility

The `agenthealth` utility validates:

- Super-user privilege
- CTE Agent installation
- CTE registration to CipherTrust Manager Server
- CTE processes/ modules that are running
- Available disk resources
- Current GuardPoints
- Tests if the agent can reach the GuardPoints
- CTE log directory resource status
- This directory contains pending CTE log files for upload. This utility reports the size and number of pending files for upload. These text files are logs that contain vmd/SecFS information. They are regenerated whenever secfs restarts. If the number of files is unexpectedly large, this can indicate a problem.

The Agent Health check script

By default, the `agenthealth` script is installed in `/opt/vormetric/DataSecurityExpert/agent/vmd/bin`.

To run the `agenthealth` check script, type:

```
./opt/vormetric/DataSecurityExpert/agent/vmd/bin/agenthealth
```

System Response

```
Checking for super-user privilege ..............  OK
CipherTrust Agent installation ..................  OK
CipherTrust policy directory ....................  OK
Registration to server ..........................  OK
Kernel modules are loaded .......................  OK
VMD is running ..................................  OK
SECFSD is running ...............................  OK
dsm4209.sjinternal.com is resolvable ............  OK
```

```
dsm4209.sjinternal.com port 8446 is reachable .... OK
dsm4209.sjinternal.com port 8447 is reachable .... OK
Can communicate to at least one server ........... OK
VMD is listening on port 7024 .................... OK
Time of last update from server .................. 2021-07-07
15:47:08.290
Checking available disk space .................... OK
Checking logging space ........................... OK
        Log directory is "/var/log/vormetric"
        File system for log data is "/", 48G free (5% full)
        Log directory contains 9 of maximum 200 files (4% full)
        Log directory contains 1 of maximum 100 Mbytes used (1% full)
Testing access to /media ......................... OK
Testing access to /usr/data/sub1 ................. OK
[root@agt4206 bin]#
```

Agent Health Return Codes

Previously, the agent health return codes were only available in `/var/log/vormetric/agenthealth.log`. Now, the following options are also available through the help pages:

Help

This agent health script checks various facets of the CipherTrust agent to make sure that everything is functioning properly. Results are also logged to `/var/log/vormetric/agenthealth.log`.

Syntax

```
./opt/vormetric/DataSecurityExpert/agent/vmd/bin/agenthealth --help
```

Return Codes

Use the return code option to get a list of the return codes and what they mean. The codes are returned if the Agent is not running.

Syntax

```
./opt/vormetric/DataSecurityExpert/agent/vmd/bin/agenthealth --
return_codes
```

Response

Return Code	Definition
EPERM	User is not root.
ENOENT	One of the programs used in this script does not exist. See `/var/log/vormetric/agenthealth.log` for which program is missing.
ENOPKG	Agent software is not properly installed. Agent configuration directory is missing or corrupt. See `/var/log/vormetric/agenthealth.log` for more details.
EPROTO	Agent is not registered to a key manager. Register the agent to a key manager and try again. Try the wait option if the a gent has never started correctly after registration. See `/var/log/vormetric/agenthealth.log` for more details.
EIO	Kernel modules are not loaded. To load a kernel module, type: `/etc/vormetric/secfs start`
ESRCH	VMD is not running. To start vmd manually, type: `/usr/bin/vmd`
SECFSD	Secfsd is not running. To start the secfsd manually, type `/usr/bin/secfsd`
EHOSTUNREACH	Unable to reach the Key Manager. Check network connectivity.
ECONNREFUSED	VMD is not listening. VMD did not finish initialization. See `/var/log/vormetric/vmd.log`
EWOULDBLOCK	VMD is attempting to connect to the Key Manager but has exceeded the designated wait time. Check `/var/log/vormetric/vmd.log` to fix any issues and retry.

Wait Time

Use --w to set a maximum wait time in seconds. The minimum is 10 seconds to test for the VMD to Key Manager initial contact. The default setting is 0, which means that there is no wait. Maximum is 1200 seconds.

Syntax

```
[root@agt4206 bin]# ./opt/vormetric/DataSecurityExpert/agent/vmd/bin/
agenthealth --w <value>
```

Example

```
[root@agt4206 bin]# ./opt/vormetric/DataSecurityExpert/agent/vmd/bin//
agenthealth --w 60
```

Response

```
Checking for super-user privilege ..... OK
CipherTrust Agent installation ..... OK
CipherTrust policy directory ..... OK
Registration to server ..... OK
Kernel modules are loaded ..... OK
VMD is running ..... OK
SECFSD is running ..... OK
dsm148.i.vormetric.com is resolvable ..... OK
dsm148.i.vormetric.com port 8446 is reachable .... OK
dsm148.i.vormetric.com port 8447 is reachable .... OK
Can communicate to at least one server ..... OK
VMD is listening on port 7024 ..... OK
Time of last update from server ..... 2021-08-18
10:34:56.665
Checking available disk space ..... OK
Checking logging space ..... OK
Log directory is "/var/log/vormetric"
File system for log data is "/", 29G free (23% full)
Log directory contains 1 of maximum 200 files (0% full)
Log directory contains 0 of maximum 100 Mbytes used (0% full)
```

If the customer did not use the wait time options, the output would look similar to the following:

```
/opt/vormetric/DataSecurityExpert/agent/vmd/bin/agenthealth
Checking for super-user privilege ..... OK
CipherTrust Agent installation ..... OK
CipherTrust policy directory ..... OK
```

```
Registration to server ..... OK
Kernel modules are loaded ..... OK
VMD is running ..... OK
SECFSD is running ..... OK
Can communicate to at least one server ..... FAILED
For more information consult the log file /var/log/vormetric/
agenthealth.log
```

agentinfo Utility (Java version)

The `agentinfo` utility collects system and CTE configuration data. The `agentinfo` utility is used to take a configuration snapshot of the system that you will send to Thales Customer Support to debug an issue, (This section describes the Java version.)

The `agentinfo` utility is a Java Script file. You can open it in a text editor to see specific functions.

The `agentinfo` utility displays status information on the screen and outputs the results to a compressed tar file. The compressed tar file name format is `ai.<os_name_ver>.qa.com.tar.gz` and it is located in the current working directory.

To create an agentinfo file, type:

```
/opt/vormetric/DataSecurityExpert/agent/vmd/bin/agentinfo
```

Backup Utility

When a backup is performed, certain files and directories may be protected against access. Therefore, those files and directories are not written to the backup repository. These include files located in Data Transformation GuardPoints or files in GuardPoints with appropriate policies. Additionally, the following files are locked by default by CTE agent:

```
/opt/vormetric/DataSecurityExpert/agent/secfs/.sec/.access
/opt/vormetric/DataSecurityExpert/agent/secfs/.sec/etc/*
/opt/vormetric/DataSecurityExpert/agent/secfs/.sec/pem/*
```

The following describes how to bypass the issue for multiple scenarios:

Agent is installed in the default location

1. Stop SecFS, type:

```
/etc/vormetric/secfs stop                //Linux

/etc/rc.d/rc2.d/S99secfs stop            //AIX
```

2. Run the backup application with the desired arguments.
3. Restart SecFS, type:

```
/etc/vormetric/ start
```

Using a backup image to install to other agents or restore to a different system

Note

When the image is used to reinstall the system, the agent will automatically start at system boot and will attempt to connect to the key manager to which it was originally registered.

To prevent multiple systems with the same agent ID, you must uninstall CTE from the system before running the backup application. The restore/install from the backup will not have an agent running.

1. Uninstall the agent, type

```
/opt/vormetric/DataSecurityExpert/agent/secfs/bin/uninstallsfs
```

2. Run the backup application with the desired arguments.

3. Re-install CTE agent.

Performing a backup while the agent is running

Before running the backup application, add the files that are protected by the agent to the exclusion rules to exclude them from the backup:

```
/opt/vormetric/DataSecurityExpert/agent/secfs/.sec/.access
/opt/vormetric/DataSecurityExpert/agent/secfs/.sec/etc/*
/opt/vormetric/DataSecurityExpert/agent/secfs/.sec/pem/*
```

The GuardPoint policy may implement access restrictions which would also cause the backup application to generate error messages. These GuardPoint/directories will also need to be added to the exclusion rule method to exclude them from the backup. Alternatively, you can temporarily unguard them while the backup application is running.

If the CTE agent reports a status of incomplete in the backup application and does not start properly, or partially starts but generates error messages at system boot time, then uninstall and reinstall the agent. The restore/clone image contains everything needed to uninstall the agent. Use the following command to perform this operation:

```
    /opt/vormetric/DataSecurityExpert/agent/secfs/bin/uninstallsfs
```

Backing up Databases after Encryption

After encrypting a database, CipherTrust Transparent Encryption cannot make a backup of the database. Both scheduled and manual backup fail. The problem was the user's policy. A policy used in this scenario must follow a few rules.

With a CBC_CS1 key, a guarded file is modified to have a 4096 byte header holding key information. When an **Apply Key** effect is specified, the CipherTrust Transparent Encryption code adjusts the length and file offset for this header. Without an **Apply Key** effect, the size and access of the offset include the CBC_CS1 header.

Thales recommends that you modify the first rule of your policy. Remove the action entry for `f_rd_att` from the first rule and add a new rule before it:

```
**action**: f_rd_att

**effect**: Permit, Apply Key
```

Policy processing starts with the first rule and continues until a matching rule is found. The effect for the matching rule is then applied.

For the `f_rd_att` action, this results in the secfs code including the CBC_CS1 key header and adjusts the file size value. Without the Apply Key effect, the file size includes the CBC_CS1 header size and the file appears as 4096 bytes larger than its real size.

check_host Utility

If a CTE software installation fails during the certificate generation and exchange stage, use the `check_host` utility to list the network addresses for the host. The utility checks network interfaces, `/etc/hosts`, DNS, and so on, to compare, test, and evaluate possible addresses for the host, and weights them based upon their network efficiency. FQDNs are the most preferred and stand-alone IP addresses are the least preferred. Some applications, such as silent-mode installation, use `check_host` to determine the best host address to submit to the CipherTrust Manager during registration.

Run the `check_host` utility on a system that is hosting CTE to display available network host names, FQDNs, and IP numbers for the host.

Type:

```
/opt/vormetric/DataSecurityExpert/agent/vmd/bin/check_host
```

check_host Syntax

```
check_host [[-h | -i | -a ] [-s name]] |
-l name:port[,name:port] | -r name
```

Syntax	Description
-h	Print the best host name for this machine
-i	Print the best IP address
-a	Print all the host names and IP addresses
-s	The name of the server (optional hint)

Syntax	Description
-r	The name of the server for name resolution checks
-l	The name and port of the server for listening checks

Displaying Information for Nested File Systems with the DF tool

When a file system mounts on top of another file system, the command `df -a` does not display the attributes for the covered file system.

On Linux environments, this is the expected behavior for any file system that is overlayed by another file system.

The secfs driver properly handles the call, which is made by the `statfs` system call, which is issued by the df tool. When the system call returns, its structures are correctly populated with the details of the nested file system. On examining the source of the df tool, it is found that when the `-a` switch is on, it nullifies the stats which are received for overlayed mounts. When the `-a` option is not enforced, the stats are maintained.

Issue the `df` command for a specific mount point, for example `df /xfs/nested-xfs` (where `/xfs` is a secfs GuardPoint). It works correctly.

fsfreeze and xfs_freeze

Users can freeze, snapshot, and unfreeze a file system with an SecFS GuardPoint using `fsfreeze|xfs_freeze` for both XFS and EXT3/4.

SecFS supports freezing with `fsfreeze|xfs_freeze` or by any other program issuing the same type of requests. Freezing SecFS results in freezing the underlying file system, as well as the primary file system.

Restrictions

There are restrictions for using `fsfreeze|xfs_freeze` support with CTE.

Platform Restrictions

The following platform restrictions occur with CTE and `fsfreeze|xfs_freeze`:

- CTE supports the `fsfreeze|xfs_freeze` utility for freezing SECFS GuardPoints on all Linux distributions for kernels >=3.0 for Redhat, SLES, and Ubuntu platforms on EXT3/EXT4/XFS file systems. (Earlier Kernels do not contain the proper freeze_super VFS code).

Target Restrictions

The expected target of the `fsfreeze|xfs_freeze` command is the path of the GuardPoint.

For example, if `/dev/sdb` is mounted as ext4 on `/data` and CTE contains the GuardPoint: `/data/protected`, then the target of `fsfreeze` must be `/data/protected`, not `/data`.

Valid:

```
fsfreeze -f /data/protected
```

Not valid:

```
fsfreeze -f /data
```

File System Restrictions

The following file system restrictions occur with CTE and `fsfreeze|xfs_freeze`:

- If multiple GuardPoints exist on the same file system, you only need to freeze one.

  For example, if `/dev/sdb` is mounted as ext4 on `/data` and the CTE GuardPoints are `/data/protected1` and `/data/protected2`, then issuing:

  ```
  fsfreeze -f /data/protected1
  ```

  freezes /data/protected1, /data/protected2 and the underlying ext4 file system.

Caution

Do not unguard a GuardPoint, or restart the CTE Agent, while the file system is frozen. The only action permitted on a frozen file system is taking a snapshot or backing up.

- If you try to freeze `/data/protected2` after freezing `/data/protected1`, it returns as busy
- If you are not permitted to freeze one GuardPoint, then you cannot freeze any GuardPoints

LDT Restrictions

- You cannot freeze a file system while it is undergoing an LDT rekey operation. If it detects a rekey, the freeze returns as busy
- You cannot start an LDT rekey on a frozen file system

Offline Data Transformation Restrictions

Do **NOT** use `fsfreeze|xfs_freeze` while an offline transform policy is in effect.

Protecting Files with Client Settings

When any file is now marked as protected (|protect|) in the client settings, that file is protected from being modified or deleted, (even from a root process).

Note

The file is not guarded and it can be external to a GuardPoint.

Previously, the only files that were protected were the following:

```
/etc/passwd
/etc/group
/etc/security/passwd
/etc/ssh/sshd_config
```

```
/etc/ssh/sshrc
/opt/testfile
```

If the file marked as |protect| does not exist, then CipherTrust Transparent Encryption creates a 0-length file in its place. This provides an efficient means to identify and implement file protection. When the agent is stopped or uninstalled, these 0-length files are deleted and then re-created if the agent is restarted. Additionally, an audit record is generated when a file operation is denied.

The `|protect|` status is displayed using `secfsd -status auth.`

Restricting Access Overrides with Client Settings

CipherTrust Transparent Encryption host/client settings are the means by which an administrator configures user authorization. Users with root privileges, on Linux or AIX systems, have the unfettered ability to override all file access and execution permissions imposed by the system.

CipherTrust Transparent Encryption access control allows you to restrict privileges of users, groups, application processes and binaries, including root users and setuid programs. By default, CipherTrust Transparent Encryption agent **DOES NOT** trust any process as authenticated. Any attempt to access a resource, by any process, will therefore be flagged with a “User Not Authenticated” notification. The CipherTrust Transparent Encryption agent must be instructed to trust the authenticator process progeny. For example, `/usr/sbin/sshd` is a process that can be trusted to authenticate the user to the system and to CipherTrust Transparent Encryption.

In some setups, when editing a host, system administrators can use the **host settings** > |`authenticator`| feature with `su` to change identities and gain access to restricted data. You can instruct CipherTrust Transparent Encryption to not trust any authentication attempt performed by certain identities by assigning restricted users to a user shell that CipherTrust Transparent Encryption can block from authenticating other processes.

Any executable path that is marked with a |`path_no_trust`| host setting marks the process, and all child processes, as not trusted. Non-trusted processes are treated as "User Not Authenticated" to prevent access on user-based policies.

CipherTrust Transparent Encryption prevents overrides from other host settings authenticators, using the |`path_no_trust`| status. If a user runs the `su` command from a

non-trusted shell, that new shell is still marked as |`path_no_trust`|, even if |`authenticator|/usr/bin/su` is specified in the host-settings. The |`path_no_trust`| feature overrides any and all authenticators under host settings.

> **Note**
>
> Using `|trust|*` before a `|path_no_trust|` host setting no longer disables the `|path_no_trust|` host setting.

For example, the following host setting denies authentication for users accessing through sshd:

```
|trust|*
|path_no_trust|/usr/sbin/sshd
```

To restrict access overrides:

1. In the CipherTrust Manager products page, click **Transparent Encryption > Clients**.
2. Click on an existing Client name to edit the host.
3. Click **Client Settings** tab.
4. Add the following to the settings:

   ```
   |path_no_trust|<path of the binary>
   ```

   Example:

   ```
   |path_no_trust|/bin/ksh
   ```

   The above example indicates that no process under the kshell executable will be authenticated.
5. Click **Apply**.

register_host Utility

Use the `register_host` utility to create certificate requests, exchange certificates between the CipherTrust Manager and the host, and to register CTE on the CipherTrust

Manager. After the host is registered, you can configure CTE, apply GuardPoints, or perform database backups. Run the `register_host` utility in text mode on a terminal window.

Caution

The default host registration timeout is 10 minutes. If the host is unable to reach the CipherTrust Manager within the allotted period because of an extremely slow network connection, set the `REGISTER_HOST_TIMEOUT` environment variable to extend the registration timeout. The variable value is an integer expressed in seconds. You might also have to extend the default TCP timeout.

secfsd Utility

The `secfsd` utility displays the following attributes of CTE:

- GuardPoints defined in the **GuardPoints*
- Authentication parameters defined in the *Host Settings* tab
- Lock status set by enabling **FS Agent Locked** and **System Locked**
- Web destination and SSL certificate for uploading log entries
- Policies applied in the **GuardPoints** tab
- Status of required processes (`secfsd` and `vmd`)
- Version of `secfs`

The `secfsd` utility is also used to mount GuardPoints for `Directory (Manual Guard)`. Normally, CTE automatically mounts the `secfs` file system when you apply a GuardPoint to a directory. On Linux, the `secfsd` utility is located in *`<install_dir>/secfs/.sec/bin`* and a symbolic link to this file is placed in `/usr/bin/secfsd`.

secfsd syntax

Command	Description
`-help`	display `secfsd` options

Status Options

Command	Description
`-status guard [-v | -tree]`	list all GuardPoints

Command	Description
`-status keys`	show current encryption key state
`-status auth`	list authentication settings
`-status lockstat`	show CTE lock status
`-status logger`	list logging details
`-status policy`	list configured policies
`-status pslist`	list protected processes
`-status devmap`	list guarded devices

Manual GuardPoint options

Command	Description
`-guard path [container ID]`	manually guard path
`-unguard path [container ID]`	manually unguard path

Version option

Command	Description
`-version`	list version of kernel module `secfs2`

** Encryption Mode option information**

Command	Description
`crypto`	Displays the encryption modes that are supported.

** Configuration Mode option information**

Command	Description
`config <config_param> <value>`	Displays the encryption modes that are supported.

secfsd Examples

Display GuardPoint Information

To display the GuardPoint paths, applied policies, policy type, and guard status, use the `secfsd -status guard` command. For example:

```
secfsd -status guard
```

```
GuardPoint    Policy           Type             ConfigState
Status        Reason
----------    ----------       -----            -----------
------        -----
/opt/apl/lib  allow AllOps_fs  local            guarded
guarded       N/A
/dev/sdb      watchaccess_rd   rawdevice        guarded
guarded       N/A
/dev/sdc      watchaccess_rd   manualrawdevice  guarded
guarded       N/A
/dev/sdd      watchaccess_rd   manualrawdevice  unguarded     not
guarded   Inactive
/opt/apl/tmp MSSQL00123        manual           unguarded     not
guarded   Inactive
```

Column	Description
GuardPoint	Full path of the GuardPoint.
Policy	Name of the policy applied to the GuardPoint.
Type	Can be local, automount, manual, raw device, or manual raw device. Configured in the **GuardPoints** tab.
ConfigState	Guard status of the GuardPoint, as recognized by the key manager. It can be guarded or unguarded.
Status	Current guard status, as recognized by CTE. State can vary.
Reason	Additional information about the status, if any.

Note

• Config State and Status can vary. As an example, if you apply a GuardPoint and someone is currently working in the GuardPoint, the policy cannot be applied at that time. In this case, the ConfigState is guarded and the Status is not guarded.
• When the user removes an auto-mounted GuardPoint from CipherTrust Manager, the CTE Agent is only deleted after the configured `autofs` timeout expires. This timeout does not start until the GuardPoint is free.
The timeout can be changed in the `auto.master` file on the host.

Display GuardPoint Information in a Different Format

To display the same information in a block format, use the `secfsd -status guard -v` command. For example:

```
secfsd -status guard -v

GuardPoint: 1
        Policy:          allowAllOps_fs
        Directory:       /opt/apps/apps1/tmp
        Type:            local
        ConfigState:     guarded
        Status:          guarded
        Reason:          N/A
GuardPoint: 2
        Policy:          allowAllRootUsers_fs
        Directory:       /opt/apps/apps1/lib
        Type:            local
        ConfigState:     guarded
        Status:          guarded
        Reason:          N/A
```

Display Host Settings

To display the SHA2 hash signature for each protected host setting, use the `secfsd -status auth` command. For example:

```
secfsd -status auth

|authenticator|/bin/su
3E765375897E04C39AB17D4C755F50A35195535B6747DBA28DF9BD4AA672DFF9
|authenticator|/usr/sbin/sshd
98FC599D459EDEA52A60AB394B394803B5DAB96B53148DC608732DDA6777FA1A
|authenticator|/usr/sbin/in.rlogind
```

```
5C9A0EDD8BF54AE513F039476D21B3032507CF957AA0CB28C368EB8AB6E684FB
|authenticator|/bin/login
0D2EE0B995A30AE382B4B1CA5104715FC8902F457D283BDABAAD857B09259956
|authenticator|/usr/bin/gdm-binary
363780522E3CCF9ABF559F059E437743F9F97BBBB0EE85769007A464AD696BD1
|authenticator|/usr/bin/kdm
BAD41BBCDD2787C7A33B5144F12ACF7ABC8AAA15DA9FDC09ECF9353BFCE614B5
```

Display Key Status

To display the status of CTE keys, use the `secfsd -status keys` command. For example:

```
secfsd -status keys
Encryption keys are available
```

Display Lock Status

To display the status of CTE locks, use the `secfsd -status lockstat` command. For example:

```
secfsd -status lockstat
FS Agent Lock: false
System Lock: false
```

The value is **true** if the lock is applied. The value is **false** if the lock is not applied. **System Lock** corresponds to **System Locked** in the *Host* window. **FS Agent Lock** corresponds to **FS Agent Locked** in the *Host* window.

Note

Before you upgrade, remove CTE software, or change operating system files, the status of FS Agent Lock and System Lock must be false.

Agent Security Configuration Protection

The Agent lock directory, `/opt/vormetric/DataSecurityExpert/agent/secfs/.sec` contains secfs secret files, configuration files, host setting signatures, etc. Thales recommends protecting the directory whenever secfs is online.

Applying improved directory protection ensures that only CTE applications (`vmd, secfsd, voradmin`, etc.) can modify the .sec directory and the files in it. All users, including root, are denied read/write access to the files. They also do not have permissions to modify `conf` and `bin` directories, using other tools.

A new command has been created to protect the directory: `voradmin secfs config`

Syntax

```
voradmin secfs config <configuration_parameter> <value>
```

Example

```
voradmin secfs config pagecache_writeback 1
```

Previously, you would have had to use the following command to achieve the same results as the example above:

```
echo 1 > /opt/vormetric/DataSecurityExpert/agent/secfs/.sec/conf/
pagecache_writeback
```

Note

When CTE is upgraded to v7.2.0 from the previous release, it may display 'Permission Denied' warnings which display when files are removed from subdirectories of the `.sec` directory. You can ignore these warnings. They are harmless.

Display CTE Log Status

To display the status of CTE log service, use the `secfsd -status logger` command. For example:

```
secfsd -status logger
```

Upload URL: https://vmSSA06:8444/upload/logupload,https://vmSSA07:8444/upload/logupload, \ https://vmSSA05:8444/upload/logupload Logger Certificate directory: /opt/vormetric/DataSecurityExpert/agent/vmd/pem

This command sequence returns the URL to which the log service sends log data. It also returns the directory that contains the CTE certificate. CTE uses the certificate to authenticate CTE when it uploads the log data to the CipherTrust Manager.

Display Applied Policies

To display the policies that are applied to CTE, use the `secfsd -status` policy command. For example:

```
secfsd -status policy

Policy: enc-audit
Type: ONLINE
```

Display CTE Process Information

To display CTE processes, use the `secfsd -status pslist` command. This command shows the process number associated with each CTE process. To show the details about a specific CTE process, use the `ps -fp <process #>` command, where `<process #>` is the process number from the `secfsd -status pslist` command.

For example:

```
secfsd -status pslist
Protected pid list:     739   731

ps -fp 739
UID     PID   PPID  C   STIME      TTY  TIME   CMD
root    739   1     0   11:04:56    -   0:00 /opt/vormetric/   \
    DataSecurityExpert/agent/vmd/bin/vmd
```

Display CTE Version Information

To display CTE version information, use the `secfsd -version` command. For example:

```
secfsd -version
version: <Release.build-number>
```

Display CTE Crypto Information

To display CTE support information for encryption modes, use the `voradmin secfs crypto` command. For example:

```
voradmin secfs crypto
AES CBC, CBC_CS1, XTS modes are supported
Encryption key protection is supported
```

Manually Enable a GuardPoint in CipherTrust Manager

To manually enable a GuardPoint on an AIX host:

1. Click **CTE > Clients> <*clientName*> GuardPoints**
2. Click **Create GuardPoint**.
3. In the Policy field, select a policy.
4. Set Type to **Manual Directory**.
5. Click **Browse** and enter the GuardPoint path.
6. Click **Create**.
7. Log onto the system hosting CTE as the root user.
8. To manually enable the GuardPoint, use the `secfsd -guard <path>` command. For example:

```
secfsd -guard /opt/apps/etcsecfsd: Path is Guarded
```

9. To verify the change, use the `secfsd -status guard` command. For example:

```
secfsd -status guard

GuardPoint      Policy          Type    ConfigState  Status
Reason
----------      ------          ----    -----------  ------
------
/opt/apps/etc   allowAllOps_fs  manual  guarded      guarded
N/A
```

Secfsd Raw and Block Devices

CTE only creates block devices. To display them, use the ls `-l /dev/secvm/dev` command. For example:

```
ls -l /dev/secvm/dev

brw-------    1 root     system       38,  1 Jan 29 16:37 hdisk1
brw-------    1 root     system       38,  2 Jan 29 16:37 hdisk2
crw-------    1 root     system       38,  3 Jan 29 16:37 rhdisk1
crw-------    1 root     system       38,  4 Jan 29 16:37 rhdisk2
```

Note

Access controls on SECVM devices are supported with **only** Direct IO. CipherTrust Transparent Encryption does not support access controls on SECVM devices when using buffered I/O. Buffered I/O on SECVM devices are only supported with a wide-open (applyKey/permit) policy.

Using Advanced Encryption Set New Instructions (AES-NI)

To verify AES-NI hardware support, type:

```
vmsec check_hwenc
```

Unlike the -c algo command above, CTE does not have to be running for this command to execute. This command displays one of the following messages to stdout:

```
"AES-NI hardware encryption is supported on this system."
```

```
"AES-NI hardware encryption is not supported on this system. "
```

User Cache Lookup Improvements

CTE has added a feature to improve the performance of the user cache lookup function, which contains information such as username and group name(s), plus timestamps and other supporting flags. It is mainly used during LDAP authentication. This feature improves lookup performance by allowing user-configurable values for lookup retries and user information refresh times. Performance can be impacted if authentication/ user lookups and timeout/ retries take a significant amount of time. Previously, these values were hard-coded. Now they are user-configurable.

The configuration options for this feature contain three new configuration parameters:

Initial cache miss

Initial check of access when no user information cache entry exists.

- **Default value**: 60 seconds
- **Default minimum**: 2 seconds
- **Default maximum**: 3600 seconds (1 hour)

To set this value, type:

```
# voradmin secfs config usrinf_miss_timeout <seconds>
```

Cache expiration timeout

Used when user information cache entry has not been used for the duration needed to trigger a refresh.

- **Default value**: 300 seconds (5 minutes)
- **Default minimum**: 60 seconds
- **Default maximum**: 86400 seconds (1 day)

To set this value, type:

```
# voradmin secfs config usrinf_expiry_timeout <seconds>
```

Cache stale timeout

When a cache entry has not been updated for the duration of the timeout, the entry will be considered inactive and removed.

- **Default value**: 300 seconds (5 minutes)
- **Default minimum**: 60 seconds
- **Default maximum**: 86400 seconds (1 day)

To set this value, type:

```
# voradmin secfs config usrinf_stale_timeout <seconds>
```

Usage

CTE uses the default values initially. If network errors occur and LDAP failure is observed (in system logs, look for timeout errors), then you have two options:

- If the network errors can be corrected in a short time, then the timeout values can remain unchanged.
- If not, set the expiration and stale timeout to large values. Then, reduce the initial timeout incrementally until the problem resolves.

> **Note**
>
> CTE must be restarted when a new timeout value is set, in order for the value to take effect.

Keep the new value until the problem is resolved. Then, once the network problems have been fixed, reset the timeout values back to the initial values.

vmutil

Usage

```
vmutil <options> <operations>
```

Options

Option	Function	Description
-a	--agent	Specify agent type (vmd or pkcs11)
-d	--extdir	Specify location of external certificate and key set
-e	--error	Specify one or more errors for state report
-h	--help	Display help and exit
-l	--loglevel	Specify log output level (debug, info, warn, error or fatal)
-f	--force enabled	Force option for delete host operation
-v	--version	Display program version and exit

Operations

Operation	Description
certexpiry	Report the certificate expiration date for this agent
deletehost	Delete this client from the server
renewcerts	Renew certificate set for this agent
reportstate	Report agent state and error conditions to server
unregister	Unregister this agent/client from the server
updatecerts	Update external certificate and key set for this agent/client

vmsec Utility

The vmsec utility allows you to manage security aspects of CTE on the host. On Linux hosts, the `vmsec` utility is located in:

```
/opt/vormetric/DataSecurityExpert/agent/vmd/bin/vmsec
```

vmsec Syntax

Syntax	Description
`checkinstall`	Show vmd kernel status
`challenge`	Enter the dynamic host password
`vmdconfig`	Display the vmd configuration
`check_hwenc`	Display kernel configuration
`hwok`	Report status of hardware signature
`passwd [-p <password>]`	Enter the static host password
`version`	Display CTE version

vmsec Examples

- Display CTE Challenge String
- Display CTE Status
- Entering a Password
- Display Kernel Status
- Display CTE Build Information
- Display Contents of Conf files

Display CTE Challenge String

To display a CTE password challenge string and enter the response string when the CipherTrust Manager is not network accessible, use the `vmsec challenge` command. This command displays a challenge string that you can send to your key manager administrator, who will then send you back the correct response information.

For example:

```
 vmsec challenge
Contact a Security Server administrator for a response.
Your host name is "Host120" Your challenge is: HPTQ-ZYLK
Response -> IHFY-W7WG-PDAO-QKKQ
```

Contact your key manager administrator and give them the challenge string. The administrator will give you the response string. Enter the response string in the **Response** field and press **Enter**. You have 15 minutes to enter the response string.

> **Tip**
>
> If you are using CipherTrust Manager, the ability to change the contact string will be added in a future release. For CipherTrust Manager, the contact string says "Contact your CM administrator".

Display CTE Status

This utility shows you if CTE is configured and running. If it is not running, you might need to start it manually. To display CTE status, use the `vmsec checkinstall` command. For example:

```
 vmsec checkinstall
The kernel component is installed and running.
```

Entering a Password

To enter the CTE static host password, use the `vmsec passwd` command. For example:

```
 vmsec passwd
Please enter password:
OK passwd
```

To enter CTE static host password on the command line so you can specify it in a batch script, specify the password using the -p option. For example:

```
 vmsec passwd -p myPass123
OK passwd
```

Display Kernel Status

To display the kernel status, use the vmsec status command. For example:

```
vmsec status
FILE_FORMAT=2
FILE_GENERATED=08/27/2019 18:54:10
SA_QOS_STATUS=0
SA_HOST_CPU_UTIL=0
GP_1_Policy=27
GP_1_Dir=/gp
GP_1_lock=1
GP_1_type=1
GP_1_gtype=manual
GP_1_opt=gtype=2,policy=27,lock=1,type=1,dir=/gp/
GP_1_config_state=unguarded
GP_1_status=not guarded
GP_1_statuschk_tm=0-00-00 00:00:00
GP_1_config_op_retry_cnt=0
GP_1_config_op_attempt_tm=0-00-00 00:00:00
GP_1_flags=0
GP_1_reason=Inactive
GP_1_usage=free
TOTAL_GP=1
KEYS_AVAILABLE=TRUE
sdk_version=<Release.build-number>
sdk_builddate=2019-08-19 15:16:46 (PDT)
coreguard_locked=false
system_locked=false
logger_upload_url=https://thl602-2114.qa.com:8447/upload/logupload,htt
ps://thl602-2116.qa.com:8447/upload/logupload
logger_cert_dir=/opt/vormetric/DataSecurityExpert/agent/vmd/pem
hostname_for_logging=vmd
QOS_PAUSED=false
vmd_STRONG_ENTROPY=false
vmd_URL=https://thl602-2114.qa.com:8446
vmd_SRV_URLS=https://thl602-2114.qa.com:8446, https://thl602-2116.qa.c
om:8446
vmd_PRIMARY_URL=https://thl602-2114.qa.com:8446
vmd_SUPPORTS_F8P=TRUE
vmd_SUPPORTS_CR256=TRUE
vmd_RANDHP=TRUE
```

```
learn_mode=false
concise_logging=false
vmd_listening_port=7024
vmd_initialization_time=2019-07-25 12:07:14.514
vmd_last_server_update_time=2019-07-25 12:12:04.747 policy_name_27=aes2
56
policy_version_27=0
policy_keyvers_27=0
policy_type_27=ONLINE
policies=27
logger_suppression_VMD=SUPPRESS
logger_intervaltime_VMD=600
logger_repeat_max_VMD=5
logger_suppression_POL=SUPPRESS
logger_intervaltime_POL=600
logger_repeat_max_POL=5
CONFIG_SA_1=27
TOTAL_CONFIG_SA=1
SA_1_NAME=27
SA_1_ALIAS=aes256
SA_1_TYPE=0
SA_1_REF=1
SA_1_HIP_REG_TIME=0
SA_1_FLAGS=1
TOTAL_SA=1
TOTAL_AUTH=0
AUTHBIN_1=|authenticator|/usr/sbin/sshd B92A3D7EEF67B82230F7F76097D6515
9FCF5722A4154A249EFDC22C20F1B572C
AUTHBIN_2=|authenticator|/bin/login 4F210D1B83ACD79B006BCF7DB247ED002A4
5FC892C42720390BFA6AE21AEA8DC
TOTAL_AUTHBIN=2
```

Display CTE Build Information

To see the CTE build version, use the `vmsec version` command. For example:

```
 vmsec version
Version 6
```

```
7.2.0.128
2022-03-17 15:15:23 (PDT)
Copyright (c) 2009-2022, Thales. All rights reserved.
```

Display Contents of Conf files

To display the contents of the `agent.conf` and `.agent.conf.defaults` files, use the `vmsec vmdconfig` command. For example:

```
 vmsec vmdconfig
appender_syslogdest_Syslog_Appender_0=127.0.0.1
VMSDK_AGENT_CONFIG_FILE=/opt/vormetric/DataSecurityExpert/agent/vmd/
etc/agent.conf
appender_layout_Syslog_Appender_0=Syslog_Layout
VMSDK_AGENT_VERSION=7.2.0.128
VMSDK_AGENT_BUILD_ID=28
PREV_URLS=https://srv.my.thales.com:8443
syslog_appender_myhost name=dev.my.thales.com
VMD_PORT=7024
...
...
appenders=Upload_Appender, File_Appender, Syslog_Appender_0
layouts=Upload_Layout, File_Layout, Syslog_Layout, Simple
CONNECT_TIMEOUT=180000
URL=https://srv.my.thales.com:8443
STRONG_ENTROPY=false
```

vmd utility

The `vmd` utility displays CTE software version information.

The `vmd` utility is located in `/opt/vormetric/DataSecurityExpert/agent/vmd/bin` and a symbolic link to this file is placed in `/usr/bin/vmd`.

Syntax

```
vmd [OPTIONS...]
```

`-h` show utility syntax

`-v` display CTE version

`-f` runs `vmd` in the foreground

Display the Installed Version

To display the installed CTE version, type:

```
 vmd -v
Version 6
<Release.build-number>
2022-02-04
Copyright (c) 2009-2022, Thales.. All rights reserved.
```

CipherTrust in-Place Data Transformation for Linux

This section contains the following topics:

Introduction to in-Place Data Transformation (IDT)

CTE offers in-Place Data Transformation (IDT) Capable Device GuardPoints on Linux. IDT-Capable GuardPoints allow you to guard devices by transforming the plain-text data to cipher-text on the host device. The data transformation process is called in-

Place Data Transformation (IDT). The term “IDT-Capable” refers to the data transformation capability available on IDT-Capable GuardPoints.

IDT is not the same as the legacy offline data transformation. IDT is a block level data transformation with built-in resiliency to recover from system crashes during the data transformation process. IDT uses the CTE Private Region to manage the entire transformation process (For details, see The CTE Private Region and IDT Device Header).

IDT partitions the data on a device in segments of 1 MB in size and transforms one or multiple segments, up to 60 segments, in parallel. The IDT process preserves existing data in a segment during transformation in the CTE Private Region, and then transforms the data in-place. IDT also maintains the segments undergoing transformation in the CTE Private Region. In the event of system crash, IDT will recover the segments undergoing transformation at the time of crash and then resume the transformation process.

Another advantage of IDT over legacy offline data transformation is that IDT does not require a separate policy for data transformation. Instead, IDT allows you to initialize each device as either a “new device” with no existing data or as an “existing device” with existing data that needs to be transformed. You can then apply any IDT policy to any combination of new and existing devices and IDT will immediately guard the new devices while starting the IDT transformation process on the existing devices. New devices are immediately available for use while existing devices are inaccessible until the IDT process completes and all data has been converted from plain-text to cipher-text.

Requirements for IDT-Capable GuardPoints

- IDT-Capable GuardPoints are available for Linux with CTE 6.3.1 or subsequent versions. All versions of CipherTrust Manager work with IDT-Capable GuardPoints.
- The host server must use the Advanced Encryption Standard instruction set (AES-NI).
- The policy assigned to the IDT-Capable GuardPoint must be an **in-Place Data Transformation** policy and use an XTS/CBC-CS1 AES 256 encryption key.
- In order to create an IDT-Capable GuardPoint on a raw device, the device must be either:
    - Exported from an external storage system to the host device.
    - On a locally-attached disk.

- Devices protected by an IDT-Capable GuardPoint cannot currently be initialized/added as physical volumes for use by LVM. When LVM support is added, it will be announced in the CTE Release Notes.
- Existing devices divided into one or more logical partitions cannot be guarded as IDT-Capable Device GuardPoints. Logical partitions in such devices cannot be accessed or separately guarded after guarding the device.

  For example, the logical partition `/dev/sda1` or `/dev/sda2` inside `/dev/sda` cannot be accessed after guarding /dev/sda as IDT-Capable GuardPoint. Using `/dev/secvm/dev/sda1` is invalid as `/dev/secvm/dev/sda1` is not a GuardPoint and cannot be guarded, and, as such, would not provide access to clear-text data on `/dev/sda1`. However, you can guard individual partitions, such as `/dev/sda1` or `/dev/sda2`, as IDT-Capable GuardPoints without guarding the entire `/dev/sda` device.
- IDT-Capable GuardPoints requires XTS-AES mode of the AES algorithm for encryption.
- CTE only supports IDT on servers with microprocessors integrated with Advanced Encryption Standard instruction set (AES-NI).

The CTE Private Region and IDT Device Header

IDT-Capable GuardPoints require a small amount of disk space in the standard CTE Private Region. The reserved space is where CTE stores metadata information to identify IDT-Capable GuardPoints and to perform all data transformation and rekey operations in a resilient manner to avoid data loss or integrity issues due to system failures. The IDT-specific reserved space within the CTE Private Region is known as the IDT Device Header. By default, when you initialize a device as an IDT-Capable GuardPoint, CTE reserves 63 MB of space starting at the first sector on the device for the CTE Private Region.

CTE writes the IDT Device Header into the CTE Private Region when the device is guarded for the first time. If there is existing data on the device, the data at the start of the device is relocated to the available free space on the device and CTE creates the CTE Private Region starting at the first sector. For details, see Initialize a Linux Device with Existing Data.

CTE Private Region Location

Normally, CTE requires that the CTE Private Region be embedded at the beginning of the device. IDT, however, allows you to specify that the CTE Private Region for an IDT-

Capable GuardPoint should be located in a central CTE metadata directory on the host called `/vte/vte-metadata-dir` (default: `/opt/vte/vte-metadata-dir`). If you use this option, CTE stores the CTE Private Region and IDT Device Header for the device in this directory. The location of the CTE Private Region for a device is determined when you first initialize the device as an IDT-Capable GuardPoint. For details, see Initializing an IDT-Capable Device.

Warning

Access to the CTE metadata directory is local to the CTE protected host. Devices whose access is shared across multiple CTE protected hosts in a cluster *must* be configured with the CTE Private Region embedded in those devices. Using a centralized metadata directory for shared devices will lead data corruption.

The location of the CTE Private Region does not affect CTE's functionality, but there are some considerations if you choose to use the centralized metadata directory `/vte/vte-metadata-dir`:

- Thales recommends that you keep the metadata for the device on the device if at all possible. You should only use the centralized metadata directory if the device cannot be expanded to accommodate the CTE Private Region.
- The centralized CTE metadata directory must be guarded by the Administrator to prevent accidental modification or deletion of the CTE metadata. If the CTE metadata directory is not guarded, any attempt to configure or enable an IDT-Capable GuardPoint using the centralized metadata directory will be rejected. The policy associated with the metadata directory must:
- Deny all users (including the root user) the ability to modify or remove any files in the metadata directory.
- Use the key rule `clear_key` so that the metadata is stored in clear text.
- You must back up this directory whenever you back up a device that uses the directory. You will not be able to restore a protected device without access to its corresponding metadata in `/vte/vte-metadata-dir`.
- Devices with existing data do not need to be resized to accommodate the CTE Private Region, so there are no disk size discrepancies between system utilities such as `fdisk` and any other applications. However, Thales still recommends that you do not shrink an IDT-Capable GuardPoint even if the CTE Private Region is not embedded on the device.

Device Size

If you embed the CTE Private Region on the device itself, after configuring and guarding the IDT-Capable GuardPoint on the device, the device size reported to applications is the size of the device minus the space reserved for the CTE Private Region. This can lead to a discrepancy between the disk size reported by some applications versus the size reported by system utilities such as `fdisk`.

Warning

Do not shrink IDT-Capable GuardPoints. Due to the relocation of user data from the CTE Private Region, if you shrink the device, you may corrupt data on the device.

The IDT Device Header contains both the available device size and the size of the CTE Private Region. To view the IDT Device Header, use the `voradmin idt status <device-name>` command. The **Exported Disk Size** field shows the disk size available for use by other applications. The **Private Region Size** field shows the disk size reserved for CTE. For example:

```
 voradmin idt status /dev/sdc2
IDT Header on /dev/secvm/dev/sdc2
    Version:                        1
    Change:                         0
    Private Region Size:            129024 sectors
    Exported Device Size:           9627648 sectors
    Key UUID:
9cc3c8e4-7ea7-310f-85c7-6f911de1ab52
    Mount Path:                     None
```

The `voradmin idt status` command also reports the UUID of the XTS/CBC-CS1 AES 256 key applied to the device.

IDT-Capable GuardPoint Encryption Keys

IDT-Capable GuardPoints must be encrypted using XTS/CBC-CS1 AES 256 keys. An XTS/CBC-CS1 AES 256 type key is a 512-bit key composed of two components:

- The first 256 bits of the key is the AES 256 encryption key component.
- The second 256 bits is the tweak component.
- Create XTS keys on CM in the Keys menu.
- For CM, refer to “Creating a New Key" for more information.

The Key Manager (CipherTrust Manager) generates a UUID, along with other relevant attributes, for each newly-added key. It then provides the key and its attributes to the CTE protected host when the policy containing the key is pushed to the host device. CTE stores the key and its attributes, including the key's UUID, in the IDT Device Header. The first time a device is guarded as an IDT-Capable GuardPoint, CTE writes the IDT Header on the device before data transformation takes place, if data transformation is required.

Key Attributes - Example

The following describes the parameters you should specify to add a new XTS/CBC-CS1 AES 256 key named IDT_DEMO_KEY_1. Note the algorithm and encryption mode specified for the key.

- Algorithm: **AES**
- Size: **256**
- Encryption Mode: **XTS/CBC-CS1**

Note the UUID of the key. This UUID is stored in the IDT Device Header on all devices encrypted with this key, allowing you to verify which key is being used on each device.

Policy Requirements for IDT-Capable GuardPoints

IDT-Capable GuardPoints require a policy of type **in-Place Data Transformation** with a single key rule specifying the key names for **Current Key** and **New Key**. The

Current Key name is `clear_key` or an XTS/CBC-CS1 AES 256 key name, depending on whether the data on the device has already been encrypted.

- If there is no existing data on the device or if the existing data on the device has not yet been encrypted, specify `clear_key` for **Current Key**. In the **New Key** field, specify the name of the XTS/CBC-CS1 AES 256 key that you want to use to encrypt the data on the device.
- If the existing data on the device has already been encrypted, specify the name of the key used to encrypt the data in the **Current Key** field and the name of the new XTS/CBC-CS1 AES 256 key you want to use to rekey the data in the **New Key** field. When the policy is pushed to the host, CTE will rekey the data on the device using the key specified in **New Key**. In other words, the **New Key** field specifies the XTS/CBC-CS1 AES 256 production key name to apply to the device.

In all cases, the **New Key** field specifies the XTS/CBC-CS1 AES 256 production key name that you want to use to encrypt the data on the device. After you guard an existing device with an in-Place Data Transformation policy, CTE transforms the existing data using the New Key. When the process is finished, the New Key becomes the Current Key for that device, and all data will be encrypted or decrypted with that key. The IDT Device Header contains the UUID of the key currently being used to encrypt/decrypt data on the device. To view the current key UUID, use the `voradmin idt status <device-name>` command.

You may add security rules to restrict certain user/process access to protected devices. For suggestions, see Use Cases Involving IDT-Capable GuardPoints.

A simple IDT policy requires:

- **Policy type**: in-Place Data Transformation
- **Rule**: Simple security rule that permits access to all users and programs
- **Current key**: `clear_key`
- **New key**: an IDT_KEY for encrypting the data on any device associated with the policy

Guarding an IDT-Capable Device on Linux

In order to guard an IDT-Capable device, you need to:

1. Make sure the devices you intend to guard meet the requirements for IDT-Capable GuardPoints. For details, see Requirements for IDT-Capable GuardPoints.
2. Install the CTE Agent and register the host with the Key Manager if it is not already registered. IDT does not require any special registration options or licenses.
3. Initialize the device using the `voradmin idt config [new|xform]` command to specify whether there is any existing data on this device that needs to be encrypted and to configure the location of the CTE Private Region. For details, see Initializing an IDT-Capable Device.
4. Log on to the Key Manager to apply an IDT-Capable GuardPoint to the device. For details, see Guard the Linux Device with an IDT-Capable GuardPoint.

Warning

For devices with shared access across multiple CTE Protected hosts in a cluster, you must designate one and only one of the nodes in the cluster as the node on which you plan to initialize and guard the device for the first time. The designated node must be the only one that accesses the device until the entire initial data transformation process has completed. This requires guarding each shared device at the designated host level rather than at the host group level if you are using a host group to manage the CTE Protected nodes in your cluster. DO NOT initialize or guard any device on multiple nodes in the cluster simultaneously, because multiple nodes attempting to transform the same data can corrupt the data on the entire device.

Initializing an IDT-Capable Device

When you initialize an IDT-Capable storage device, the process specifies:

- Whether there is existing data on the device that needs to be encrypted.
- Where you want to store the CTE Private Region, which contains the IDT Device Header along with metadata that identifies the IDT-Capable device as a guarded device. You can embed the CTE Private Region on the device itself or in the central CTE metadata directory on the host. (For details, see The CTE Private Region and IDT Device Header.)

How you initialize the device depends on whether it is a new device or an existing device that already has data that needs to be transformed into cipher-text. For details, see:

- Initialize a New Linux Device
- Initialize a Linux Device with Existing Data

Initialize a New Linux Device

Run the `voradmin idt config new` command to initialize a new device. The new option specifies that the device does not hold user data so no initial data transformation is required. For a shared device that is accessed from multiple protected hosts, you must initialize the device only once and on only one protected host.

> **Note**
>
> To configure devices with multiple IO paths for Linux, see Changing the Encryption Key on Linux IDT-Capable Devices.

1. Log into the device as `root`.
2. Run the `voradmin idt config [-external] new [-c <n>] <device-name>` command, where:
    - `-external` is an optional parameter that tells CTE you want to use the centralized CTE metadata directory instead of embedding the CTE Private Region on the device

itself. If you use this option, you must have configured and guarded the centralized CTE metadata directory as described in CTE Private Region Location.

- `new` (required) indicates that the device contains no data (it is a new disk). As soon as you push the IDT policy, the device will be available as a guarded IDT-Capable GuardPoint.
- `-c <n>` (optional). If you use this option, CTE sets the number of data transformation jobs to run in parallel to the number specified in `<n>`. `<n>` can be an integer between 1 and 60 (default: 8).

  Each data transformation job transforms 1MB worth of data and requires CPU resources in addition to three I/O operations as part of data transformation. Each job reads 1MB of data from the device, preserves the data in the CTE Private Region, rekeys the data to cipher-text, and writes the transformed data to the device. If you increase the number of parallel jobs, the data transformation process will complete faster but there will be an increased performance impact on the system. Only increase the `-c` option if you are certain that the system resources are available to handle the additional load.

  The value for the `-c` option you specify here remains in effect for all subsequent data transformations (such as any data rekeys) until you specify a new value.
- `<device-name>` (required). Specifies the device name. For example, `/dev/sdc2`.

For example, if you want to initialize a new Linux disk named `/dev/sdc2` using 10 parallel data transformation jobs with the CTE Private Region embedded on the device, you would specify:

```
voradmin idt config new -c 10 /dev/sdc2
```

If you want to initialize a new Linux disk named `/dev/sdc2` using the default number of parallel data transformations but with the CTE Private Region in the centralized CTE metadata directory, you would specify:

```
voradmin idt config -external new /dev/sdc2
```

3. To verify that the disk has been initialized, run the `voradmin idt status` command.

```
 voradmin idt status /dev/sdc2
Device /dev/sdc2 is configured to guard as IDT-Capable GuardPoint
```

4. At this point the Administrator can protect the device as an IDT-Capable GuardPoint through the Key Manager. For details, see Guard the Linux Device with an IDT-Capable GuardPoint.

Note

The initialization process prepares the device to be guarded but does not actually guard it. You need to assign an IDT-Capable GuardPoint to the device in the Key Manager before the device is actually protected.

Initialize a Linux Device with Existing Data

If the device has existing data, you need to use the `voradmin idt config xform` command to initialize the disk for CTE. Unless you are using the centralized CTE metadata directory, this command examines the current disk size and computes the size required to hold the existing data plus the CTE Private Region at the beginning of the device. After the CTE initialization is complete, you then need to resize the device before you can guard it with an IDT-Capable GuardPoint.

Warning

If access to the device is shared access across multiple CTE Protected hosts in a cluster, be sure to initialize the device on one and only one of the CTE hosts.

The following procedure describes how to initialize the device for CTE. Note that the existing data is not altered in any way until after you perform this procedure and you guard the data with an IDT-Capable GuardPoint. CTE does *not* begin transforming the data from clear-text to cipher-text until the IDT-Capable GuardPoint has been applied and the encryption key has been pushed to the device through the GuardPoint Policy.

1. Log into the device as `root`.
2. Run the `voradmin idt config [-external] xform [-c <n>] <device-name>` command, where:
    - `-external` is an optional parameter that tells CTE you want to use the centralized CTE metadata directory instead of embedding the CTE Private Region on the device

itself. If you use this option, you will not have to resize the device but you must have configured and guarded the centralized CTE metadata directory as described in CTE Private Region Location.

- `xform` (required) indicates that the device contains existing data. CTE will transform all existing data on the device from clear-text to cipher-text as soon as you guard the device. The device will be unaccessible until the transformation is complete, and the device must remain offline during the entire transformation process. No user access will be permitted until all data has been transformed.
- `-c <n>` (optional). If you use this option on Linux, CTE sets the number of data transformation jobs to run in parallel to the number specified in `<n>`. `<n>` can be an integer between 1 and 60 (default: 8).

  Each data transformation job transforms 1MB worth of data and requires CPU resources in addition to three I/O operations as part of data transformation. Each job reads 1MB of data from the device, preserves the data in the CTE Private Region, rekeys the data to cipher-text, and writes the transformed data to the device. If you increase the number of parallel jobs, the data transformation process will complete faster but there will be an increased performance impact on the system. Only increase the `-c` option if you are certain that the system resources are available to handle the additional load.

  The value for the `-c` option you specify here remains in effect for all subsequent data transformations (such as any data rekeys) until you specify a new value.
- `<device-name>` (required). Specifies the device name. For example, `/dev/sdc3`.

For example, if you want to initialize an existing Linux disk named `/dev/sdc3` using 10 parallel data transformation jobs with the CTE Private Region embedded on the device, you would specify:

```
voradmin idt config xform -c 10 /dev/sdc3
Device /dev/sdc3 must be resized to at least 9893888 sectors (4831
MBs) before guarding as IDT-Capable GuardPoint
```

In this case you must manually resize the Linux disk by at least 9893888 sectors before you can guard it. After you guard the disk, you can expand it again later but you cannot shrink it unless you remove the GuardPoint.
If you want to initialize the same device using the centralized CTE metadata directory, you would specify:

```
voradmin idt config xform -external -c 10 /dev/sdc3
```

Note that you do not get a message about resizing the device because the CTE Private Region will not be embedded on the device.

3. To verify that the disk has been initialized, run the `voradmin idt status` command.

```
voradmin idt status /dev/sdc3
Device /dev/sdh is configured to guard as an IDT-Capable
GuardPoint
```

4. If you are embedding the CTE Private Region on the device, at this point, you need to resize the device using your standard disk management tools before you can guard it. Make sure you increase the device size by at least the amount shown in the `voradmin idt config xform` message.
   You cannot assign an IDT-Capable GuardPoint to the device until it has been resized. If you do not resize the device, the GuardPoint assignment will fail.

5. After the device has been resized or the centralized CTE metadata directory has been configured and guarded, the Administrator can protect the device as an IDT-Capable GuardPoint through the Key Manager as described in Guard the Linux Device with an IDT-Capable GuardPoint.

> **Note**
>
> The initialization process prepares the device to be guarded but does not actually guard it. You need to assign an IDT-Capable GuardPoint to the device in the Key Manager before the device is actually protected. In addition, the initialization process is only kept in memory until the device is guarded or rebooted. If the device is rebooted before you guard it, you will need to perform the initialization procedure again.

Guard the Linux Device with an IDT-Capable GuardPoint

After the device has been initialized, you can guard the device as an IDT-Capable GuardPoint from the Key Manager. For existing devices, as soon as the GuardPoint configuration has been pushed to the host and the status changes to guarded, CTE begins transforming the data on the disk using the encryption key associated with the GuardPoint Policy.

Warning

If access to the device is shared access across multiple CTE Protected hosts in a cluster, be sure to guard the device on one and only one of the CTE hosts.

Note

For details about how to create a GuardPoint in CM, see, “Managing GuardPoints", CTE Administration Guide.

To see the data transformation progress, use the `voradmin idt xform status <device-name>` command, as described in Viewing Device Status and the IDT Device Header.

After the device is initialized and guarded, the protected device must be accessed through the CTE device pathname. This pathname corresponds to the secvm device. For example, the Linux device pathname `/dev/sdc2` becomes `/dev/secvm/dev/sdc2` as soon as the process is complete.

Note

- Be sure to use the secvm device name when using file system management tools such as `mkfs` and `fsck`.
- Do not use the device mapper names corresponding to IDT-Capable GuardPoints for GuardPoint administration on protected hosts.

Data Relocation and Transformation on Existing Linux Devices

When you add an IDT-Capable GuardPoint to a device that has been initialized with the `voradmin idt xform` command and you opted to embedded the CTE Private Region on the device, CTE first relocates existing data in the region of the device designated as CTE Private Region. The data is relocated to the end of the device, into the new space allocated when you resized the device. The relocation occurs once when the device is guarded for the first time. No relocation is necessary for subsequent rekeys on the device.

Relocation of data is transparent to applications accessing data through the IDT-Capable GuardPoint. CTE will map application I/O requests over the private region to the relocated region. After guarding the device, you can grow the device size further if necessary. However, you cannot shrink the device size.

IDT does not require a separate policy for data transformation. If you initialized the device with the `xform` option, CTE starts the IDT process when transformation is required. During the IDT process, access to the device is blocked until the IDT process completes and all the data on the device has been encrypted.

```
 voradmin idt status xform /dev/sdc3
Status:         In-Process
        Relocation Zone 9764864 (relocated = 1)
        SegSpc 27, Xformation Range: 3217 ... 4799, SegIDs: 4795 4796
4791 4792 4797 4798 4799
        KeyID:          2793    Key Name:     IDT_DEMO_KEY_1
        Old KeyID:      0       Old Key Name: clear_key

# dd if=/dev/secvm/dev/sdc3 of=/dev/null bs=512 count=1
dd: failed to open 'dev/secvm/dev/sdc3': Resource temporarily
unavailable

# voradmin idt status xform /dev/sdc3
Status:         Complete
        Relocation Zone 9764864 (relocated = 1)
        SegSpc 27, Xformation Range: 3217 ... 20189, SegIDs: none
        KeyID:          2793    Key Name:     IDT_DEMO_KEY_1
        Old KeyID:      0       Old Key Name: clear_key

# dd if=/dev/secvm/dev/sdc3 of=/dev/null bs=512 count=1
1+0 records in
1+0 records out
512 bytes (512 B) copied, 0.000989039 s, 518 kB/s
```

Thin-Provisioned Devices

IDT skips transforming thin-provisioned regions of a device. Data returned to IDT as sequence of clear-text zeros, in sector size granularity, is indication of possible sparse or un-allocated regions of the device that do not have to be transformed.

IDT Recovery From Crash

IDT is fault tolerant in the event of system crashes. IDT keeps track of the transformation process over the entire device. In the event of a crash, IDT will automatically resume transformation from the point of failure as soon the GuardPoint is enabled after system startup.

If you find the transformation status set to **In-Progress** when the GuardPoint is not enabled, the **In-Progress** state reflects an earlier system crash after which the GuardPoint has not been enabled to recover from the interruption in the IDT process.

Example of Creating an IDT-Capable GuardPoint on an Existing Linux Device

The following example shows the process of initializing an existing Linux device using `voradmin idt config xform` and guarding it as an IDT-Capable GuardPoint from the viewpoint of the Linux root user. In this example, all files in `/bin/*` are copied to a temporary location outside the device, then compared with the corresponding files on the device after the device has been resized and encrypted. The comparison proves that the file system is unchanged after the encryption process has completed.

First, we verify that the device is not protected, then we check the current size of the disk and create the copy of the files in `/bin/*`. After that, we run the `voradmin idt config xform` command to initialize the device.

```
 voradmin idt status /dev/sdc1
Device /dev/sdc1 is not configured as IDT-Capable
# fdisk -l /dev/sdc1
Disk /dev/sdc1: 21.1 GiB, 21103640576 bytes, 41218048 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 4194304 bytes
# mkfs.xfs /dev/sdc1
meta-data=/dev/sdc1              isize=256    agcount=4, agsize=1288064
blks
         =                       sectsz=512   attr=2, projid32bit=1
         =                       crc=0        finobt=0, sparse=0
```

```
data     =                       bsize=4096   blocks=5152256,
imaxpct=25
         =                       sunit=0      swidth=0 blks
naming   =version 2              bsize=4096   ascii-ci=0 ftype=1
log      =internal log           bsize=4096   blocks=2560, version=2
         =                       sectsz=512   sunit=0 blks, lazy-
count=1
realtime =none                   extsz=4096   blocks=0, rtextents=0
# mount -t xfs /dev/sdc1 /xfs
# cp /bin/* xfs
# voradmin idt config xform /dev/sdc1
Device /dev/sdc1 must be resized to at least 41347072 sectors (40378
MBs) before guarding as IDT-Capable GuardPoint
```

At this point, you need to resize the device using your device management tools. You must increase the size by at least 41347072 sectors (40378 MBs). After the device has been resized, you can verify the new size:

```
 fdisk -l /dev/sdc1
Disk /dev/sdc1: 21.2 GiB, 21169700864 bytes, 41347072 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 4194304 bytes
```

After the device has been resized, the Administrator can guard the device with the desired in-Place Data Transformation policy. If the Administrator chooses Auto Guard, data transformation begins as soon as the policy is pushed to the host. If the Administrator chooses Manual Guard, data transformation does not begin until the Linux root user initiates it with the `secfsd -guard` command. Once data transformation begins, the Linux root user can check the progress using the `voradmin idt status xform` command.

```
 secfsd -guard /dev/sdc1
secfsd: Path is guarded
# voradmin idt status xform /dev/sdc1
        Status:         In-Process
        Relocation Zone 9764864 (relocated = 1)
        SegSpc 27, Xformation Range: 3217 ... 4799, SegIDs: 4795 4796
4791 4792 4797 4798 4799
```

```
        KeyID:          2793    Key Name:     IDT_DEMO_KEY_1
        Old KeyID:      0       Old Key Name: clear_key
```

After the status has changed to completed, you can compare the current version of the files in `/bin/*` with the ones you copied earlier.

```
 voradmin idt status xform /dev/sdc1
        Status:         Complete
        Relocation Zone 9764864 (relocated = 1)
        SegSpc 27, Xformation Range: 3217 ... 20189, SegIDs: none
        KeyID:          2793    Key Name:     IDT_DEMO_KEY_1
        Old KeyID:      0       Old Key Name: clear_key
# voradmin idt status /dev/sdc1
IDT Header on /dev/secvm/dev/sdc1
        Version:                        1
        Change:                         0
        Private Region Size:            129024 sectors
        Exported Device Size:           41218048 sectors
        Key UUID:                       9cc3c8e4-7ea7-310f-85c7-6f911
de1ab52
        Mount Path:                     None
# mount -t xfs /dev/secvm/dev/sdc1 /xfs
# for file in '/bin/ls /sfx'; do cmp /bin/$file /xfs/$file; done
# unmount /xfs
```

Changing the Encryption Key on Linux IDT-Capable Devices

To meet various compliance requirements, you may want to change the key that CTE has used to encrypt IDT-Capable GuardPoints. Thales refers to this changing of encryption keys as “Key rotation” or “Rekey”. Unlike the Live Data Transformation product offered by Thales for file systems on traditional storage devices, to change the encryption key on an IDT-Capable GuardPoint, the device must be taken offline. The data on the device will be inaccessible during the key rotation process.

For CipherTrust Manager, see Key Version Modify Request for more information.

Warning

For devices with shared access across multiple CTE Protected hosts in a cluster, you must designate one and only one of the nodes in the cluster as the node on which you plan to initialize and guard the device for rekey. The designated node must be the only one that accesses the device until the entire rekey process has completed. This requires guarding each shared device at the designated host level rather than at the host group level if you are using a host group to manage the CTE Protected nodes in your cluster. DO NOT initialize or guard any device on multiple nodes in the cluster simultaneously, because multiple nodes attempting to transform the same data can corrupt the data on the entire device.

1. Log on to the Key Manager.
2. Make sure that you know what Policy you want to associate with the GuardPoint or create a new **in-Place Data Transformation** policy if needed. The key rule must specify the current XTS/CBC-CS1 AES 256 key that the GuardPoint is currently using as well as the new XTS/CBC-CS1 AES 256 key that you want to use to transform the protected data.
   You can either create a new in-Place Data Transformation policy or you can change the keys assigned to an existing in-Place Data Transformation policy. If you use an existing policy however, the new key you specify cannot have been previously used to encrypt the IDT-Capable GuardPoint. If you want to rekey the GuardPoint using a previously-used key, you must create a new policy in order to do so.
   Include both the current key and the new key in the policy to ensure that both keys will be available during the rekey process, even if the key information stored in the CTE Private Region becomes unavailable.
   Do not push this policy to the host yet.
3. Shut down any applications accessing the GuardPoint you are planning to rekey. If the GuardPoint is mounted as a file system, you must also unmount the file system.
4. Once the data can no longer being accessed, you can unguard the GuardPoint on the Key Manager.
   a. If the GuardPoint is a manual device GuardPoint, you must first unguard it using the `secfsd -unguard` command on the CTE host before you unguard it

on the Key Manager. If it is an automatic GuardPoint, you can skip this step and simply unguard the GuardPoint in the Key Manager.

The following example checks the guard status of `/dev/sdc1` and gets the current key name, then unguards the device.

```
secfsd -status guard
GuardPoint  Policy             Type             ConfigState
Status   Reason
----------  ------             ----             -----------
------   ------
/dev/sdc2   IDT_DEMO_POLICY_1  rawdevice        guarded
guarded  N/A
/dev/sdc3   IDT_DEMO_POLICY_1  rawdevice        guarded
guarded  N/A
/dev/sdc1   IDT_DEMO_POLICY_1  manualrawdevice  guarded
guarded  N/A
# voradmin idt status xform /dev/sdc1
        Status:         Complete
        Relocation Zone 0 (relocated = 0)
        SegSpc 27, Xformation Range: 4294967295 ...
4294967295, SegIDs: none
        KeyID:          2793    Key Name:     IDT_DEMO_KEY_1
        Old KeyID:      0       Old Key Name: clear_key
# secfsd -unguard /dev/sdc1
secfsd: Path is not guarded
# secfsd -status guard
GuardPoint  Policy             Type             ConfigState
Status       Reason
----------  ------             ----             -----------
------       ------
/dev/sdc2   IDT_DEMO_POLICY_1  rawdevice        guarded
guarded      N/A
/dev/sdc3   IDT_DEMO_POLICY_1  rawdevice        guarded
guarded      N/A
/dev/sdc1   IDT_DEMO_POLICY_1  manualrawdevice  unguarded
not guarded  Inactive
```

b. Return to the Key Manager, select the GuardPoint in the GuardPoints table, and click **Unguard** to unguard the device in the Key Manager.
Wait until the GuardPoint has been removed from the Key Manager.

5. Return to the device and run the `voradmin idt rekey` command. After you run the `voradmin` command, the IDT Device Header is temporarily removed from the device.

```
voradmin idt rekey /dev/sdc1
Enter YES to prepare device /dev/sdc3 for rekey -> YES
# voradmin idt status /dev/sdc1
Device /dev/sdc1 is configured to guard as IDT-Capable GuardPoint
```

> **Note**
>
> For manual GuardPoints, you must unguard the device both using `secfsd -unguard` and the Key Manager before you can use the `voradmin idt rekey` command.

6. In the Key Manager, guard the device with the new **in-Place Data Transformation** policy you created earlier. If you selected a Manual GuardPoint, use `secfsd -guard` to activate the new policy and start the data transformation to the new key.
During the IDT process, user access to the GuardPoint is blocked until IDT completes the transformation process.
The following example shows how to use `secfsd -guard` on manual GuardPoint `/dev/sdc1`, and the status messages that occur during the rekey process. If you are using an automatic GuardPoint, you do not need to use the `secfsd -guard` command. Instead, the rekey process starts as soon as you push the new policy from the Key Manager.

```
secfsd -guard /dev/sdc1
secfsd: Path is guarded
# secfsd -status guard
GuardPoint  Policy             Type             ConfigState
Status   Reason
----------  ------             ----             -----------
------   ------
```

```
/dev/sdc2   IDT_DEMO_POLICY_1  rawdevice        guarded
guarded  N/A
/dev/sdc3   IDT_DEMO_POLICY_1  rawdevice        guarded
guarded  N/A
/dev/sdc1   IDT_DEMO_POLICY_2  manualrawdevice  guarded
guarded  Data transformation in progress
# voradmin idt status xform /dev/sdc1
        Status:                 In-Progress
        Relocation Zone 0 (relocated = 0)
        SegSpc 27, Xformation Range: 3987 ... 3993, SegIDs: 3991
3987 3988 3992 3989 3990 3993
        KeyID:          2921    Key Name:     IDT_DEMO_KEY_2
        Old KeyID:      2793    Old Key Name: IDT_DEMO_KEY_1
```

7. After the `xform` status shows as completed, you can restart all application workloads on the guarded device.

```
voradmin idt status xform /dev/sdc1
        Status:         Complete
        Relocation Zone 0 (relocated = 0)
        SegSpc 27, Xformation Range: 4768 ... 4768, SegIDs: none
        KeyID:          2921    Key Name:     IDT_DEMO_KEY_2
        Old KeyID:      2793    Old Key Name: IDT_DEMO_KEY_1
# secfsd -status guard
GuardPoint  Policy             Type             ConfigState
Status   Reason
----------  ------             ----             -----------
------   ------
/dev/sdc2   IDT_DEMO_POLICY_1  rawdevice        guarded
guarded  N/A
/dev/sdc3   IDT_DEMO_POLICY_1  rawdevice        guarded
guarded  N/A
/dev/sdc1   IDT_DEMO_POLICY_2  manualrawdevice  guarded
guarded  N/A
```

Guarding an IDT-Capable Device with Multiple IO Paths on Linux

Each individual IO path from a server node to a storage controller is treated as a separate device on the host. DM-Multipath on a Linux host provides a management framework to group the individual IO paths to the same LUN into a single multipath device. If you use DM-Multipath to manage devices on the protected host, the individual devices that correspond to each IO path to the LUN cannot be configured for guarding as IDT-Capable, as those devices are under control of DM-Multipath. To guard such devices, you must guard the device mapper generated by DM-Multipath (multipathd) under the `/dev/mapper` directory.

> **Note**
>
> IDT is the only feature of CTE that exclusively supports guarding of a device mapper generated device under DM-Multipath framework.

The following example illustrates the procedure for guarding a device mapper generated device with the alias name `/dev/mapper/mpathA`.

1. Create an **in-Place Data Transformation** policy using an XTS/CBC-CS1 AES 256 key as the key rule.
2. On the host, prepare the device to be configured as IDT-Capable using the `voradmin idt config [-external] new|xform [-c n] <mapper-alias-name>` command. For example, if the disk is a new disk with no existing data, you would enter:

   ```
   voradmin idt config new /dev/mapper/mpathA
   ```

   If the disk has existing data that you want to encrypt, you would enter:

   ```
   voradmin idt config xform /dev/mapper/mpathA
   ```

3. Guard `/dev/mapper/mpathA` as Device GuardPoint using the policy created above. Be sure to check the **in-Place Data Transformation** check box.
4. For Manual Guard configuration, enable the GuardPoint using the `secfsd` command as follows:

```
secfsd -guard /dev/mapper/mpathA
```

5. For Auto Guard, wait for the `/dev/mapper/mpathA` device to be guarded on the protected host.
6. Once the device is guarded, provide the pathname of the secvm device to applications and/or file system operations. For example, `/dev/secvm/dev/mapper/mpathA`.

Viewing Device Status and the IDT Device Header

After you guard a device, you can view the status of that device using the `voradmin idt [xform] status <device-name>` command, where:

- `xform` (optional). If you specify this option, CTE shows the status of any data transformation processes happening on the device. If you do not specify this option, CTE displays the IDT Device Header for the device.
- `<device-name>` (required). The standard Linux name of the device whose status you want to view. (For example, `/dev/sdc2`.)

For example, if you want to view the IDT Device Header for the Linux device `/dev/sdc2`, you would enter:

```
voradmin idt status /dev/sdc2
IDT Header on /dev/secvm/dev/sdc2
        Version:                        1
        Change:                         0
Private Region Size:            129024 sectors
Exported Device Size:           9627648 sectors
        Key UUID:
9cc3c8e4-7ea7-310f-85c7-6f911de1ab52
        Mount Path:                     None
```

If you want to view the data transformation status on `/dev/sdc2`, you would enter:

```
voradmin idt status xform /dev/sdc3
        Status:         In-Process
```

```
        Relocation Zone 9764864 (relocated = 1)
        SegSpc 27, Xformation Range: 3217 ... 4799, SegIDs: 4795 4796
4791 4792 4797 4798 4799
        KeyID:          2793    Key Name:     IDT_DEMO_KEY_1
        Old KeyID:      0       Old Key Name: clear_key
```

The **Status** field displays **In-Progress** if a data transformation process is running, and **Completed** if the process has finished.

Linux System and IDT-Capable GuardPoint Administration

Note

For details about how to create a GuardPoint in CM, see the chapter, “Managing GuardPoints", CTE Administration Guide.

Voradmin IDT Commands on Linux

The `voradmin` command is a command line utility for management of CTE specific configuration and status reporting. The `voradmin` command also supports configuration management related IDT-Capable GuardPoints (IDT).

For details about the Linux `voradmin idt` command options, see the man page for the `voradmin` command.

File System Mount Points on Linux

You can create and mount a file system on an IDT-Capable GuardPoint. CTE imposes one restriction on the mount point pathname selected for a device. Once you mount the device on a pathname, you cannot change the mount point to a different pathname. This restriction is enforced to allow the file system mount point to be guarded using a separate policy to enforce access control rules on the mounted file system namespace.

The following example shows the mount point of the IDT-Capable GuardPoint as the `/xfs` directory. The example also shows a failed attempt to mount the file system on a different directory pathname.

```
 voradmin idt status /dev/sdc1
IDT Header on /dev/secvm/dev/sdc1
        Version:                          1
        Change:                           0
Private Region Size:             129024 sectors
Exported Device Size:            9627648 sectors
        Key UUID:
9cc3c8e4-7ea7-310f-85c7-6f911de1ab52
Mount Path:                      /xfs
# unmount /xfs
# makedir /other-xfs
# mount -t xfs /dev/secvm/dev/sdc2 /other-xfs
mount: permission denied
# mount -t xfs /dev/secvm/dev/sdc2 /xfs
```

Auto Mount Options for File System Devices on Linux

IDT-Capable GuardPoints containing file systems can also be added to the `/etc/fstab` configuration file for auto mount at startup or unmount at shutdown. An entry can be for a GuardPoint configured for either Auto Guard or Manual Guard. For more information about Auto and Manual Guard options, see Guard the Linux Device with an IDT-Capable GuardPoint.

Use the device path corresponding to an IDT-Capable GuardPoint device when specifying `fstab` entries, such as `/dev/secvm/dev/sdh`. Do not use the native device pathnames, such as `/dev/sdh`, or device mapper device names. You must also include several settings in the `fstab` entry for each IDT-Capable GuardPoint, as shown in the following table:

Option	Description
`x-systemd.requires= secvm-barrier.service`	Ensure that the IDT-Capable GuardPoint is enabled before the device is mounted at startup and disabled after the device is unmounted at shutdown. The `secvm-barrier.service` service is a proxy for all the services that make up CTE.
`nofail`	The system boot will proceed without waiting for the IDT-Capable device if it can't be mounted successfully.

Option	Description
`x-systemd.wanted-by=<idt device>.device`	Required for Linux distributions running `systemd` 242 or later. Instructs `systemd` to add a `Wants=` dependency on the IDT-Capable device to ensure that, when the device becomes available, this mount operation is executed. `<idt device>.device` is the name of the device specified in `fstab` with the '/' replaced with '-'. For example, `/dev/secvm/dev/sdb` becomes `dev-secvm-dev-sdb.device`.

This is an example of an entry in `/etc/fstab` for an IDT-Capable GuardPoint with an xfs file system that is mounted on /xfs:

```
/dev/secvm/dev/sdh /xfs xfs x-systemd.requires=secvm-barrier.service,
\
x-systemd.wanted-by=dev-secvm-dev-sdh.device,nofail 0 0
```

For information about configuring systemd for CTE, see [CTE and systemd].

Linux System Utilities for Signing

The following table includes recommendations on the system and file system specific utilities for inclusion in the signature set to allow or deny root execution.

EXT Utilities	Deny/Allow	XFS	Deny/Allow	Generic Utilities	Deny/Allow
`badblock`	Allow	`fsck.xfs`	Allow	`mount`	Allow
`debugfs`	Deny	`mkfs.xfs`	Allow	`umount`	Allow
`e2freefrag`	Allow	`xfs_repair`	Allow	`dmsetup`	Allow
`e2fsck`	Allow	`xfs_admin`	Allow		
`e2image`	Allow	`xfs_bmap`	Allow		
`e2label`	Allow	`xfs_check`	Allow		
`e2undo`	Allow	`xfs_copy`	Deny		
`filefrag`	Allow	`xfs_db`	Deny		
`fsck.ext2`	Allow	`xfs_estimate`	Allow		
`fsck.ext3`	Allow	`xfs_freeze`	Allow		

EXT Utilities	Deny/Allow	XFS	Deny/Allow	Generic Utilities	Deny/Allow
`fsck.ext4`	Allow	`xfs_fsr`	Allow		
`logsave`	Allow	`xfs_growfs`	Allow		
`mke2fs`	Allow	`xfs_info`	Allow		
`mkfs.ext2`	Allow	`xfs_logprint`	Allow		
`mkfs.ext3`	Allow	`xfs_mdrestore`	Allow		
`mkfs.ext4`	Allow	`xfs_metadump`	Allow		
`resize2fs`	Allow	`xfs_mkfile`	Deny		
`tune2fs`	Allow	`xfs_ncheck`	Allow		

Resizing Guarded IDT Devices

Devices configured as IDT-Capable can be resized using the system-provided resizing utilities. If you are using a file system on the GuardPoint, you can mount the file system after resizing the device and then grow the file system to the new size using the appropriate utility such as `xfs_growfs` or `resize2fs`.

Warning

Do not shrink IDT-Capable GuardPoints. Due to relocation of user data from the CTE Private Region, if you shrink the device, you may corrupt data on the device.

1. Stop applications from accessing the IDT-Capable GuardPoint.
    - Unmount the file system if the device is mounted.
    - Disable the IDT-Capable GuardPoint if using Auto Guard or on the host with the `secfsd unguard <device-name>` command if using Manual Guard.
2. Use the native disk management tools to resize the device.
3. After resizing the device, check the size of the device with the `fdisk -l` or similar command. Note that you cannot use the `voradmin idt status` command to verify the new size of the device at this point because the size information is not updated in CTE until the IDT-Capable GuardPoint is re-enabled.

4. If the reported size does not match what you expect, you may need to rescan your storage devices using the command appropriate for the device's connection type.
5. Once the expected size is achieved, enable the IDT-Capable GuardPoint and restart your applications.

Resizing Guarded IDT Devices

Devices configured for in-Place Data Transformation can be resized using the system-provided resizing utilities. If you are using a file system on the GuardPoint, you can mount the file system after resizing the device and then grow the file system to the new size using the appropriate utility such as `xfs_growfs` or `resize2fs`.

Warning

Do not shrink IDT GuardPoints. Due to relocation of user data from CTE private region, if you shrink the device, you may corrupt data on the device.

1. Stop applications from accessing the GuardPoint.
    - Unmount the file system if the device is mounted.
    - Disable GuardPoints: auto-guard if it is enabled on the CipherTrust Manager, or manual-guard if it is enabled on the protected host.
2. Use the native disk management tools to resize the device.
3. After resizing the device, check the size of the device. For Linux, you can use the `fdisk -l` command.
4. If the reported size does not match what you expect, you may need to rescan your storage devices using the command appropriate for the device's connection type.
5. Once the expected size is achieved, enable the GuardPoint and restart your applications.

Examples

The following example show the administrative steps to grow an XFS file system mounted on an ES GuardPoint. The example shows a device of size 41,347,072 sectors resized to 43,395,072 sectors.

```
# fdisk -l /dev/sdh

Disk /dev/sdh: 21.2 GB, 21169700864 bytes, 41347072 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 4194304 bytes

# xfs_info /dev/secvm/dev/sdh
meta-data=/dev/secvm/dev/sdh     isize=512    agcount=4, agsize=1288064 blks
         =                       sectsz=512   attr=2, projid32bit=1
         =                       crc=1        finobt=0 spinodes=0
data     =                       bsize=4096   blocks=5152256, imaxpct=25
         =                       sunit=0      swidth=0 blks
naming   =version 2              bsize=4096   ascii-ci=0 ftype=1
log      =internal               bsize=4096   blocks=2560, version=2
         =                       sectsz=512   sunit=0 blks, lazy-count=1
realtime =none                   extsz=4096   blocks=0, rtextents=0
# voradmin idt status /dev/sdh
ESG Header on /dev/secvm/dev/sdh
        Version:                        1
        Change:                         0
        Notifications:                  None
        Storage Status:                 None
        Private Region Size:            129024 sectors
        Exported Device Size:           41218048 sectors
        Key UUID:                       b1644bbd-db1e-3a8f-b829-5893dd2fd0b0
        Mount Path:                     N/A (GuardPoint not enabled)
# umount /xfs
# secfsd -unguard /dev/sdh
secfsd: Path is not guarded
```

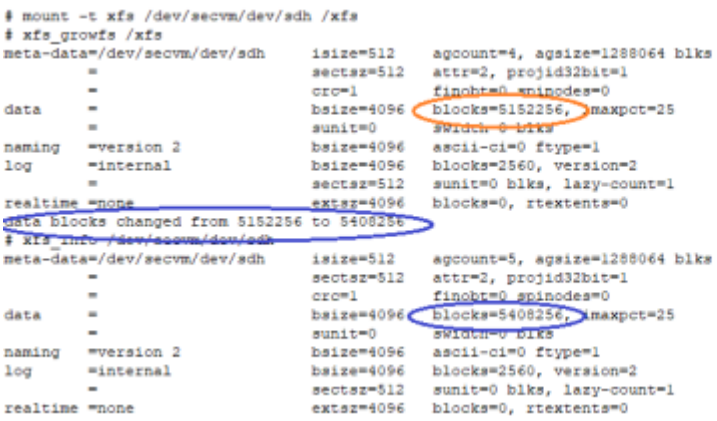
In this example we resize the device on the storage array to 43395072 sectors. Notice that the `voradmin` command reports the old device size until the device is guarded.

```
# fdisk -l /dev/sdh

Disk /dev/sdh: 22.2 GB, 22218276864 bytes, 43395072 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 4194304 bytes

# voradmin idt status /dev/sdh
ESG Header on /dev/secvm/dev/sdh
        Version:                        1
        Change:                         0
        Notifications:                  None
        Storage Status:                 None
        Private Region Size:            129024 sectors
        Exported Device Size:           41218048 sectors
        Key UUID:                       b1644bbd-db1e-3a8f-b829-5893dd2fd0b0
        Mount Path:                     N/A (GuardPoint not enabled)
# secfsd -guard /dev/sdh
secfsd: Path is guarded
# voradmin idt status /dev/sdh
ESG Header on /dev/secvm/dev/sdh
        Version:                        1
        Change:                         0
        Notifications:                  None
        Storage Status:                 None
        Private Region Size:            129024 sectors
        Exported Device Size:           43266048 sectors
        Key UUID:                       b1644bbd-db1e-3a8f-b829-5893dd2fd0b0
        Mount Path:                     /xfs
```

The final step shown in below is to mount and resize the XFS file system to include the extended space into the file system.



Use Cases involving IDT GuardPoints

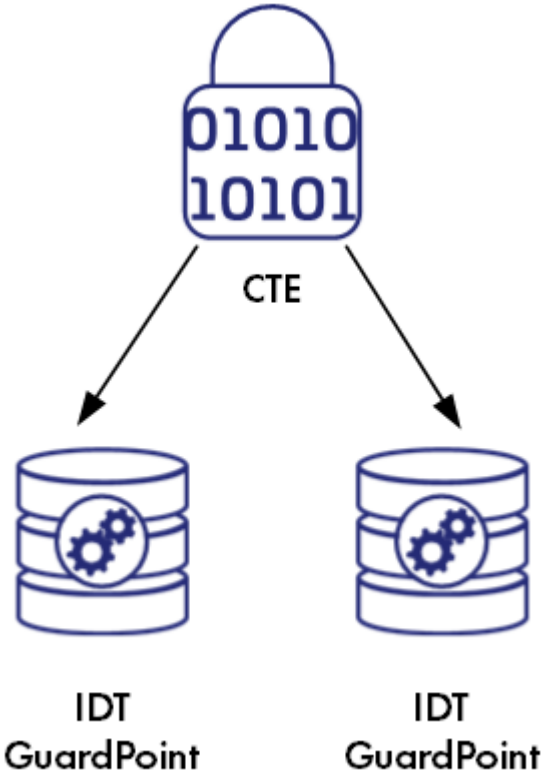
in-Place Data Transformation GuardPoints support the following use cases for managing customer data in GuardPoints. This section describes those potential use cases.

- Use Case 1: Single Encryption Key
- Use Case 2: Device-Level GuardPoints
- Use Case 3 | Part 1: Directory-Level GuardPoints
- Use Case 3 | Part 2: Challenges with Root Access on Linux
- Use Case 4: Using CTE-in-place Device GuardPoint with LVM
- Best Practices for the Migration of a legacy Raw Device GuardPoint to an in-place Device GuardPoint GuardPoint
- Migration Prerequisites
- Migration

Use Case 1: Single Encryption Key

Applications, such as an Oracle Database, store structured data in one or multiple LUNs guarded as an in-Place Data Transformation GuardPoint. In this use case, a LUN may be an independent datastore or a member of a disk group managed by an application, for example an Oracle ASM disk group. In this use case, the policy applied to the GuardPoints specifies one key rule for encryption, and, potentially, a second key

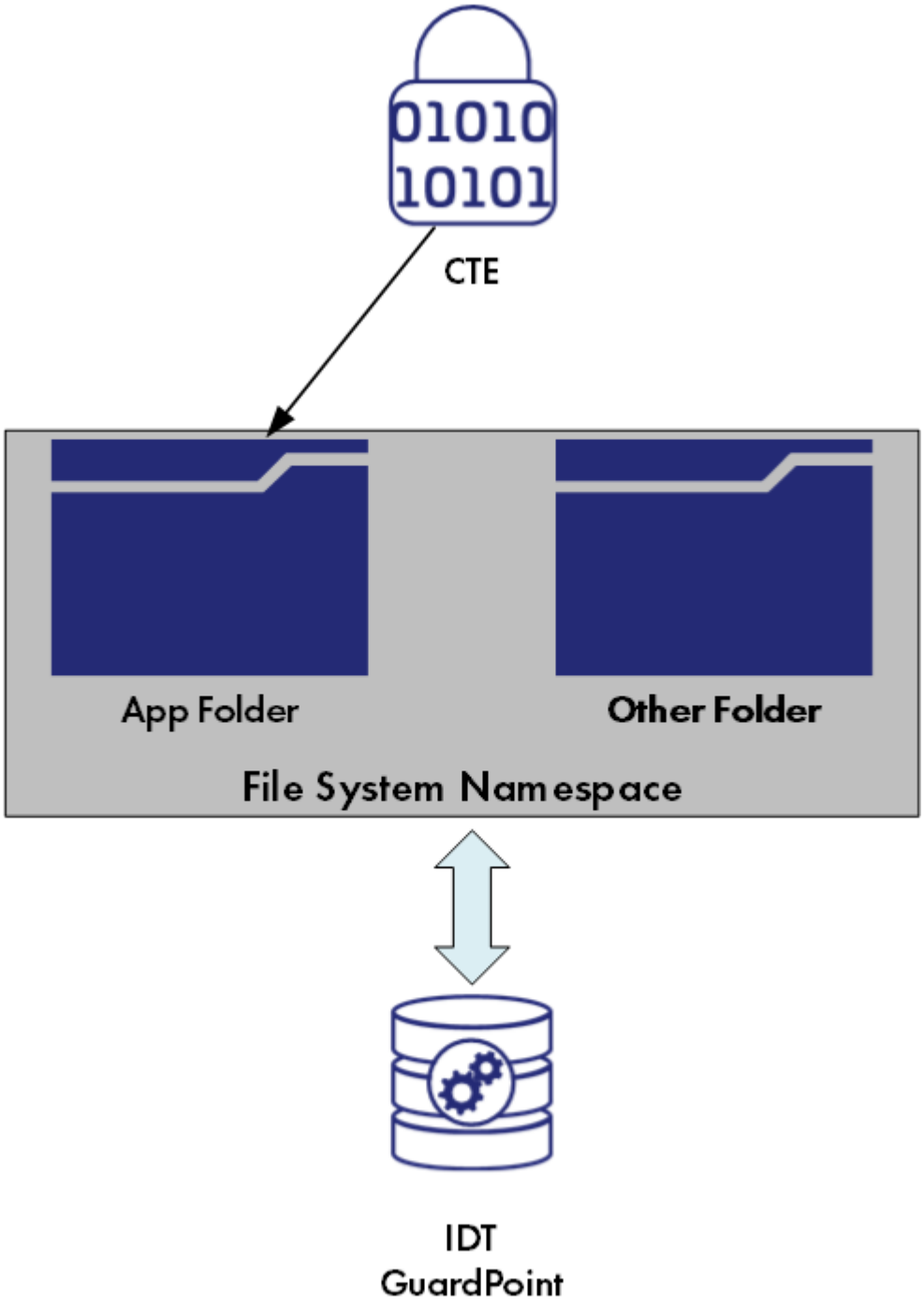
rule with an empty resource set for rekey. The policy may include access rules for user or process level access control.



Use Case 2: Device-Level GuardPoints

Protect structured or unstructured data stored in data files. The data files are organized inside one or more directories within a file system namespace, such as ext4 or XFS, without any protection on the directories or the file system namespace. In this use case, the file system resides in the device guarded as in-Place Data Transformation using a policy with a key rule and *no user specified access rule*. (Access rules are not applicable in this use case and should not be used.) Similar to use case 1, Linux policies supporting this use case can also specify the second key rule with an empty resource set for rekey.

File system resides in device guarded as an in-Place Data Transformation GuardPoint



Example

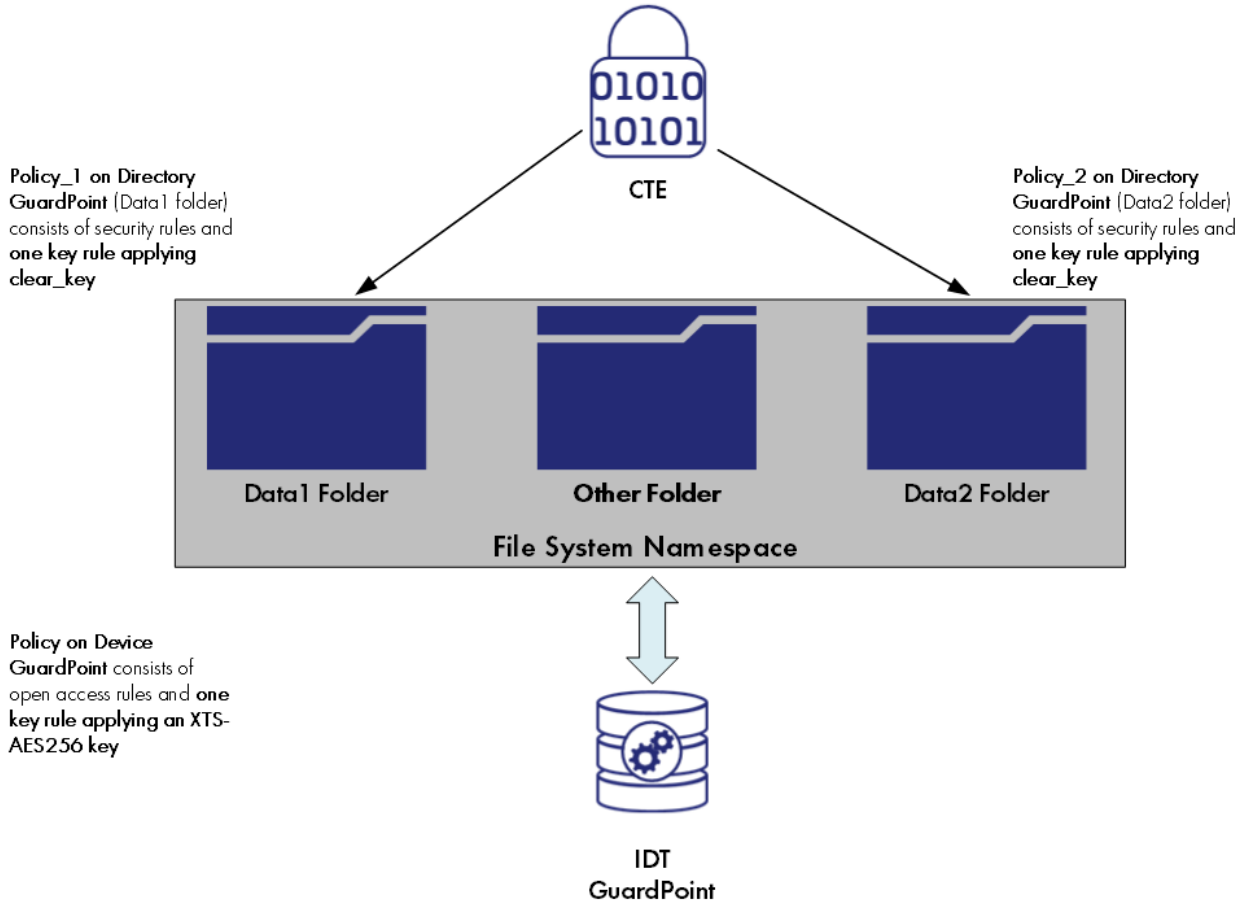
Below is an example of this use case where a Linux file system is created in an in-Place Data Transformation GuardPoint and then mounted. The policy used for the GuardPoint does not specify user or process-level access rules because I/O operations to the GuardPoint are from the file system module accessing the device on behalf of application I/O operations to the files inside the mounted file system.

```
# secfsd -status guard | grep sdh
/dev/sdh                    DEMO_POLICY_2           manualrawdevice  unguarded    not guarded  Inactive
# voradmin idt config new /dev/sdh
# secfsd -guard /dev/sdh
secfsd: Path is guarded
# voradmin idt status /dev/sdh
ESG Header on /dev/secvm/dev/sdh
        Version:                        1
        Change:                         0
        Notifications:                  None
        Storage Status:                 None
        Private Region Size:            129024 sectors
        Exported Device Size:           43266048 sectors
        Key UUID:                       b16445bd-db1e-3a8f-b829-5893dd2fd0b0
        Mount Path:                     None
# mkfs.xfs /dev/secvm/dev/sdh
meta-data=/dev/secvm/dev/sdh     isize=512    agcount=4, agsize=1352064 blks
         =                       sectsz=512   attr=2, projid32bit=1
         =                       crc=1        finobt=0, sparse=0
data     =                       bsize=4096   blocks=5408256, imaxpct=25
         =                       sunit=0      swidth=0 blks
naming   =version 2              bsize=4096   ascii-ci=0 ftype=1
log      =internal log           bsize=4096   blocks=2640, version=2
         =                       sectsz=512   sunit=0 blks, lazy-count=1
realtime =none                   extsz=4096   blocks=0, rtextents=0
# mount -t xfs /dev/secvm/dev/sdh /xfs
# mount | grep xfs
/dev/secvm/dev/sdh on /xfs type xfs (rw,relatime,seclabel,attr2,inode64,noquota)
```

Use Case 3: Directory-Level GuardPoints

Protect structured or unstructured data stored in data files. The data files are organized inside one or multiple directories within a file system namespace, such as ext4 or XFS, where the entire file system namespace is guarded with one policy as a Directory GuardPoint. In this use case, the file system resides in a device guarded as an in-Place Data Transformation GuardPoint. Similar to use case 1, Linux policies supporting this use case can also specify the second key rule with an empty resource set for rekey.

All Data in file system Device Encrypted through an in-Place Data Transformation GuardPoint



The second policy protecting the device is the same policy as use case 2.

Example

Below is an example of this use case where a file system created in a guarded device and mounted on `/xfs` is protected under a policy that denies root access to the files under `/xfs/dir1`:

```
# mount | grep xfs
/dev/secvm/dev/sdh on /xfs type xfs (rw,relatime,seclabel,attr2,inode64,noquota)
# find /xfs -print
/xfs
/xfs/non-secret
/xfs/dir1
/xfs/dir1/secret
# cat /xfs/dir1/secret
This file holds highly sensitive data.

# cat /xfs/non-secret
This file does not hold sensitive information.

# secfsd -guard /xfs
secfsd: Path is guarded
# secfsd -status guard | grep sdh
/dev/sdh                        DEMO_POLICY_2           manualrawdevice  guarded      guarded      N/A

# ls /xfs
dir1  non-secret
# cat /xfs/dir1/secret
cat: /xfs/dir1/secret: Permission denied
# cat /xfs/non-secret
This file does not hold sensitive information.
```

As depicted above, the root user is denied access to read/write the files associated with the resource set representing files under `/xfs/dir1` subdirectory.

Challenges with Root Access on Linux

As demonstrated in the example in use case 3, data is protected at two levels using two separate policies and GuardPoints. The data is encrypted at the device level through the policy on the device guarded as in-Place Data Transformation, and user access controls are enforced at the file system level through the policy on the mount point directory. Splitting the full protection through separate GuardPoints poses new challenges with respect to root privilege on Linux.

With the GuardPoint on the file system mount point enabled, the access rule(s) denying root access is enforced. However, when the GuardPoint on the file system mount point is disabled, root gains full access to the files in the file system. As shown below, the file holding secret information and protected against root is exposed as soon as the GuardPoint on file system mount point is disabled.

```
# secfsd -status guard | grep idt
/dev/sdh                  idt DEMO_POLICY_2      manualrawdevice  guarded      guarded      N/A
/xfs                      idt MOUNTED_FS_POLICY  manual           guarded      guarded      N/A
# cat /xfs/dir1/secret
cat: /xfs/dir1/secret: Permission denied

# secfsd -unguard /xfs
secfsd: Path is not guarded
# cat /xfs/dir1/secret
This file holds highly sensitive data.
```

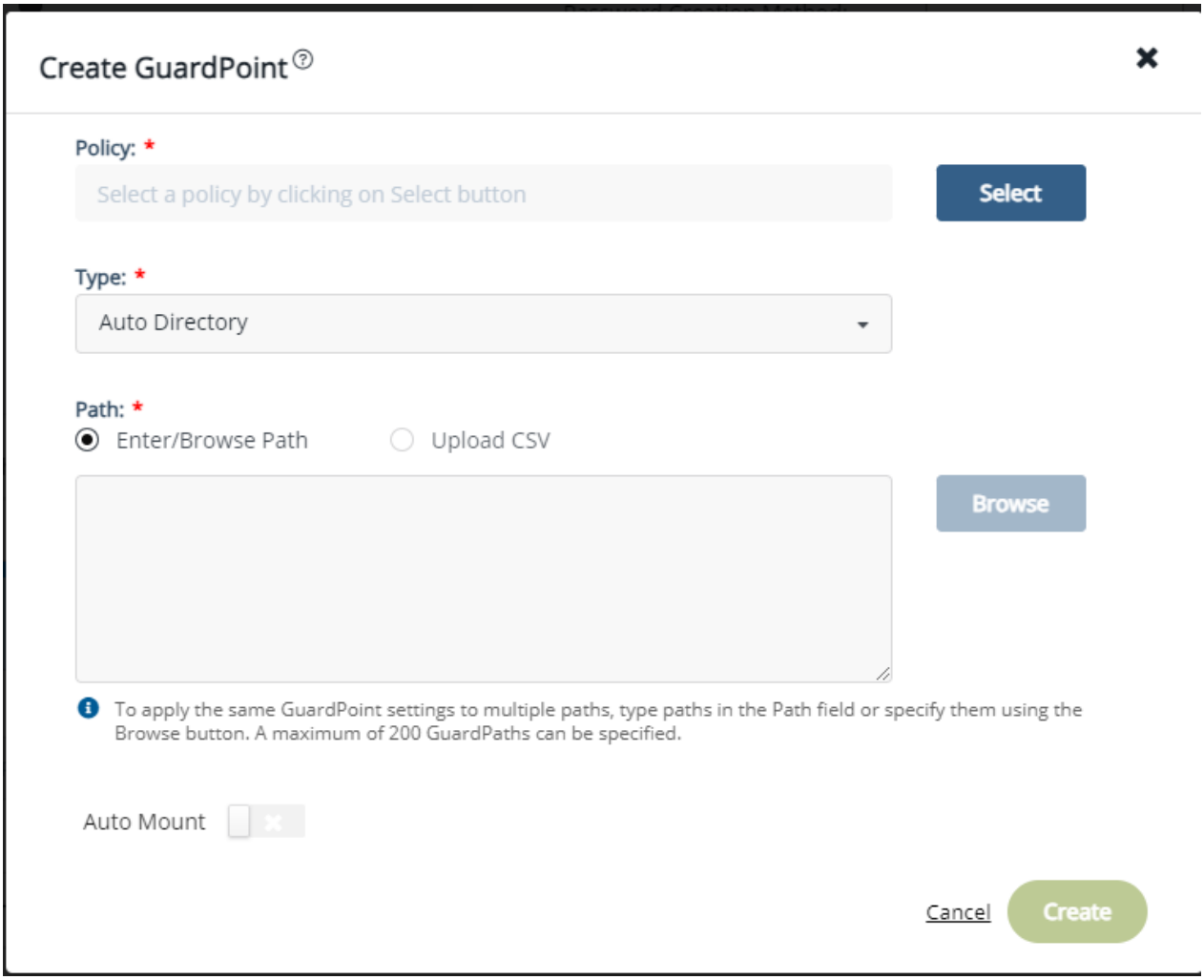
The next two sections describe the challenges with root access and solutions to overcome these challenges.

Challenge 1 - Deny root access to the files in mounted Linux file system

As the above example shows, the policy IDT_MOUNTED_FS_POLICY denies the root user access to the files associated with the resource set `dir1`. Enforcement of the rule become ineffective as soon as the GuardPoint on the file system mount point is disabled. Since data is in clear-text at the file system level, root would gain access to clear-text in the files associated with the resource set, which includes files with sensitive information.

The solution to this problem is to force the file system to unmount when the GuardPoint on the file system mount point is disabled. Basically, the file system is guarded/enabled immediately when the file system mounts, and the file system is unmounted as soon as the GuardPoint is disabled. To enforce this, the GuardPoint on the file system mount point directory must be guarded with **Auto Mount** option checked on the CipherTrust

Manager. With this option, CTE immediately guards the mount point directory as soon as the file system mounts, and similarly, CTE disables the GuardPoint before unmounting the file system. The file system mount point is guarded with **Auto Mount** option checked.



This solution imposes one policy protecting the entire file system namespace. Enforcement of a single policy over entire file system namespace may seem restrictive if you wish to impose different sets of access rules to different directories within the file system name. Basically, your option of enforcing one policy with a set of specific access rules for guarding a specific directory within the mounted file system namespace is not possible with this solution. Instead, you have to create a resource set for each directory, which you would have guarded, and specify the desired access rules specific to the directory through association of the rule with the resource set. Let's see the effect of auto-mount on root user attempts to view files that root users are not allowed to read. As shown below, the GuardPoint `/xfs` is automatically mounted as soon as the file system mounts, and the file system unmounts as soon as the GuardPoint is disabled, hence there is no opportunity for root, or any other privileged user, to read the protected files.

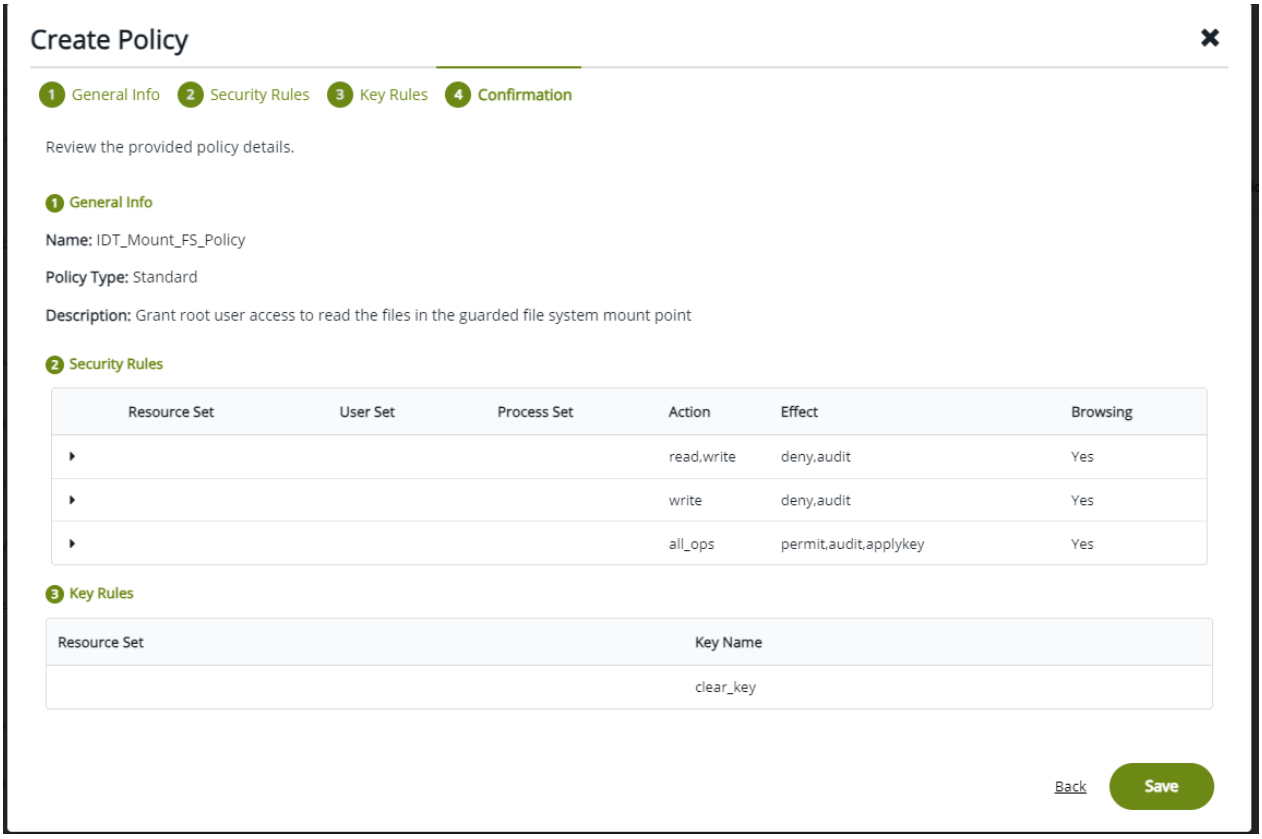
```
# mount -t xfs /dev/secvm/dev/sdh /xfs
# secfsd -status guard | grep idt
/dev/sdh                    idt DEMO_POLICY_2       manualrawdevice  guarded      guarded      N/A
/xfs                        idt MOUNTED_FS_POLICY   automount        guarded      guarded      N/A

# cat /xfs/dir1/secret
cat: /xfs/dir1/secret: Permission denied

# umount /xfs
# secfsd -status guard | grep idt
/dev/sdh                    idt DEMO_POLICY_2       manualrawdevice  guarded      guarded      N/A
/xfs                        idt MOUNTED_FS_POLICY   automount        guarded      not guarded  Inactive

# cat /xfs/dir1/secret
cat: /xfs/dir1/secret: No such file or directory
```

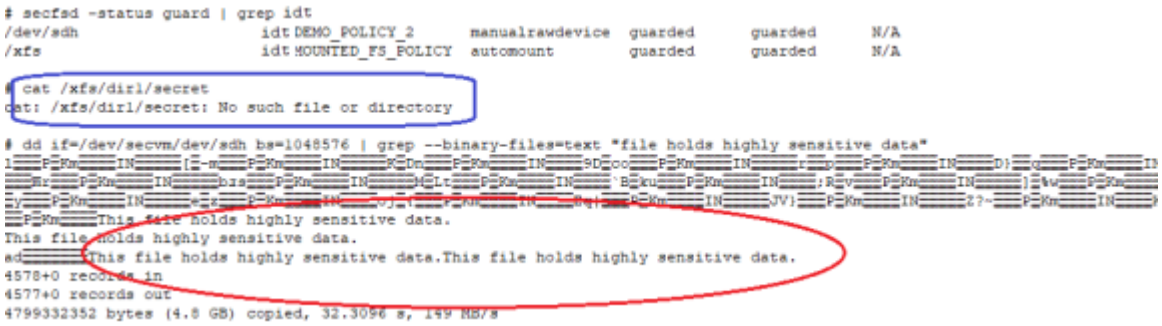
As depicted above, the policy IDT_MOUNTED_FS_POLICY enforces a single rule to block root access to the files under the `dir1` subdirectory under the mounted file system GuardPoint. You need to add another security rule to the policy to grant root user access to read the files under `dir2` subdirectory in the guarded file system mount point. Note that `dir1` and `dir2` may have been guarded separately under different policies.



Challenge 2 - Deny root access to view sensitive data in protected Linux files

Another challenge with root user privilege is that root can still view sensitive information stored in the in-Place Data Transformation GuardPoint device. As explained, the policy

IDT_MOUNTED_FS_POLICY denies root access to the read/writes files under `/xfs/dir1`. Although this policy enforces the rule on the Directory GuardPoint `/xfs`, the rule is not enforced if the root user dumps the content of the in-Place Data Transformation GuardPoint. As the example below shows, the root user can view sensitive information in the protected files under `/xfs/dir1`.



To block the root user from dumping context of the device, the policy on the in-Place Data Transformation GuardPoint must enforce a security rule that denies root access to the GuardPoint device. However, denying root access to GuardPoint devices is not feasible because file system utilities require root access for file system administration. Rather than enforcing complete denial of root access, you can impose a restriction that allows only a limited set of system utilities to access in-Place Data Transformation GuardPoints, such as `mount, fsck, mkfs, dmsetup,` etc., and deny access to other utilities, such as `dd`.

The solution is to restrict root to execute a limited set of utilities, which do not expose content of the file system devices, on in-Place Data Transformation Storage Devices. If the root user attempts to execute other utilities on the device, the root user will be denied access. You can impose this restriction by selecting the administrative utilities that root must execute on in-Place Data Transformation GuardPoints. The following contains the the list of utilities that root must be able to execute:

The following table includes recommendations on the system and file system specific utilities for inclusion in the signature set to allow or deny root execution.

EXT Utilities	Deny/Allow	XFS	Deny/Allow	Generic Utilities	Deny/Allow
`badblock`	Allow	`fsck.xfs`	Allow	`mount`	Allow
`debugfs`	Deny	`mkfs.xfs`	Allow	`umount`	Allow
`e2freefrag`	Allow	`xfs_repair`	Allow	`dmsetup`	Allow
`e2fsck`	Allow	`xfs_admin`	Allow		
`e2image`	Allow	`xfs_bmap`	Allow		
`e2label`	Allow	`xfs_check`	Allow		

EXT Utilities	Deny/Allow	XFS	Deny/Allow	Generic Utilities	Deny/Allow
e2undo	Allow	xfs_copy	Deny		
filefrag	Allow	xfs_db	Deny		
fsck.ext2	Allow	xfs_estimate	Allow		
fsck.ext3	Allow	xfs_freeze	Allow		
fsck.ext4	Allow	xfs_fsr	Allow		
logsave	Allow	xfs_growfs	Allow		
mke2fs	Allow	xfs_info	Allow		
mkfs.ext2	Allow	xfs_logprint	Allow		
mkfs.ext3	Allow	xfs_mdrestore	Allow		
mkfs.ext4	Allow	xfs_metadump	Allow		
resize2fs	Allow	xfs_mkfile	Deny		
tune2fs	Allow	xfs_ncheck	Allow		

To implement this solution, you can create a signature set on your CipherTrust Manager and add the system utilities that root is permitted to execute on in-Place Data Transformation GuardPoints. Those utilities can be added to the signature set for signing. After signing the binary files of those system utilities, you can add a security rule to the policy on the in-Place Data Transformation GuardPoint that grants root the right to execute the system utilities in the signature set to access the in-Place Data Transformation GuardPoint. In the above example, since `dd` is not in the signature set, the `dd` command is denied access to read the file system device guarded as an in-Place Data Transformation GuardPoint.

The resource set ESG_FS_ResourceSet consists of the binary files listed in the next screenshot. Following are the steps to add the security rule to limit root access:

1. Select the system utilities that must be granted access to in-Place Data Transformation Storage devices.
2. In the Policy Elements, click **Signature Set** > **Create Signature Set** to add a signature set.
3. On the **Create Signature Sets** page, enter the name of the signature set to create and then click **Next**.
4. Click **Add Source** to select the protected client where the file system utilities are located. Click **Apply** to display the available sources.

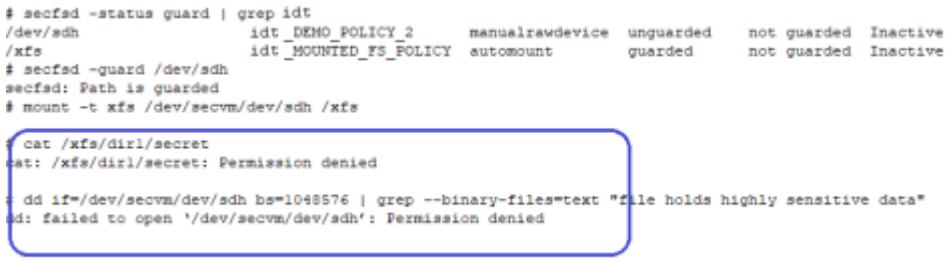
5. Click on a drive and **Add Source** to add each system utility from the selected client to the signature set. After adding the system utilities to the set, then click **Add**.
6. Click **Next** to review the system utilities added the signature set.
7. Click **Save** to sign the binary files on the selected protected Client.

At this point, you have created a signature set consisting of the system utilities that root is allowed to execute on in-Place Data Transformation storage devices. Next, create a process set from the signature set. The process set will be included in a security rule in the policy protecting the in-Place Data Transformation GuardPoint. The security rule will allow root, or other privileged users, to access the device only through the system utilities in the signature set. Continue with the following steps:

1. In the Policy Elements, click on **Process Sets**.
2. Click **Create Process Set** to add a process set.
3. Enter the name of the process set in the **Name** entry and then click **Next**.
4. Click **Create Process**.
5. Select the signature set that you just created and click **Add** and go back to **Create Process** to add the **Signature Set** to the Process set.
6. Click **Add** on the **Add Process Set** page.
7. Click **Next** and **Save** to create the process set associated with the selected signature set and the protected client.
8. Edit the policy protecting in-Place Data Transformation GuardPoints to add a security rule. In the following figure, the process set that you just created is included in the security rule. The rule allows only the processes listed in the process set to access the in-Place Data Transformation GuardPoints. This rule prevents any privileged user from reading or dumping the content of an in-Place Data Transformation GuardPoint.

After adding the security rule, the policy protecting in-Place Data Transformation GuardPoints will be the same policy as IDT_DEMO_POLICY_2.

After applying the revised policy over the in-Place Data Transformation GuardPoint, root can no longer dump the contents of the device.



Use Case 4: Using CTE-IDT with LVM

Security rules for access control enforcement on I/O operations are based on the context of real users and/or processes requesting the I/O operations. For IDT devices, security rules are checked and enforced at the CTE `secvm` layer. If the context of the real user or process that requested the I/O operation is not available at the `secvm` layer, then the enforcement of access rules is invalid and may have unpredictable results and potentially cause system failures.

The previous use cases have described environments where CTE can get the context of the real users or processes requesting I/O operations at the `secvm` layer and can properly enforce the access rules in those environments. For example, as described in Use Case 3: Directory-Level GuardPoints, CTE can detect when an IDT device is mounted with a file system and it can apply the access rules in the policy on the IDT device without enforcing the rules on I/O operations from the file system layer.

However, there are other environments where CTE cannot determine how the IDT device is actually used and whether a real user or process context is available at the `secvm` layer. In these environments, CTE cannot properly enforce the access control rules on the IDT device. An example of such an environment is Linux LVM (Logical Volume Manager). A physical device under LVM control is a shared storage resource that can be fully or partially allocated to a logical volume. Adding an IDT device as a physical device to LVM is fully supported for data encryption but not for access control enforcement. In this case, access rules cannot be enforced because when the logical volume with file system is mounted, the mount operation is on the logical volume, not the IDT device. Consequently, CTE cannot determine that the IDT device is part of mounted file system, and therefore, it cannot get the real user or process context for applying access rules.

In order to use CTE in an environment like LVM, the CTE system administrator needs to explicitly tell CTE whether or not it should enforce the access rules for I/O operations

on the device when they initialize the device using the `voradmin idt config` command. As described in Initialize a New Linux Device, this initialization must be done before the device is guarded as an in-Place Data Transformation GuardPoint for the first time.

In addition, for LVM specifically, the device must be a new device with no existing data. Existing LVM disks cannot be used with CTE unless CTE can get the proper context at the `secvm` layer.

When you initialize the new device, use the `voradmin idt config -noacc new [-c <n>] <device-name>` command where:

- `-noacc` (required) tells CTE to disable access control rules on this device at the `secvm` layer level.
- `new` (required) indicates that the device contains no data (it is a new disk).
- `<device-name>` (required). Specifies the device name. For example, `/dev/sdh`.

For example, if you want to initialize a new LVM Linux disk named `/dev/sdh`, type:

```
voradmin idt config -noacc new /dev/sdh
```

Caution

When you use `-noacc`, CTE will not perform *any* access checks on read/write operations to the specified device, even if the policy for guarding the device includes security rules enforcing read and/or write access checks. Before you use this option, make sure that your environment requires it. You should always use the options described in the first three use cases if at all possible, so that CTE can apply access controls to all I/O operations and therefore perform all required read/write access checks.

The following example shows the steps for initializing and guarding the device `/dev/sdb`, then adding or using the guarded device under LVM to create a file system.

1. Initialize the device using the `-noacc` option:

   ```
   voradmin idt config -noacc new /dev/sdb
   ```

2. Guard the device as described in Guard the Linux Device with an in-Place Data Transformation GuardPoint.

3. Create and mount the file system.

```
pvcreate /dev/secvm/dev/sdb
vgcreate secvm_sdb_vg /dev/secvm/dev/sdb
lvcreate -y -l 100%FREE -n my_volume
mkfs.xfs /dev/mapper/my_volume
mount /dev/mapper/my_volume /mnt
```

Initialize Linux IDT Devices

When you initialize a Linux in-Place Data Transformation device, the process creates a private region on the device for CTE to write the in-Place Data Transformation header along with metadata that identifies the ES device as a guarded device. The CTE private region also contains the metadata for the initial transformation of clear-text data on device to cipher-text, and for the subsequent transformation of cipher-text on the device to another encryption key as needed.

How you initialize the device depends on whether it is a new device or an existing device that already has data that needs to be transformed into cipher-text. For details, see:

- Initialize a New Linux Device
- Initialize and Resize an Existing Linux Device

Initialize a New Linux Device

Run the `voradmin idt config` new command to initialize a new device. The new option specifies that the device does not hold user data, and that CTE can reserve the first 63MB of storage on the device for the CTE private region. The remaining storage space is available for new user data. The device size reported to applications is the actual device size minus CTE private region size.

For a shared device that is accessed from multiple protected hosts, you must initialize the device only once and on only one protected host.

Note

To configure devices with multiple IO paths for Linux, see Guarding a CTE-in-Place Data Transformation Device with Multiple IO Paths on Linux.

1. Log into the device as root.
2. Run the `voradmin idt config new [-c <n>] <device-name>` command, where:
   - `new` (required) indicates that the device contains no data (it is a new disk). CTE will create the CTE private region at the beginning of the disk and the rest of the disk will be available for user data.
   - `-c <n>` (optional). If you use this option on Linux, CTE sets the number of data transformation jobs to run in parallel to the number specified in `<n>`. `<n>` can be an integer between 1 and 60 (default: 8).

     Each data transformation job transforms 1MB worth of data and requires CPU resources in addition to three I/O operations as part of data transformation. Each job reads 1MB of data from the device, preserves the data in the CTE private region, rekeys the data to cipher-text, and writes the transformed data to the device. If you increase the number of parallel jobs, the data transformation process will complete faster but there will be an increased performance impact on the system. Only increase the `-c` option if you are certain that the system resources are available to handle the additional load.

     The value for the `-c` option you specify here remains in effect for all subsequent data transformations (such as any data rekeys) until you specify a new value.
   - `<device-name>` (required). Specifies the device name. For example, `/dev/sdh`.

   For example, if you want to initialize a new Linux disk named `/dev/sdh` using 10 parallel data transformation jobs, you would specify:

   ```
   voradmin idt config new -c 10 /dev/sdh
   ```

3. To verify that the disk has been initialized, run the `voradmin idt status` command.

   ```
    voradmin idt status /dev/sdh
   Device /dev/sdh is configured to guard as an IDT GuardPoint.
   ```

4. At this point the Administrator can protect the device as an IDT GuardPoint through the CipherTrust Manager console. For details, see Guard the Linux Device with an ES GuardPoint.

Note

The initialization process prepares the device to be guarded but does not actually guard it. You need to assign an ES GuardPoint to the device in the CipherTrust Manager before the device is actually protected. In addition, the initialization process is only kept in memory until the device is guarded or rebooted. If the device is rebooted before you guard it, you will need to perform the initialization procedure again.

Initialize and Resize an Existing Linux Device

If the device has existing data, you need to use the `voradmin idt config xform` command to initialize the disk for CTE-in-Place Data Transformation. This command examines the current disk size and computes the size required to hold the existing data plus the CTE Private Region at the beginning of the device. After the CTE initialization is complete, you then need to resize the device before you can guard it with an ES GuardPoint.

The following procedure describes how to initialize the device for CTE. Note that the existing data is not altered in any way until after you perform this procedure and you guard the data with an ES GuardPoint. CTE does *not* begin transforming the data from clear-text to cipher-text until the ES GuardPoint has been applied and the encryption key has been pushed to the device through the GuardPoint Policy.

1. Log into the device as root.
2. Run the `voradmin idt config xform [-c <n>] <device-name>` command, where:
    - `xform` (required) indicates that the device contains existing data. CTE will transform all existing data on the device from clear-text to cipher-text as soon as you guard the device. The device will be inaccessible until the transformation is complete, and the device must remain offline during the entire transformation process. No user access will be permitted until all data has been transformed.

- `-c <n>` (optional). If you use this option on Linux, CTE sets the number of data transformation jobs to run in parallel to the number specified in `<n>`. `<n>` can be an integer between 1 and 60 (default: 8).

  Each data transformation job transforms 1MB worth of data and requires CPU resources in addition to three I/O operations as part of data transformation. Each job reads 1MB of data from the device, preserves the data in the CTE private region, rekeys the data to cipher-text, and writes the transformed data to the device. If you increase the number of parallel jobs, the data transformation process will complete faster but there will be an increased performance impact on the system. Only increase the `-c` option if you are certain that the system resources are available to handle the additional load.

  The value for the `-c` option you specify here remains in effect for all subsequent data transformations (such as any data rekeys) until you specify a new value.

- `<device-name>` (required). Specifies the device name. For example, `/dev/sdh`.

For example, if you want to initialize an existing Linux disk named /dev/sdh using 10 parallel data transformation jobs, you would specify:

```
 voradmin idt config xform -c 10 /dev/sdh
Device /dev/sdh must be resized to at least 21100544 sectors
(20606 MBs) before guarding as an IDT GuardPoint.
```

In this case you must manually resize the Linux disk by at least 20606 MBs before you can guard it. After you guard the disk, you can expand it again later but you cannot shrink it unless you remove the GuardPoint.

3. To verify that the disk has been initialized, run the `voradmin idt status` command.

```
 voradmin idt status /dev/sdh
Device /dev/sdh is configured to guard as an IDT GuardPoint.
```

4. At this point, you need to resize the device using your standard disk management tools before you can guard it. Make sure you increase the device size by at least the amount shown in the `voradmin idt config xform` message.
   You cannot assign an ES GuardPoint to the device until it has been resized. If you do not resize the device, the GuardPoint assignment will fail.

5. After the device has been resized, the Administrator can protect the device as an ES GuardPoint through the CipherTrust Manager console, as described in Guard the Linux Device with an ES GuardPoint.

Note

The initialization process prepares the device to be guarded but does not actually guard it. You need to assign an ES GuardPoint to the device in the CipherTrust Manager before the device is actually protected. In addition, the initialization process is only kept in memory until the device is guarded or rebooted. If the device is rebooted before you guard it, you will need to perform the initialization procedure again.

Guard the Linux Device with an IDT GuardPoint

After the device has been initialized, you can guard the device as an in-Place Data Transformation GuardPoint from the CipherTrust Manager console. For existing devices, as soon as the GuardPoint configuration has been pushed to the host and the status changes to guarded, CTE begins transforming the data on the disk using the encryption key associated with the GuardPoint Policy.

1. Log on to the CipherTrust Manager as an administrator of type Security with Host role permissions, type Domain and Security, or type All.
2. Make sure that you know what Policy you want to associate with the in-Place Data Transformation GuardPoint or create a new standard policy if needed.

   Note

   The policy you use must use an XTS/CBC-CS1 AES 256 key as the key rule.

3. Select **Hosts > Hosts** on the menu bar. The *Hosts* window opens.
4. Click the target host in the **Host Name** column. The Edit Host window opens to the General tab for the selected host.
5. Click the **GuardPoints** tab and then click **Guard**. The Guard window opens.

6. In the **Policy** field, select the Policy you identified or created earlier in this procedure. CTE will use the XTS/CBC-CS1 AES 256 key associated with this policy to encrypt the data on the device.

7. In the **Type** field, select either **Raw or Block Device (Auto Guard)** or **Raw or Block Device (Manual Guard)** .

   If you select **Auto Guard**, CTE starts the guard process as soon as the policy is pushed to the host. You enable, disable, guard, and unguard the GuardPoint in the CipherTrust Manager. If you want to have the device automatically guarded and mounted at system start up, add the device to `/etc/fstab`. For details, see Auto Mount Options for File System Devices on Linux.

   If you select **Manual Guard**, You guard the GuardPoint on the protected host with the `secfsd -guard <path>` and `secfsd -unguard <path>` commands. At system startup, you must guard the device and then mount it. This gives you more control over when data transformations occur because CTE will not start encrypting or rekeying the device until you manually start the process.

8. In the **Path** field, add the path for the device you want to guard. For example, `/dev/sdh`.

   If you specify multiple paths in this field, all specified devices will be guarded and all will be encrypted with the encryption key specified in the associated policy.

9. Click **OK**.

   The CipherTrust Manager pushes the policy and the GuardPoint configuration to the host and the CTE agent on the host writes the in-Place Data Transformation Header into the CTE private region for the specified devices. If this is a new device, the status changes to guarded and the disk is available for user access immediately.

   If there is existing data on the device, CTE begins transforming the data from clear-text to cipher-text as soon as the GuardPoint configuration is available and the device status changes to guarded. The device will remain inaccessible until this data transformation completes. The length of time required to transform the data depends on the amount of existing data and the number of parallel data transformation jobs specified on the `voradmin config` command.

   To see the data transformation progress, use the `voradmin idt xform status <device-name>` command.

After the device is initialized and guarded, the protected device must be accessed through the CTE device pathname. This pathname corresponds to the secvm device.

For example, the Linux device pathname `/dev/sdh` becomes `/dev/secvm/dev/sdh` as soon as the process is complete.

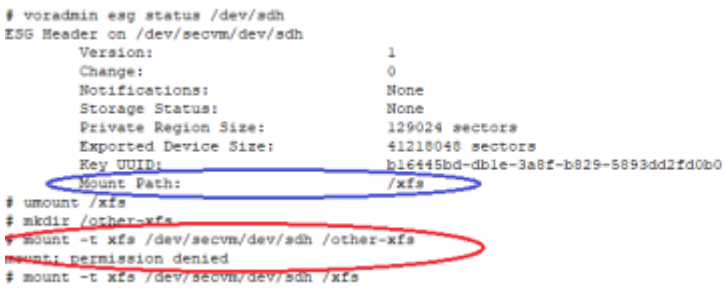
Note

- Be sure to use the secvm device name when using file system management tools such as `mkfs` and `fsck`.
- Do not use the device mapper names corresponding to in-Place Data Transformation GuardPoints for GuardPoint administration on protected hosts.

File System Mount Points on Linux

You can create and mount a file system on an IDT GuardPoint. CTE imposes one restriction on the mount point pathname selected for a device. Once you mount the device on a pathname, you cannot change the mount point to a different pathname. This restriction is enforced to allow the file system mount point to be guarded using a separate policy to enforce access control rules on the mounted file system namespace. For a use case involving a directory GuardPoint guarded over a mounted IDT GuardPoint, see Use Case 3: Directory-Level GuardPoints.

The following example shows the mount point of the IDT GuardPoint as the `/xfs` directory. The example also shows a failed attempt to mount the file system on a different directory pathname.



Auto Mount Options for File System Devices on Linux

IDT GuardPoints containing file systems can also be added to the `/etc/fstab` configuration file for auto mount at startup or unmount at shutdown. An entry can be for an IDT GuardPoint configured for Auto Guard or Manual Guard. For more information about Auto and Manual Guard options, see Guard the Linux Device with an IDT GuardPoint.

Use the device path corresponding to an IDT GuardPoint device when specifying `fstab` entries, such as `/dev/secvm/dev/sdh`. Do not use the native device pathnames, such as `/dev/sdh`, or device mapper device names. You must also include several settings in the `fstab` entry for each IDT GuardPoint, as shown in the following table:

Option	Description
`x-systemd.requires = secvm-barrier.service`	Ensure that the IDT GuardPoint is enabled before the device is mounted at startup and disabled after the device is unmounted at shutdown. The `secvm-barrier.service` service is a proxy for all the services that make up CTE.
`nofail`	The system boot will proceed without waiting for the IDT GuardPoint device if it can't be mounted successfully.
`x-systemd.wanted-by = <IDT device>.device`	Required for Linux distributions running `system` 242 or later. Instructs `systemd` to add a `Wants=` dependency on the ES device to ensure that, when the device becomes available, this mount operation is executed. `<es device>.device` is the name of the device specified in `fstab` with the '/' replaced with '-'. For example, `/dev/secvm/dev/sdb` becomes `dev-secvm-dev-sdb.device`.

This is an example of an entry in `/etc/fstab` for an IDT GuardPoint with an xfs file system that is mounted on `/xfs`:

```
/dev/secvm/dev/sdh /xfs xfs x-systemd.requires=secvm-barrier.service,
\
x-systemd.wanted-by=dev-secvm-dev-sdh.device,nofail 0 0
```

For information about configuring systemd for CTE, see CTE and systemd.

Guarding an IDT Device with Multiple IO Paths on Linux

Each individual IO path from a server node to a storage controller is treated as a separate device on the host. DM-Multipath on a Linux host provides a management framework to group the individual IO paths to the same LUN into a single multipath device. If you use DM-Multipath to manage devices on the protected host, the individual devices that correspond to each IO path to the LUN cannot be configured for guarding as ES GuardPoints, as those devices are under control of DM-Multipath. To guard such devices, you must guard the device mapper generated by DM-Multipath (multipathd) under the `/dev/mapper` directory.

The following example illustrates the procedure for guarding a device mapper generated device with the alias name `/dev/mapper/mpathA`.

1. Create a standard policy using an XTS key as the key rule.
2. On the host, prepare the device to be configured as ESG using the `voradmin` command with new or xform option. For example:

   ```
   voradmin idt config new /dev/mapper/mpathA
   ```

3. On the CipherTrust Manager, guard `/dev/mapper/mpathA` as Device GuardPoint using the policy created above.
4. For Manual-Guard configuration, enable the GuardPoint using the `secfsd` command as follows:

   ```
   secfsd -guard /dev/mapper/mpathA
   ```

5. For Auto-Guard, wait for the `/dev/mapper/mpathA` device to be guarded on the protected host.

6. Once the device is guarded, provide the pathname of the secvm device to applications and/or file system operations. For example, `/dev/secvm/dev/mapper/mpathA`.

Best Practices for the Migration of a legacy Raw Device GuardPoint to an IDT GuardPoint

Starting with CTE Agent v7.1.1, users can migrate an existing Raw Device GuardPoint to an in-Place Data Transformation GuardPoint. The migration steps described in this section make use of the in-Place Data Transformation process to re-encrypt existing data, within a device GuardPoint that was using an AES-CBC encryption key, to a new in-Place Data Transformation GuardPoint using an XTS/CBC-CS1 encryption key, all while preserving the existing data on the device.

You must use an in-Place Data Transformation policy for migration of legacy Raw Device GuardPoints to IDT.

Once the migration to an in-Place Data Transformation GuardPoint is completed, the new in-Place Data Transformation GuardPoint automatically makes use of the data reduction feature.

Migration Prerequisites

As part of the migration of a Raw Device GuardPoint to an in-Place Data Transformation GuardPoint, consider the following prerequisites:

- Access to the device, for all users and applications, is blocked during the entire migration process
- Devices that will be migrating to an in-Place Data Transformation GuardPoint are resized to provide sufficient disk space for IDT metadata
- Agent is running CTE v7.1.1 or a subsequent version
- An in-Place Data Transformation policy is required to guard the device as an in-Place Data Transformation GuardPoint. When creating the policy, set the AES-CBC key that is used in the Standard policy as the Current Key. Set the XTS/CBC-CS1 key as the New Key in the key rule.

Migration

Before beginning the migration of a Raw Device GuardPoint to an in-Place Data Transformation GuardPoint, stop all user and application access to the Raw Device GuardPoint.

1. Backup the device if possible.
2. On the CipherTrust Manager, navigate to the client tab and select the target client Name that contains the raw device that you want to migrate.
3. In the **GuardPoints** tab, select the target GuardPoint and click **Unguard** to unguard the raw device GuardPoint.
4. On the client, run the following command to configure the device to be guarded as an in-Place Data Transformation GuardPoint, type:

   ```
   voradmin idt config xform <device-name>
   ```

   > **Note**
   >
   > Make sure that you specify the native Linux device name of your device, such as `/dev/sdh` in the voradmin command, and resize the device before guarding the device using the IDT policy.

5. On the CipherTrust Manager, navigate to the client tab and select the target Host Name.
6. Under the **GuardPoints** tab, click **Guard** to set a Raw or Block Device GuardPoint using the In-Place Data Transformation policy created for this device for migration. If this option is not selected, the host will not enable the device as an in-Place Data Transformation GuardPoint.
7. Click **OK**.
   CTE begins transforming the data using the previous AES-CBC key and encrypting to the new XTS/CBC-CS1 key as soon as the device is guarded. During data transformation, the device remains inaccessible until this process completes. The length of time required to transform the data depends on the amount of existing data and the number of parallel data transformation jobs specified during the voradmin config command.

8. To see the data transformation progress, use the `voradmin idt xform status <device-name>` command.
9. After transformation is completed and the device is guarded, the protected device must be accessed through the CTE device pathname that corresponds to the raw device.
   For example, the Linux device pathname `/dev/sdh` becomes `/dev/secvm/dev/sdh` as soon as the guard process completes.

Alerts and Errors on Linux

This section lists the alerts and errors that may be encountered during system operations.

Encryption key on device has not been made available

This alert may appear as the reason for a GuardPoint not to have been enabled. The message appears in the output of `secfsd -status guard` command. The status indicates that the encryption key specified in the policy for the GuardPoint has not been made available to the protected host.

Solution: Check the host's connectivity with the Key Manager.

Specified policy disagrees with metadata set on the Guard Path

This alert may appear as the reason for a GuardPoint not to have been enabled. The message appears in the output of `secfsd -status guard` command. The status indicates that the key specified in the policy for the GuardPoint does not match with the key stored in the IDT Device Header.

Solution: Un-guard the device and check the name and UUID of the key in the IDT Device Header using `voradmin idt status <device-name>` and `voradmin idt status xform <device-name>` commands and compare the name and UUID with the key name specified in the policy. Correct the discrepancy and re-guard the device.

Device has not been configured for IDT-Capable

This alert may appear as the reason for a GuardPoint not to have been enabled. The message appears in the output of `secfsd -status guard` command. The status indicates that the device has been properly guarded as an IDT-Capable GuardPoint on the Key Manager but the device has not been initialized for guarding as IDT-Capable.

Solution: The most probable cause of this error is that you did not initialize the device for guarding as IDT-Capable on protected host. It's also possible that the guarded device has already configured for guarding as Ean IDT-Capable GuardPoint. If the device needs to be initialized for guarding as IDT-Capable, see Initializing an IDT-Capable Device.

Device not resized for guarding as IDT-Capable

This message appears in the output of `secfsd -status guard` command and indicates that the IDT-Capable GuardPoint was not successfully enabled. The status indicates that the newly guarded IDT-Capable GuardPoint, which has been initialized with `xform` option of `voradmin`, has not been resized to accommodate storage space for the CTE Private Region.

Solution: Unguard the IDT-Capable GuardPoint, resize the LUN and verify that the host sees the expanded size, then guard the IDT-Capable GuardPoint from the Key Manager.

Data transformation failed

This message appears in the output of `secfsd -status guard` command and indicates that the IDT-Capable GuardPoint was not successfully guarded. The status indicates that the protected host encountered an error while transforming the data on the device during IDT.

Solution: Consult with the system and/or storage admin to check on the health of the LUN in the storage array. You may contact Thales Support for troubleshooting and recovery if there has not been a report of any error on the LUN.

Data transformation in progress

This message appears in the output of `secfsd -status guard` command and indicates that the IDT-Capable GuardPoint was not successfully enabled. The status indicates that the protected host is transforming the data on the device.

Solution: You must wait for data transformation to complete. Check the status of transformation by running `voradmin idt status xform <device name>`. Access to the device is blocked until transformation completes.

Device <device-name> is configured to guard as IDT-Capable GuardPoint

This error message is reported by the `voradmin` command when initializing a device as an IDT-Capable GuardPoint that has already been initialized for guarding as an IDT-Capable GuardPoint.

Solution: The device has already been initialized for guarding as IDT-Capable and is probably waiting to be guarded as an IDT-Capable GuardPoint through the Key Manager. Alternatively, if you want to remove the IDT-Capable configuration, use the `voradmin idt delete <device-name>` command.

Device <device-name> is configured as IDT-Capable GuardPoint

This error message is reported by the `voradmin` command when initializing a device that is already being guarded as an IDT-Capable GuardPoint.

Solution: The device is already guarded by an IDT-Capable GuardPoint.

GuardPoint for device <device-name> still guarded on Key Manager

This error message is reported by the `voradmin` command when attempting to initialize a device for rekey or removing the device as IDT-Capable.

Solution: Unguard the IDT-Capable GuardPoint, wait for the protected host to process the update, and then rerun the `voradmin` command.

Failed to open device <device-name>, error Device or resource busy

This message occurs when `voradmin` detects that the target device is busy.

Solution: The device may already be in use by other application. Rerun the `voradmin` command when the device is no longer in use.

Device <device-name> is not configured as IDT-Capable

This error message is reported by the `voradmin` command when attempting to delete a device as an IDT-Capable GuardPoint.

Solution: This message indicates that the target device has been not initialized for guarding as IDT-Capable. The message may also be reported if the specified device has been initialized for guarding as IDT-Capable but it has not been guarded yet. In this case, it removes the preparation made by `voradmin`. You can remove the IDT-Capable configuration status on the device by running `voradmin idt delete <device-name>`. You may see the same error message again, and if you do, you can ignore it.

Abort! Error: Could not stop secfs, secvm device(s) busy

This error occurs during CTE shutdown when there is a busy CTE protected device.

Solution: Verify that all applications directly accessing `secvm` protected GuardPoints have been shut down. Ensure that all file systems on top of a `secvm` protected devices are under the control of `systemd` and have been unmounted before attempting to shut down the agent.

Abort! Error: Could not unmount file systems

This error occurs during CTE shutdown when a file system under the control of `systemd` fails to unmount.

Solution: Verify that file systems on top of an IDT-Capable GuardPoint devices are not busy and then rerun the agent shut down command.

A dependency job for idt.mount failed. See 'journalctl -xe' for details

This error occurs during CTE startup when system fails to mount a file system. This error message is typically accompanied by a long timeout during the CTE startup process.

Solution: Check that the underlying device is available and that the IDT-Capable GuardPoint was successfully applied on the device. Once the device is available, the file system will automatically finish mounting.

ESG/IDT-ALERT: IO error on header for [GuardPoint]

This is an alert message to the Key Manager. It occurs when CTE encounters a general error when attempting to access the private region on an IDT-Capable device.

Solution: An I/O error attempting to read or write to the device may have been caused by errors on the host or storage array. Consult with the system and/or storage admin to check on the health of the LUN in the storage array. You may contact Thales Support for troubleshooting and recovery if there has been no report of errors on the LUN.

ESG/IDT-ALERT: Data transformation failure on [GuardPoint]

This is an alert message to the Key Manager. It occurs when protected host encounters an error transforming the data on device during IDT process.

Solution: Contact Thales Support for troubleshooting and recovery.

ESG/IDT-INFO: Data transformation complete on [GuardPoint]

This message is a notification to the Key Manager admin that the protected host has completed the data transformation of the specified IDT-Capable GuardPoint.

ESG/IDT-ALERT: Failed to resize <device-name>

This alert is a notification to the Key Manager admin that the protected host has failed to update the change to the device size in the IDT Device Header on the device.

Solution: Contact Thales Support for troubleshooting and recovery.

FSADM-ALERT: ESG/IDT required Signature Set for system utilities may have to be resigned

This alert is a notification to the Key Manager admin that the recent system upgrade to the protected host may have updated the binary files listed in the signature set for restricting root access.

Solution: Upon this notification you must immediately re-sign the affected or all the binary files to prevent them from accessing protected data.

File System is not automatically mounted after IDT completes

An IDT-Capable device with a file system that has been configured to automatically mount may fail to automatically mount while the device undergoes data transformation through IDT. Following is a sample log message in the kernel ring buffer that reports a failed attempt to access a device during IDT. You can ignore this message.

```
Vormetric SecVM: secvm_map during IDT ((dc 00000000c3537611)
```

When access to a device is to mount the file system, the kernel ring buffer may also report a second error message indicating that a file system failed to mount due to data

corruption. You can ignore this message. This occurs because the mount command is issued after the device has been guarded but before data transformation through IDT is complete. While IDT is in progress, the device cannot be used, so any attempt to mount the file system will fail.

Solution: Manually mount the file system after IDT is complete.

Upgrading CTE on Linux

This chapter describes how to upgrade an existing CipherTrust Transparent Encryption (CTE) client and contains the following sections:

Upgrading CTE

This section describes the generic instructions for interactively upgrading CTE. If there are any changes to this procedure for the current release of CTE, those changes will be documented in the CTE Release Notes.

If you want to schedule an upgrade to occur the next time the system boots, see Scheduled Upgrade Feature.

1. Stop any application accessing files in the GuardPoint.
2. Log on to the host where you will upgrade CTE. You must have root access.
3. Copy or mount the installation file onto the host system.
4. Start the upgrade by executing the install program for the release to which you want to upgrade. If you want to automatically accept the CTE License Agreement, you can include the `-y` parameter.
   For example, the following command upgrades the product to version after the user manually reviews and accepts the CTE License Agreement:

```
./vee-fs-7.3.0-135-rh8-x86_64.bin
```

The following command upgrades the product to version but automatically accepts the CTE License Agreement:

```
./vee-fs-7.3.0-135-rh8-x86_64.bin -y
```

5. Follow the prompts. During an upgrade, the following message displays. Enter Y at the prompt:

```
Upgrade detected: this product will be stopped and restarted.
Do you wish to proceed with the upgrade? (Y/N) [Y]: Y
Installation success.
You will not do the registration steps since CTE is already reg
istered with the Key Manager.
```

6. To verify that the upgrade was successful, use the `vmd -v` command:

```
vmd -v
Version 6, Service Pack 2
<Release.build-number>
2022-02-04
Copyright (c) 2009-2022, Thales. All rights reserved.
```

Scheduled Upgrade Feature

- Warnings for CTE for Linux
- Using the Scheduled Upgrade Feature
- Performing a Manual Upgrade When an Upgrade is Already Scheduled

Note

Scheduled upgrade on reboot is not supported on HDFS nodes.

Warnings for CTE for Linux

- Prior to upgrading your system, perform a backup or take a snapshot of your system.
- As with prior CTE versions, Key Manager connectivity is required during upgrade.
- Yum updates, or OS patches, should be done prior to CTE upgrade on reboot.
- If you upgrade from a compatible kernel to an incompatible kernel, the `secfs` module will fail to load on the next reboot.
- You may see the following behavior if the upgrade on reboot fails due to a crash, or a power failure, (this is similar to a failure during a normal upgrade).
    - If a crash, or power failure, occurs before the upgrade executes, the upgrade will not take place, and the currently installed CTE version continues to run after the reboot. Restart the system to upgrade successfully.
    - If a crash, or power failure, occurs during the upgrade, CTE may enter an inconsistent state. Perform a restore from your backup, or roll back to the snapshot that you just took. Then, start the upgrade again.
    - If a crash, or power failure, occurs after a successful upgrade, then the new version will run on the next reboot. No user intervention is required in this case.
- During reboot or shutdown, all applications and services dependent on CTE services must be stopped before a scheduled update takes place. Failure to stop these services can result in an aborted scheduled upgrade during the system reboot. Examples of situations that may cause an aborted upgrade are applications with open files in a CTE GuardPoint, or a third party anti-virus software doing periodic scans.

  For examples of how to set up CTE start/stop dependencies with other programs, see CTE and systemd.

Using the Scheduled Upgrade Feature

The following procedure describes how to use voradmin to schedule an upgrade that will be applied the next time the machine reboots.

1. If you want to check which version of CTE for Linux you currently have installed, use the `vmd -v` command:

```
vmd -v
Version 6, Service Pack 2
```

```
7.0.0.47
2022-02-04
Copyright (c) 2009-2022, Thales. All rights reserved.
```

2. To schedule an upgrade on reboot, use the following commands:

```
voradmin upgrade schedule <path_to_CTE_installer_binary>  y [-t
<custom_extraction_path>]
```

where:

- `<path to CTE installer>` is the full path to the CTE installation file for the release to which you want to upgrade. For example, `./vee-fs-7.0.0-129-rh8-x86_64.bin`.
- `-y` is an optional parameter that automatically accepts the CTE License Agreement. If you do not specify this parameter, the installer displays the CTE License Agreement and you must manually accept it before the upgrade can be scheduled.
- `[-t <custom_extraction_path>]` is an optional parameter that specifies the path to a custom binary extraction path directory in which you want CTE to store the temporary files it needs during the upgrade. The default is `/var/tmp/`, but in some systems, `/var/tmp/` is restricted and not available for use.

  Exceptions: Do not use the -t option on protected paths, GuardPoint paths, or paths which do not have sufficient permissions to copy/extract the target binary.

For example, if you are upgrading to version 7.2.0.xx and you want to automatically accept the license agreement and use a custom directory, you would type:

```
voradmin upgrade schedule ./vee-fs-7.2.0-98-rh8-x86_64 -y -t /
my_custom_dir
```

> **Note**
>
> The [-t] option is only supported by CTE v7.2 and subsequent versions.

3. If you want to verify that the upgrade was successfully scheduled, use the `voradmin upgrade show` command:

```
voradmin upgrade show
Upgrade on reboot is currently scheduled.
Current CTE version is 7.0.0.47, upgrade on reboot scheduled
for CTE 7.1.0.66.
```

4. Reboot the machine, then log in and verify that the upgrade was successful.

```
vmd -v
Version 6, Service Pack 2
7.1.0.66
2022-02-04
Copyright (c) 2009-2022, Thales. All rights reserved.
```

> **Note**
>
> Appropriate logs will be logged in syslog.

Performing a Manual Upgrade When an Upgrade is Already Scheduled

If an administrator runs a manual upgrade after an upgrade has already been scheduled, the installer displays the following warning:

```
WARNING: upgrade on reboot is already scheduled for 7.1.0.66.

Do you want to cancel scheduled upgrade on reboot ? (Y/N) [Y] :
```

If the administrator does *not* cancel the scheduled upgrade, the scheduled upgrade takes precedence and the manual upgrade fails with the message:

```
Already scheduled upgrade on reboot remains intact.

Installation failure.
```

If the administrator wants to proceed with the manual upgrade immediately, they must enter Y at the prompt to cancel the scheduled upgrade:

```
WARNING: upgrade on reboot is already scheduled for 7.1.0.66.

Do you want to cancel scheduled upgrade on reboot ? (Y/N) [Y] : Y

WARNING: upgrade on reboot is cancelled for 7.1.0.66. Proceeding wi
th manual upgrade.

Upgrade detected: this product will be stopped and restarted.

Do you wish to proceed with the upgrade? (Y/N) [Y]: Y

.............

Upgrade success.
```

To verify that the upgrade succeeded, the administrator can use the `vmd -v` command:

```
vmd -v
Version 6, Service Pack 2
7.1.0.66
2022-02-04
Copyright (c) 2009-2022, Thales. All rights reserved.
```

To cancel an existing scheduled upgrade on reboot:

```
voradmin upgrade cancel
Successfully cancelled upgrade on reboot
```

Upgrading CTE agents in an LDT Communication Group from 7.4.0 to 7.5.0 and post 7.5.0

When upgrading an LDT Communication Group, you must stop CTE on all of the nodes to be upgraded before upgrading them.

1. Disable the GuardPoints on the nodes to be upgraded:
   a. Go to the CipherTrust Manager UI.
   b. In the GuardPoint window, click the ellipsis on the right side of a GuardPoint and select disable to disable the GuardPoint.
   c. Repeat the steps for all of the GuardPoints on the nodes to be upgraded.
2. Stop the CTE service on all of the nodes to ensure that all of the GuardPoints are unguarded.
   For **Windows**:
   a. Go to **Control Panel > Services (local)**.
   b. Select **secfsd**.
   c. Select **Stop the Service**.
   For **Linux**, type:

```
/etc/vormetric/secfs stop
```

3. Upgrade CTE on all of the nodes.
4. If your setup contains **manual directory** type GuardPoints, then you must run `secfsd -guard <gp>` to guard the GuardPoints again after the upgrade.
5. Verify that the GuardPoints are guarded on all of the nodes. Type:

```
# secfsd -status guard
```

Uninstalling CTE from Linux

Considerations

- The CTE Agent must be removed from the Linux host before the host is removed from the key manager with which it is registered.

- Database applications like DB2 and Oracle can lock the user space while they run. If the uninstall fails because a GuardPoint is in use, determine which applications are using the files in the GuardPoint and stop them. Then run the uninstall again.
- Commands like `fuser` and `lsof` might not reveal an active GuardPoint because they detect active usage, not locked states. Although it may appear that a GuardPoint is inactive, it may be in a locked state. Under this condition, software removal may fail with an error similar to the following:

```
/home: device is busy.
```

Procedure

1. Stop any application from accessing files in the GuardPoint.
2. In the key manager with which this host is registered, do the following:
    - Decrypt any data you want to use after uninstall. After the CTE Agent software is removed, access to data is no longer controlled. If data was encrypted, it will remain encrypted. If decrypted or copied out of the GuardPoint, the data is visible as clear text.

      This decryption must be done on *every* GuardPoint on the host if you want to access all existing data on the host.
    - Make sure the Agent and System locks have been disabled for the host.
    - Thales recommends that you remove all GuardPoints from the host before you uninstall the CTE Agent.

   Do not remove the host from the key manager yet.
3. Log on to the host as `root`.
4. Change the directory to an unguarded location (for example, /.).

Caution

Do not change (`cd`) into the `/opt/vormetric` directory or into any directory below `/opt/vormetric`. If you run the uninstaller from `/opt/vormetric` or any of its subdirectories, the package removal utility may fail and return the following message:

```
You are not allowed to uninstall from the /opt/vormetric directory or any of its sub-directories.
Agent uninstallation was unsuccessful.
```

5. Start the uninstall. Type:

```
 /opt/vormetric/DataSecurityExpert/agent/vmd/bin/uninstall
Would you like to uninstall the vee-fs package? (Y/N) [Y]: Y
.............
Success!
```

6. Remove the host record from the key manager.

7. **Remove the host record from the key manager.**

CM host registration when switching Agent from CTE-U to CTE

When you have a CTE-U agent that is already registered with CipherTrust Manager, and you want to change it from CTE-U to CTE, you **must** perform the following steps in order:

1. Unenroll the client
2. Delete the entry from CipherTrust Manager
3. Add the client back into CipherTrust Manager
4. Install and Register the new Agent

Installation and Registration on Linux

1. Log on to the host where you will install the CTE Agent as `root`. You cannot install the CTE Agent without `root` access.
2. Copy or mount the installation file to the host system. If necessary, make the file executable with the `chmod` command.
3. Install the CTE Agent. A typical installation uses the following syntax:

   ```
   ./vee-fs-<release>-<build>-<system>.bin
   ```

   For example:

   ```
   ./vee-fs-7.3.0-135-rh8-x86_64.bin
   ```

   To install the CTE Agent in a custom directory, use the `-d <custom-dir>` option. For example:

   ```
   ./vee-fs-7.3.0-135-rh8-x86_64.bin -d /home/my-cte-dir/
   ```

   Note

   If possible, Thales recommends that you use the default directory `/opt/vormetric`.

   To view all installer options, use the `-h` parameter. For example:

   ```
   ./vee-fs-7.3.0-135-rh8-x86_64.bin -h
   ```

4. The Thales License Agreement displays. When prompted, type **Y** and press Enter to accept.
   The install script installs the CTE Agent software in either `/opt/vormetric` or your custom installation directory and then prompts you about registering the CTE Agent with a key manager.

Welcome to the CipherTrust Transparent Encryption File System Agent Registration Program.

```
Agent Type: CipherTrust Transparent Encryption File System Agent
Agent Version: <Release.build-number>

In order to register with a CipherTrust Manager you need a valid
registration token from the CM.

Do you want to continue with agent registration? (Y/N) [Y]:
```

5. Type **N** and press Enter to end the installation procedure without registering the CTE Agent with either key manager.
6. Enter **Y** to continue with the registration process. The install script prompts you to enter the host name or IP address of the CipherTrust Manager with which you want to register CTE. For example:

   Do you want to continue with agent registration? (Y/N) [Y]: Y

   ```
   Please enter the primary key manager host name: 10.3.200.141
   ```

   > **Note**
   >
   > The default communication port is 443. If you want to specify a different communication port, enter it with the primary key manager host name in the format: *<hostName>:<port#>*

   You entered the host name 10.3.200.141
   Is this host name correct? (Y/N) [Y]: Y

7. Enter the client host name when prompted.

   ```
   Please enter the host name of this machine, or select from the fo
   llowing list.

   [1] sys31186.qa.com
   [2] 10.3.31.186
   ```

```
Enter a number, or type a different host name or IP address in ma
nually:
What is the name of this machine? [1]: 2
You selected "10.3.31.186".
```

8. Enter the CipherTrust Manager registration token, profile name, host group and host description. If you omit the profile name, CipherTrust Manager associates the default client profile with this client.

```
Please enter the registration token: 12345
Please enter the profile name for this host: My-Profile
Please enter the host group name for this host, if any:
Please enter a description for this host: RHEL7 system West Coast
Datacenter

Token            : 12345
Profile name     : My-Profile
Host Group       : (none)
Host description : RHEL7 system West Coast Datacenter
Are the above values correct?  (Y/N) [Y]: Y
```

9. At the hardware association prompt, select whether you want to enable the hardware association feature to prevent cloning. The default is Y (enabled):

   It is possible to associate this installation with the hardware of this machine. If selected, the agent will not contact the key manager or use any cryptographic keys if any of this machine's hardware is changed. This can be rectified by running this registration program again. Do you want to enable this functionality? (Y/N) [Y]: Y

10. At the LDT prompt, specify that you want this client to use CTE-LDT by typing **Y** and pressing Enter:

```
Do you want this host to have LDT support enabled on the server?
(Y/N) [N]: Y
```

11. If you are planning to create GuardPoints on NFS shares, enter the name of the LDT Communication Group that this node will join.

```
Enter the LDT Communication Group name: LCG1
```

> **Warning**
>
> **The registration token, profile name, client group name and LDT Communication Group name are case-sensitive. If any of these are entered incorrectly, the client registration will not succeed. If the registration fails, click Back in the installer and verify that the case is correct for all entries on this page**.

12. At the Cloud Object Storage (COS) prompt, specify whether you want this client to use CTE COS.

```
Do you want to configure this host for Cloud Object Storage? (Y/N)
[N]:
```

13. CTE finishes the installation and registration process.

```
Generating key pair for the kernel component...done.
Extracting SECFS key
Generating EC certificate signing request for the vmd...done.
Signing certificate...done.
Enrolling agent with service on 10.3.200.141...done.
Successfully registered the CipherTrust Transparent Encryption CTE
Agent with the CipherTrust Manager on 10.3.200.141.

Installation success.
```

14. In CipherTrust Manager, change the client password using the manual password creation method. This password allows users to access encrypted data if the client is ever disconnected from the CipherTrust Manager. For details on changing the password, see the CipherTrust Manager documentation.

Support Contacts

If you encounter a problem while installing, registering, or operating the product, please refer to the documentation before contacting support. If you cannot resolve the issue, contact your supplier or Thales Customer Support.

Thales Customer Support operates 24 hours a day, 7 days a week. Your level of access to this service is governed by the support plan arrangements made between Thales and your organization. Please consult this support plan for further information about your entitlements, including the hours when telephone support is available to you.

Customer Support Portal

The Customer Support Portal, at Thales Customer Support, is where you can find solutions for most common problems. The Customer Support Portal is a comprehensive, fully searchable database of support resources, including software and firmware downloads, release notes listing known problems and workarounds, a knowledge base, FAQs, product documentation, technical notes, and more. You can also use the portal to create and manage support cases.

Tip

You require an account to access the Customer Support Portal. To create a new account, go to the portal and click on the REGISTER link.

Telephone Support

If you have an urgent problem, or cannot access the Customer Support Portal, you can contact Thales Customer Support by telephone at +1 410-931-7520. Additional local telephone support numbers are listed on the support portal.

Email Support

You can also contact technical support by email at technical.support@Thales.com.