THALES

Vormetric Data Security Platform

Vormetric Transparent Encryption (VTE)

Data Transformation Guide

50-1000160-07

AUGUST 21, 2018
DOCUMENT VERSION 1

Vormetric Data Security
VTE Data Transformation Guide
Version v6.1.0, Document Version 1
August 21, 2018
Produced in the United States of America

NOTICES, LICENSES, AND USE RESTRICTIONS

7,143,288

7,283,538

7,334,124

Thales Data Security includes a restricted license to the embedded IBM DB2 database. That license stipulates that the database may only be used in conjunction with the Thales Vormetric Security Server. The license for the embedded DB2 database may not be transferred and does not authorize the use of IBM or 3rd party tools to access the database directly.

Contents

PREFACE

The *Data Transformation Guide* describes how to use Vormetric Transparent Encryption (VTE) on hosts to transform GuardPoint data from clear to encrypted or from encrypted to clear.

DOCUMENTATION VERSION HISTORY

The following table describes the changes made for each document version.

Table 1: Documentation Version History

	Date	Changes
v6.0.1 v1	09/15/17	Release for v6.0.1
v6.0.2 v1	10/10/17	Release for v6.0.2
v6.0.2 v2	12/6/17	Release for v6.0.2 patch. Changed images to match Thales standard.
v6.0.3 v1	03/6/18	Rereleased the 6.0.2 patch version as 6.0.3 GA release.
v6.1.0 v1	08/21/18	Release for v6.1.0

SCOPE

This document describes the detailed steps for encrypting and rekeying GuardPoint data.

INTENDED AUDIENCE

The *VTE Data Transformation Guide* is for security teams who want to rekey the existing GuardPoint data, or who need to perform an initial encryption of their GuardPoint data.

Assumptions

The Data Transformation Guide assumes that you have:

- A working Vormetric Data Security Platform installed and configured (DSM and protected hosts).

- Experience creating policies (see the *Getting Started Guide* and the *VDS Administrators Guide*).
- Root access to the protected host containing the data to be encrypted or rekeyed.

SERVICE UPDATES AND SUPPORT INFORMATION

The license agreement that you have entered into to acquire the Thales products (“License Agreement") defines software updates and upgrades, support and services, and governs the terms under which they are provided. Any statements made in this guide or collateral documents that conflict with the definitions or terms in the License Agreement, shall be superseded by the definitions and terms of the License Agreement. Any references made to “upgrades” in this guide or collateral documentation can apply either to a software update or upgrade.

For support and troubleshooting issues:

- https://help.thalesesecurity.com/hc/en-us
- Email questions to support@vormetric.com or call 877-267-3247

For Vormetric Sales:

- http://enterprise-encryption.vormetric.com/contact-sales.html
- Email questions to sales@vormetric.com or call 888-267-3732

1 Data Transformation Overview

This chapter provides a detailed overview of the data transformation process. It consists of the following sections:

The Data Transformation Process

Data transformation is the process of transforming GuardPoint data from:

- clear to encrypted
- encrypted to clear
- encrypted with one key to encrypted with version of a another key

Data transformation is used for:

- **Initial data transformation** — This is encrypting GuardPoint data for the first time.
- **Rekeying** — Changing the encryption key for GuardPoint data, also called *key rotation*.
- **Reverse transformation** — Decrypting GuardPoint data to clear text. (Not a common procedure.)

In addition to being disruptive to data center operations, data transformation is complex. It is strongly recommended that you read and understand this section before proceeding to the initial data transformation and rekey chapters.

How Vormetric Data Security Protects Files

The File System Agent encrypts the data within a file one block at a time. It does not encrypt file metadata such a file's name or size, thus enabling administrators to manage files without being able to view or modify their contents. Whether initially encrypting files, rekeying them, or decrypting them, the File System Agent must therefore:

- **Read** each block of file data to be transformed.
- **Transform** the block by encrypting, decrypting, or rekeying it.
- **Write** the transformed block, either to its original location, or to an alternate one.

Vormetric Transparent Encryption (VTE) protects data either at the file level or at the storage device level. VTE file-level protection consists of two main components:

- **Data Security Manager (**DSM**)**

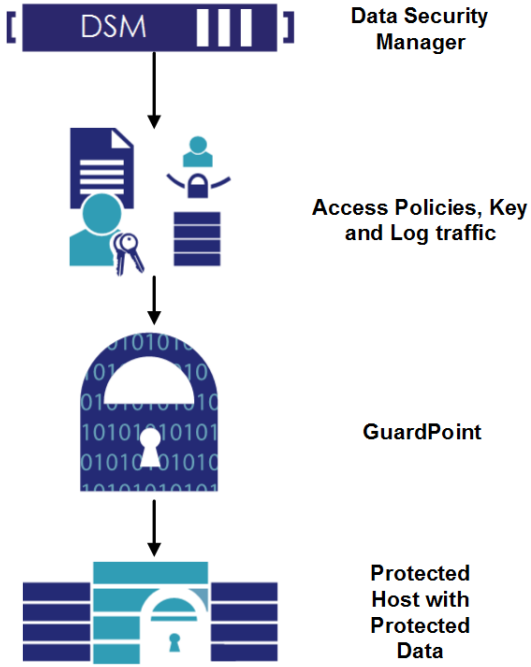
  An appliance that manages a database of the file sets protected by VTE **GuardPoints**, the **encryption keys** that protect them, **policies** that specify access rights and encryption protections that can be applied to GuardPoints. The DSM is also a central point for logging events related to accessing protected files.

- **File System Agents**

  Software components that run on hosts with protected file sets (called **protected hosts**). A File System Agent manages the files behind a GuardPoint by enforcing the policy associated with it, and communicates data access events to the DSM for logging.

A GuardPoint is usually associated with a Linux mount point or a Windows volume, but may also be associated with a directory sub-tree. A File System Agent sits between applications and the file system that hosts files within the GuardPoint. It intercepts every file access request, and enforces the access and encryption rules in the GuardPoint's policy.

Figure 1: VTE operation



VTE Policies

VTE policies consist of ordered lists of rules that specify:

- **Actors**

  Users, groups, and processes that are permitted to access protected data.

- **Actions**

  The actions available to authorized actors. For example create/delete, read/write, decrypt, modify permissions, and so on.

- **Files acted upon**

  Policy rules may apply to entire directories and mount points, or only to files named in a specific way (for example, .docx files may be encrypted and restricted to read-only access by designated users, while other files may be stored clear and read and written by anyone).

In addition, each VTE policy specifies an encryption key used to encrypt blocks of file data when applications write them and decrypt them when they are read.

VTE encryption is transparent to applications. This means that the File System Agent encrypts blocks of data as they are written, and decrypts data when they are read by authorized users and applications. This architecture separates administration of files from access to the data in them. Backup programs, for example, may be authorized to read files, but not view the data in them. Therefore, data can be backed up and taken off-site while remaining encrypted so that security is not breached.

Inherent Properties of Data Transformation

For large file sets (hundreds of gigabytes or more), bulk transformation is time-consuming. Managing transformation time is important, because file set content must be frozen (inaccessible to applications) throughout the transformation process. Once transformation starts, it must continue until complete, so transformation time determines the window of data unavailability. Two major components contribute to transformation time:

- **Number of blocks of file data**

  Because VTE must read, transform, and rewrite each block of file data, this component can be estimated by multiplying the number of file blocks to be transformed by the average read, transformation, and write time for a block.

- **Number of files**

  Because the File System Agent transforms data file by file, each file must be "looked up," opened, and closed during transformation, using underlying file system mechanisms. This typically requires multiple disk accesses. Therefore, file sets that consist of many small files, per-file overhead, can actually exceed file block transformation time.

Other factors, such as file system fragmentation, and load from concurrent applications, may also affect transformation time. Mainly, the number of blocks and number of files to be transformed are fundamental in that they cannot be reduced or eliminated.

Data Transformation Techniques

Two basic methods are available for initially encrypting and rekeying files protected by VTE:

- **Copy/Restore**

  Using the operating system file copy utility, a protected host administrator can copy unprotected files into a location protected by a VTE GuardPoint with a standard/production policy.

- **The VTE `dataxform` utility**

  Every File System Agent includes a utility program that can encrypt or transform protected files. The `dataxform` utility encrypts, rekeys, or decrypts data in place.

Both methods have advantages and limitations that make them suitable in different scenarios. These are discussed in the sections that follow.

In addition to encrypting data, you can also reverse the transformation. To **decrypt** protected files, copy them from a protected location to an unprotected one.

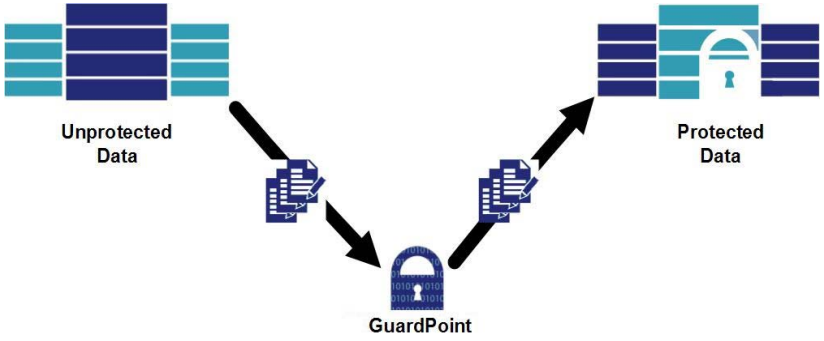
NOTE: VTE can also be configured to protect data at the disk level. For data protected in this way, only the copy transformation technique is available for encryption.

Properties of copy and restore transformation method

The copy method performs initial encryption, rekeying, and decryption by copying data from one directory, or GuardPoint, to another directory or GuardPoint.

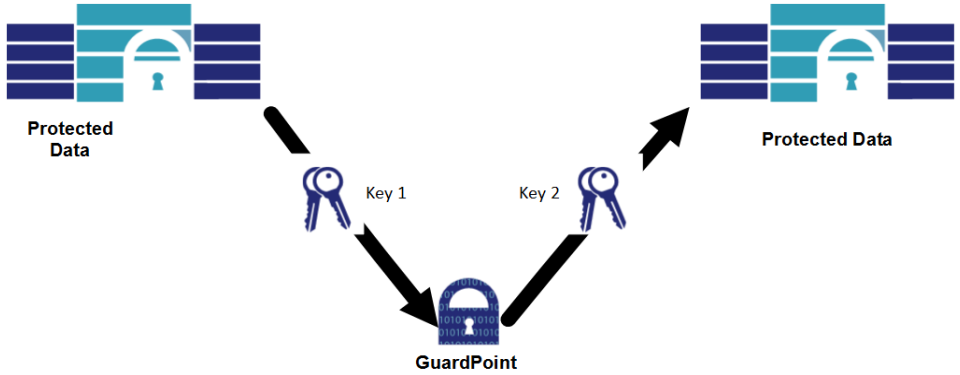
In the following figure, the administrator of the protected hosts encrypts a file set by copying it to a directory protected by a VTE GuardPoint with a standard policy. Encryption is transparent to the copy utility.

Figure 2: Initial Encryption



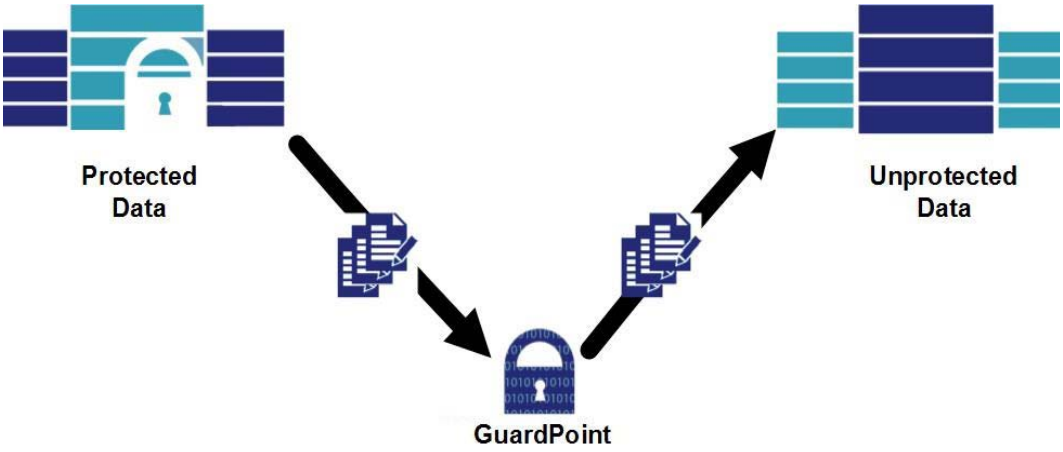
In the following figure, already-encrypted files protected by a VTE GuardPoint are rekeyed by copying them to a directory protected by another GuardPoint with a different encryption key. Both decryption and re-encryption are transparent to copy utilities.

Figure 3: Rekeying Protected Data



In the following figure, you can decrypt a protected file set by copying files from their protected location to unprotected directories. The File System Agent decrypts file blocks before delivery to the copy utility for rewriting.

Figure 4: Decrypting Data by Copying



NOTE: Exercise care here. If the governing policy does not authorize the copy utility user to access data, VTE delivers encrypted file blocks to it.

Encrypting and rekeying files by copying has two important advantages:

- **Simplicity**

  Once you have installed an Agent and GuardPoints are activated on a protected host, the protected host's administrator can encrypt, decrypt, or rekey file sets simply by copying them from one location to another. There are no procedures to learn, and no requirement to coordinate with the DSM Security Administrator. Data transformation is simply another routine administrative task.

- **Recoverability**

  If a copy-based transformation is interrupted, for example, by a power failure or system crash, the transformation resumes at or prior to the point of interruption. This is because all of the source files remain available and can be recopied, overwriting files at the destination that may have been only partially re-encrypted.

Offsetting these advantages are two limitations inherent to the copy method:

- **Storage resource consumption**

  Copying a file set requires that both source and destination files exist simultaneously. Storage capacity sufficient for both must be available during initial

encryption. For very large protected data sets, “extra” temporary storage may be a significant expense. However, a greater concern is likely to be the impact of moving production file sets as they are transformed. File data is unprotected while in the copy utility’s buffers.

- **Impact on operating procedures**

  Original and copied file sets have different path names and/or network addresses. After transformation, either both file sets must be renamed (the old path to a new name, and the new path to the old name), or applications must be adapted to process the transformed data set at the new directory. For a small data center with a few protected file sets, some combination of these options is usually practical. For data centers with hundreds of protected file sets, the administrative complexity and consequent chance of error make copying a complex option.

See “Initial Data Encryption” on page 27 and “Rekeying” on page 43 for detailed operational information on the copy method.

Restore transformation method

A variation of the copy method is to make a backup of the files for transformation and restore the backup to the destination location. This works because:

- Backing up data causes it to be read and decrypted.
- Restoring data causes it to be written (re-encrypting it with an alternative key).
- VTE protection is transparent to backup programs.

This technique also creates a backup of the data set. However, a disadvantage is the time required to copy data twice (once from the source location to backup, and once from backup to destination location).

These considerations suggest that copying data to transform it is more suitable for initial encryption (and final decryption), and less so for rekeying. Additionally, the simplicity of recovering an interrupted transformation makes the copy/restore method useful in situations where the probability of interruption during transformation is significant.

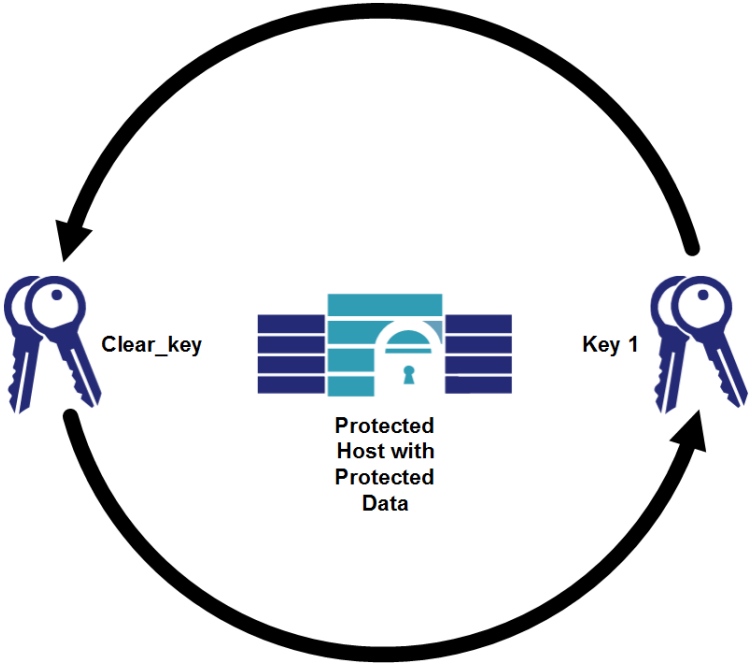
See “Restore encryption method” on page 28 for detailed operational information on the restore method.

Properties of the VTE `dataxform` Utility

The VTE `dataxform` utility transforms data-in-place and contains two components:

- User-mode that controls the overall operation
- Kernel-mode that transforms files block-by-block

Figure 5: Offline Rekey



Transforming data in place has two important advantages:

- **Minimal storage requirements**

  Because `dataxform` transforms files in place, where they reside, it does not require temporary file storage. However, the utility does need storage in which to create a list of files for transformation.

- **Security**

  The period of time that the data transformed by the `dataxform` utility appears in memory, outside the GuardPoint and therefore, unprotected, is shorter than with copying. This is significant for rekeying (compared to copying), which holds clear file data in memory between reading and rewriting. Moreover, `dataxform` requires coordination between the protected host and DSM Security Administrators, so that no one individual can subvert security during transformation.

Offsetting these advantages is the complexity of recovering from an interrupted `dataxform` run. Because `dataxform` transforms files in-place, data in a file undergoing transformation at the time of a failure may be only partly transformed. There is no way to determine which blocks have been transformed and which have not. These files must be recreated after the `dataxform` runs from a backup copy. The protected host administrator must determine (by examining `dataxform` logs) which files may have been incompletely transformed, delete them from the transformed file set, and recreate them by selective copying from a backup. The table below summarizes the strengths and weaknesses of the two file set transformation methods.

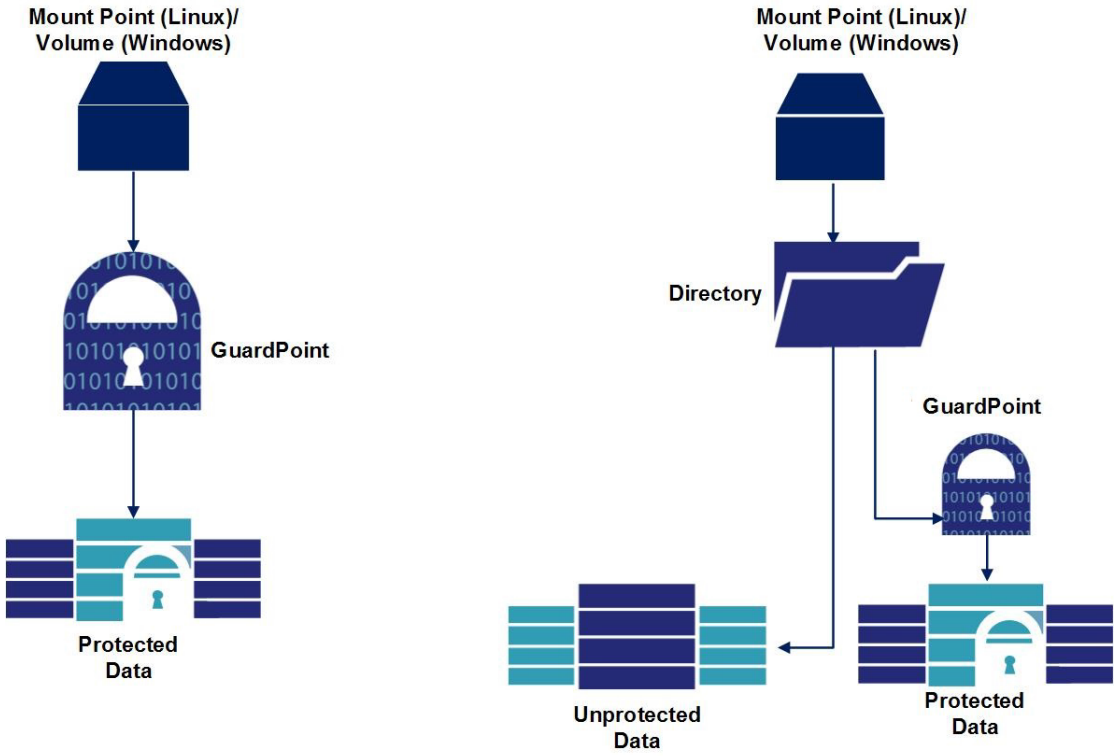
Issue	Copy Method	`dataxform` method
Temporary storage required	Equal to size of file set.	Sufficient to hold a list of path names of files in file set.
Security	File data is unprotected while in copy utility's buffers.	File data is never outside the VTE GuardPoint.
Initial encryption	Files can be copied directly from source directory to a VTE-protected directory.	Files must be in a protected location before transformation.
Operational impact	No access to files during transformation. Path names or operating procedures must be adjusted after transformation.	No access to files during transformation. No other impact on operating procedures.
Recoverability	Restart copy operation at, or prior to, point of failure.	Files undergoing transformation at point of failure must be discovered from dataxform logs and restored from backup.

VTE Protection Policies

The basic unit of VTE data protection policy application is the GuardPoint. GuardPoints are typically associated with file system mount points, but may also be associated with directory sub-trees.

NOTE: Nested mount points within a directory, or mount points protected by a GuardPoint, are also protected in Linux environments.

Figure 6: VTE GuardPoints



All files in the directory hierarchy, below a GuardPoint, are subject to the GuardPoint's policy, which consists of rules that specify:

- **Protected files**: Filenames or filename patterns (example: *.dat) to which the policy applies.

- **Authorized users**: User(s) group(s), and application(s) permitted to access the protected files.
- **Permissions**: Actions permitted to users (example: create/delete, read/write, rename, decrypt).

Policies also specify the name of an encryption algorithm and a key for encrypting protected files. For example, a policy might specify that all Excel workbooks protected by a GuardPoint be encrypted using an AES256 key called EXCEL-KEY. Additionally, only users in group 128 have access to the files. All other files that are not encrypted, are freely accessible to all users.

File System Agents use two types of policies:

- **Initial Data Transformation**

  Dataxform policies contain the elements listed above, plus a ***data transformation key***, used by the `dataxform` utility to rekey file data. Transformation policies contain strict access control rules that prevent application and user access to files during transformation. VTE only uses dataxform policies for the initial transformation. Afterwards, you replace it with a production policy.

  Dataxform operates on a per-GuardPoint basis. For *initial encryption*, the `dataxform` policy specifies a ***clear*** production key (meaning that the utility does not decrypt data because the data is unencrypted) and a new data transformation key to encrypt the data.

- **Production/Standard**

  Production policies contain the elements listed above. They protect data within GuardPoint(s) during day-to-day IT operations.

For *decryption*, the policy specifies a clear data transformation key (that is, the utility does not re-encrypt files as it rewrites them) and the current production key. A **rekeying transformation** policy specifies both a current production ("old") key and a transformation ("new") key.

The following tables show various policy components of a typical `dataxform` rekey policy.

Security Rule (Policy Rules)

Order	Resource	User	Process	Action	Effect	When	Browsing
1				key_op	Audit, Permit,		Yes

Order	Resource	User	Process	Action	Effect	When	Browsing
2				all_ops	Deny		Yes

Key Selection Rules (Production/Standard)

Order	Resource	Key
1		Original_Key

Data Transformation Rules (Key)

Order	Resource	Key
1		New_Key

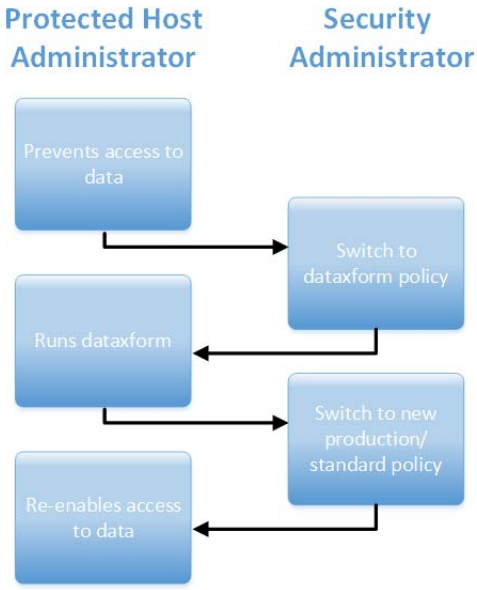
Overview of the `dataxform` Utility

The `dataxform` utility is installed on a protected host during File System Agent installation. The utility:

- Reads each file block-by-block
- Uses the production key to decrypt each block
- Re-encrypts it with the data transformation key
- Rewrites it to its original location.

The following figure illustrates that transforming data with `dataxform` requires collaboration between the Protected Host Administrator and the DSM Security Administrator.

Figure 7: Administrator Collaboration



1. The protected host administrator disables access to the files to be transformed, by stopping applications and/or databases that use them. They inform the DSM Security Administrator when they are inaccessible.
2. The DSM Security Administrator creates a `dataxform` policy with appropriate encryption key(s) and applies it to the GuardPoint.
3. The protected host administrator runs the `dataxform` utility on the GuardPoint directory with the appropriate parameters and options, and informs the DSM Security Administrator when the utility completes.
4. The DSM Security Administrator replaces the `dataxform` policy with a production/standard policy (or an initial test policy used to create the production policy). These policies use production keys that are the same as the encryption key used in the `dataxform` policy. After the switch, the DSM Security Administrator informs the protected host administrator.
5. The protected host administrator re-enables user access to the protected data.

Protected host and DSM Security Administrators must coordinate with each other to transform protected data sets using `dataxform`. While this makes it impossible for a single individual to subvert VTE security, close coordination can be difficult to arrange, particularly in large data centers with many protected hosts administered by different individuals. To partially relax this requirement without compromising security, `dataxform` execution can be automated. See "Automatic Data Transformation " on page 22.

In addition to rekeying protected files, you can use `dataxform` for the initial encryption and decryption of file sets. For initial encryption, the DSM Security Administrator specifies `clear_key` as the transformation policy's encryption key and a new encryption key as the production key. This instructs the utility not to decrypt files before "re-encrypting" them. Similarly, to decrypt protected data, the DSM Security Administrator specifies `clear_key`, as the transformation key and the existing encryption key as the production key, causing `dataxform` to decrypt, but bypass re-encryption.

The `dataxform` utility starts by creating a list of all files that may require transformation. To guarantee that the list remains correct until the transformation is completed, the admin must prevent all file access by including an "`all_ops`" "deny" rule in the policy. Without this rule, `dataxform` does not start.

When using `dataxform`, it is important to keep in mind the following issues:

- Ensure that the pre-transformation production policy, the `dataxform` policy, and post-transformation production policy, all specify the same files to be affected.
- Similarly, the `dataxform` policy must specify the same production key as the pre-transformation production policy. The post-transformation production policy must specify the same production key as the `dataxform` policy's transformation key.

NOTE: VTE does not cross-check key relationships. This is the responsibility of the VDA Security Administrator.

Multithreading in the `dataxform` utility

The `dataxform` utility is almost always I/O bound. You can reduce end-to-end run time by configuring the utility to transform multiple streams of data concurrently, in separate kernel threads. `dataxform` can be multi-threaded in two dimensions:

- **File concurrency**

`dataxform` can transform up to 32 files in concurrent execution threads. Each time a kernel thread finishes transforming a file, it informs the user component, which responds with a command to transform the next file in its work list. Number of threads is set with the `--thd` option.

- **File chunking**

  You can also configure the kernel component to divide individual files into chunks and transform up to 16 chunks concurrently. The chunk size defaults to 128 KB, but you can adjust it using the `--buf_size` option.

File concurrency is useful for transforming large numbers of files, but less-so with file sets that consist of a few large files. For the latter, chunking is typically more advantageous.

Concurrent transformation reduces run time, and therefore the period during which protected files are unavailable to applications. On the other hand, because files undergoing transformation at the moment of a system crash must be recovered from a backup, more active files means more time-consuming post-run recovery. Moreover, more concurrent transformation activity consumes more processing, memory, and I/O resources, which are unavailable to other applications running concurrently.

`dataxform` space requirements

Because it transforms data in place, `dataxform` must run to completion once it starts. If the utility does not run to completion, some files will have been completely transformed, some will not have been transformed, and some will only be partially transformed. Transformed files will be encrypted with the transformation key, not-yet-transformed ones with the pre-transformation production key, and those undergoing transformation will be in an indeterminate encryption state. To enable successful completion of an interrupted transformation, `dataxform` adopts two strategies:

- **Master file list**

  Before transforming the files in a set, the utility makes a disk-based list of path names. The list determines the order of transformation, and is also used to determine the restart point if transformation is interrupted. When `dataxform` finishes transforming a file set, it deletes the master file list, so only temporary storage for the list is required.

- **Status logging**

Each time `dataxform` finishes transforming a file, the utility records the status of the transformation in the disk-based status file. Status includes the path names of files being transformed at the time of recording. This enables `dataxform` to restart after interruption and recover incompletely transformed files from a backup.

For large directories (e.g., those containing 100,000 or more files) the size of the `dataxform` file list can run to tens of gigabytes (the size is largely determined by file path name lengths). By default, `dataxform` stores its master file list in the directory in which it writes log entries. Before running `dataxform`, the protected host administrator should ascertain that the file system containing the logging directory contains sufficient space to hold a list of full path names within the GuardPoint (see "Monitoring dataxform" on page 59). If this is not practical, the administrator can designate an alternate location for the list and status files as a `dataxform` command line option.

`dataxform` Execution Time

During transformation, a file set must remain static. Therefore, it is inaccessible to users and applications. Once started, transformation must complete before admins can permit applications access to the transformed files. This includes any restarts and manual recovery of incorrectly transformed files. In effect, the transformation process determines the duration of the outage. There are two elements to consider when choosing a window of time during which transforming a data set does not adversely affect the business function it supports:

- **Length of run**

  The run time, assuming that the run is problem-free.

- **Success of run**

  Maximizing the chance of success (and therefore minimizing the need for time-consuming manual recovery).

The following sections discuss these two considerations.

Length of Run

The `dataxform` utility includes a **dry run** capability that uses a combination of sampling and calculation to estimate the duration of a problem-free run against a given file set. A dry run can execute while data is online, however, this results in less accurate runtime estimates. It counts both the number of files in the set and the amount of data they contain. It also performs some sample transformations of

dummy files to estimate how long an actual transformation would run. The result of a dry run is an estimate of `dataxform` run time. In most cases, however, the estimate is conservative, provided that other system activity is minimal during the actual transformation. See "To estimate a dataxform runtime period" on page 50.

Success of Run

To maximize the chances of successful transformation, the protected host administrator should ensure that the required resources will be available during the run:

- **Storage space**

  See "dataxform space requirements " on page 16.

- **Kernel threads**

  The utility uses kernel threads for actual file data transformation. The protected host administrator can specify as many as 512 concurrent file and data chunk transformation threads. The admin must also ensure that sufficient kernel threads are pre-configured in the operating system (usually at system startup time) so that the specification can be met in the presence of other system activity occurring during transformation. See "Multithreading in the dataxform utility " on page 15.

- **Processing power and I/O bandwidth**

  You can maximize the power and bandwidth by limiting other system activity during transformation. I/O bandwidth, particularly disk accesses, is especially important for sets containing large numbers of files, because each file must be located and opened (both of which require disk accesses) in addition to having its data read and overwritten.

NOTE: Minimizing `dataxform` run time should be a priority because once the utility starts, it must transform the entire data set before users and applications can access it again.

`dataxform` and sparse files

Sparse files are files in which storage space is allocated in file block addresses, into which data is written. Sparse files exist in both Linux and Windows file systems. Most Linux file systems do not allocate space for file blocks until the blocks are actually written. Thus, if an application creates a file and writes the first and 1000th blocks, the second through 999th blocks are represented as a ***hole*** in the file

system's data structures. When an application reads file blocks that have never been written, the file system returns zeros. Application reads from holes are thus indistinguishable from reads of file blocks that actually contain zeros. When an application writes data to file block addresses in the midst of a hole, the file system allocates storage space for the data, subdividing the hole into two smaller ones if necessary.

In Windows, when a file is created with a specified size or when an application writes to a file block address greater than had previously been written, the file system allocates space for all blocks from the first to the highest-numbered. Windows applications must issue special I/O requests to create holes. Sparse files are therefore encountered less frequently in Windows.

The File System Agent and the `dataxform` utility cannot distinguish a file block that contains zeros from a hole—both return blocks containing zeros when read. When `dataxform` decrypts and re-encrypts such blocks and writes them back to their original locations, file systems allocate storage space for blocks that may previously have been holes. For large, mostly sparse files, this can result in run times and storage consumption that far exceed expectations based on pre-transformation file sizes. Therefore, `dataxform` provides administrators with two options for dealing with sparse files:

- **Recognize holes**

  A protected host administrator can configure `dataxform` to detect and bypass the processing of file blocks that contain all zeros. The result is that holes remain holes, and file blocks that contained zeros prior to transformation continue to contain zeroed after transformation. Application reads of either return decrypted zeroed, and applications' first writes cause the file system to allocate storage and write encrypted data to them. See the `--preserve_sparse_files` option in the "dataxform man page" on page 82.

- **Ignore holes**

  Alternatively, a protected host administrator can configure `dataxform` to ignore holes. The utility decrypts, re-encrypts, and rewrites all file blocks. Any file blocks that previously corresponded to holes have storage allocated for them, and thus, after transformation, files are no longer sparse. See the `--encrypt_sparse_file_holes` option in the "dataxform man page" on page 82.

In most cases, it is advantageous for the protected host administrator to configure `dataxform` to recognize holes, so that sparse files remain sparse. Failure to do so can result in longer than expected transformation times and post-transformation

file sets that consume significantly more storage space than prior to transformation.

`dataxform` and duplicate elimination

A similar phenomenon can occur with files that are deduplicated or compressed by the under-lying file system or volume manager. If deduplication or compression boundaries do not align with the file blocks that are the units in which VTE encrypts data, the size of a transformed file set may be greater or less than that of the same file set prior to transformation.

`dataxform` and Linked Files

When using `dataxform`, protected host administrators must be cognizant of the utility's treatment of linked files—files for which two or more directory entries point to a single data image. In general, the utility encrypts and rekeys linked files correctly, but the relationship of links to GuardPoints means that administrators must be aware of links and how the utility handles them prior to transformation. A link may be **hard**—it may be a directory entry that points to a file inode to which one or more other directory entries in the same file system also point. A link may be **soft**—it may represent a file whose data consists of the path name of another file in the same or a different file system.

The `dataxform` utility can detect that a directory entry is a hard or soft link. It transforms any file with multiple hard links to it when it first encounters any of the links to the file. Thereafter, it **skips** the already-transformed file, and creates a skipped log entry. This is not an error, but an indication that `dataxform` recognizes that it has already transformed the file's data. Soft links are simply skipped.

There are two situations with linked files in which data corruption can potentially result:

- **External links**

  Links in directories outside a GuardPoint, that point to files in directories within the GuardPoint.

- **Links to external files**

  Links in directories protected by a GuardPoint that point to files in directories outside the GuardPoint.

In these cases, the File System Agent does not have complete control over application access to file data. For example, in a GuardPoint that protects a directory sub-tree (rather than a mount point), if a hard link in a directory outside the GuardPoint points to a file within the GuardPoint, the File System Agent does not see every access to the file's data. If the file is opened through the external link and data is written to it, the GuardPoint does not intercept the writes, and no encryption occurs. If the file is later opened through the protected path, and the data written from outside is read, the File System Agent decrypts it, even though it was never encrypted. This situation does not occur with GuardPoints that protect entire file systems, because hard links can only refer to file data within the same file system name space as the files to which they point.

This problem does not occur with an external soft link, because a file opened through a soft link in a directory outside the GuardPoint is ultimately opened through its actual path, which lies within a protected directory.

Similarly, if a hard link in a protected directory refers to file data outside the GuardPoint, the `dataxform` kernel component opens it and transforms the data in it. If the file is subsequently opened through a path outside the GuardPoint, data is not decrypted as it is read, and therefore appears corrupt to applications. If applications access the file from outside the GuardPoint and write data to it, the GuardPoint intercepts subsequent reads through the link, and the File System Agent decrypts data that was never encrypted.

To assist protected host administrators in dealing with linked files, the `dataxform` utility includes a facility for listing the hard linked files within a GuardPoint. Administrators can analyze these lists to determine whether the links cross GuardPoint boundaries and therefore represent potential for operational errors and data corruption.

See "Checking for Hard-Link Files Inside the GuardPoint with dataxform" on page 57 for more information.

Running dataxform and VSS

Vormetric Data Security supports transparent decryption of Microsoft Volume Shadow Services (VSS) files. However, if you use VSS and you use `dataxform` to change the encryption keys, then you must make a VSS shadow copy before and after running `dataxform` to ensure that the latest encrypted data is backed up. VSS snapshot does not go through the file system, so it takes the encrypted on-disk data for backup. However, VSS restore does go through the file system, so

Vormetric Data Security decrypts the data transparently. To make the shadow copy, you must have storage space available equal to two times the size of the data you are copying.

You must keep the old encryption keys to decrypt VSS shadow copies. You also must unguard and guard with the old encryption key policies. If a particular file or directory is inside a GuardPoint, you must change the existing policy with the old key. You must also change the policy before accessing any VSS files, and then before a VSS restore.

Automatic Data Transformation

Using `dataxform` for initial encryption or rekeying of data is a two-party procedure requiring cooperation between the DSM Security Administrator and administrators of protected hosts. The DSM Security Administrator creates policies and applies them to GuardPoints before and after transformation. The protected host administrators disable access to protected file sets, run the `dataxform` utility, and re-enable file access after transformation. The two-party architecture preserves security by making it impossible for a single individual to subvert data protection.

In small data centers, DSM Security administrators and protected host administrators typically work closely together and have an understanding of each others' priorities and constraints. In larger organizations, organizational and physical distances between them often exist. Moreover, a DSM cluster often manages data security and key management for dozens, or hundreds, of protected hosts.

Simplifying `dataxform` Data Transformation

VTE can be configured to partially automate data transformation with `dataxform`, reducing the need for administrator coordination. The administrator of a protected host enables automatic transformation of a protected data set by creating a file named `dataxform_auto_config` in the GuardPoint's root directory. This file contains information used to verify version compatibility with the File System Agent, as well as some parameters to be input to **`dataxform`** (for example, the location of the disk space to be used to construct the utility's file list).

If a `dataxform_auto_config` file is present when the DSM Security Administrator activates a **`dataxform`** policy (one that contains both production

and transformation keys), the File System Agent in the protected system automatically starts **`dataxform`**. Conversely, the protected host administrator can disable automatic transformation by deleting the `dataxform_auto_config` file from a GuardPoint's root directory.

When `dataxform` execution completes (or aborts), it leaves behind status files that it uses to regulate subsequent executions. Whenever the `dataxform` starts, it looks for these files, and if it finds them, displays an informative message and exits without transforming any files. This prevents `dataxform` from running repeatedly. Prior to running `dataxform`, a protected host administrator must execute the utility's cleanup function to eliminate status files from previous runs (see "Cleaning up a previous dataxform session" on page 56). If a transformation fails, the protected host administrator must repair the problem, complete the transformation, and *then* execute the cleanup function (see "Recovering a Failed dataxform Session" on page 69).

Even with automatic data transformation, the DSM Security Administrator must monitor `dataxform` progress (for example, by observing the DSM log), and replace the GuardPoint's `dataxform` policy with a post-transformation production policy when the run completes. Protected host administrators remain responsible for blocking access to data (for example, stopping databases and applications or unmounting file systems) so applications do not have access to files. Finally, protected host administrators are responsible for re-enabling application access to files after transformation is complete and the DSM Security Administrator has replaced the `dataxform` policy with a post-transformation production policy.

To summarize, the DSM Security Administrator and protected host administrator interact during automatic data transformation as follows:

- ***Enable automation*** (protected host administrator)

  To enable automation, the protected host administrator creates a `dataxform_auto_config` file in the root directory protected by the GuardPoint. This is a one-time action. The `dataxform_auto_config` file needs to only be updated when parameters change, or deleted when the administrator wishes to disable automation.

- **Clean up from previous transformation** (protected host administrator)

  The protected host administrator executes the `dataxform` cleanup function (`--cleanup`) to enable transformation to begin automatically when a transformation policy is activated for the GuardPoint.

- **Disable access to data** (protected host administrator)

The protected host administrator disables access to data, and informs the DSM Security Administrator that it is safe to replace the GuardPoint's pre-transformation production policy with a `dataxform` policy. The time between disabling access and activation of the `dataxform` policy is part of the overall window of data unavailability.

- **Monitor** `dataxform` **progress** (DSM Security Administrator)

  The DSM Security Administrator monitors the progress of the utility, and when the run is complete, replaces the `dataxform` policy with a new post-transformation production policy. Once the post-transformation production policy has been activated, the DSM Security Administrator notifies the protected host administrator that it is safe to re-enable application access to the protected file set. The time between completion of the `dataxform` run and re-enabling of data access is part of the overall window of file unavailability. See "Monitoring dataxform" on page 59 for operational details.

Partial automation of data transformation reduces the number of interactions between protected host and DSM Security Administrators. Expect that, over time, VTE will evolve to reduce the interactions to those required to maintain the fundamental security precepts of the software.

See "To run automatic data transformation" on page 53 for detailed operational examples.

An Ounce of Prevention

The `dataxform` utility transforms files in place, overwriting file blocks as it transforms them. In-place transformation is advantageous, because it minimizes the temporary storage required to transform large file sets. However, if `dataxform` fails partway through, files being transformed at the moment of failure must be restored from backups after the run is complete and access to the data set has been re-enabled. This implies that a reliable, up-to-date backup copy of a protected file set is a necessary prerequisite to in-place transformation.

If a large file set must be fully backed up prior to running `dataxform`, backup time must be added to the estimated transformation time to calculate the window of unavailability. (Backup and transformation must be consecutive so that files do not change between the two.) For large data sets, backup time can be substantial. In no case, however, can one recommend bypassing the backup step. If files are worth protecting with VTE, they are presumably of significant value to the enterprise.

Therefore, every reasonable effort must be made to protect them against loss as well as theft.

Summary

Periodic rekeying of encrypted data is increasingly becoming a regulatory or policy necessity, particularly in the health care and financial fields. In addition, a sometimes-overlooked problem in deploying online data encryption is the initial encryption of legacy data sets. There are two basic techniques for both initial encrypting and rekeying:

- Copy data from its source to a destination protected by the desired encryption policy
- Encrypt or rekey data in place, overwriting it block by block

As data center operations grow more complex, and as file sets grow larger, copying becomes a less viable option. Running `dataxform` requires cooperation between the DSM Security Administrator and administrators of protected hosts. File sets must remain offline for the duration of a `dataxform` run, so the expected outage window must be estimated, and everything possible done beforehand to ensure that the run will succeed. The utility includes facilities for estimating run time, and for discovering potential problems, such as linked files, prior to running the software.

2

Initial Data Encryption

After installing and configuring your Vormetric Data Security Platform, one of the most common procedures is to encrypt the data you want to protect. This is called *initial data encryption*. There are two methods for encrypting data with the VTE platform. The first is by copying or restoring your clear data into a GuardPoint with a production/standard encryption policy. The second is by using the `dataxform` utility to encrypt the data in place. This chapter describes both of these methods in the following sections:

NOTE: Before using the procedures in this chapter read the Chapter 1, "Data Transformation Overview."

Data encryption overview

The first step in the data encryption process is to determine the optimal encryption method for your environment. There are three methods:

- **Copy**

  Clear data is encrypted by copying it into a GuardPoint with an encryption policy.

- **Restore**

  Clear data stored on a backup device is encrypted by restoring it to a GuardPoint with an encryption policy.

- **`dataxform`**

  Data is encrypted in-place using the `dataxform` command line utility.

The optimal method depends on the following:

1. Whether you are encrypting data on a block device or directory.
2. The amount of disk space you have.

3. Speed of your backup devices.

NOTE: Whichever method you select, backup your data before encrypting it.

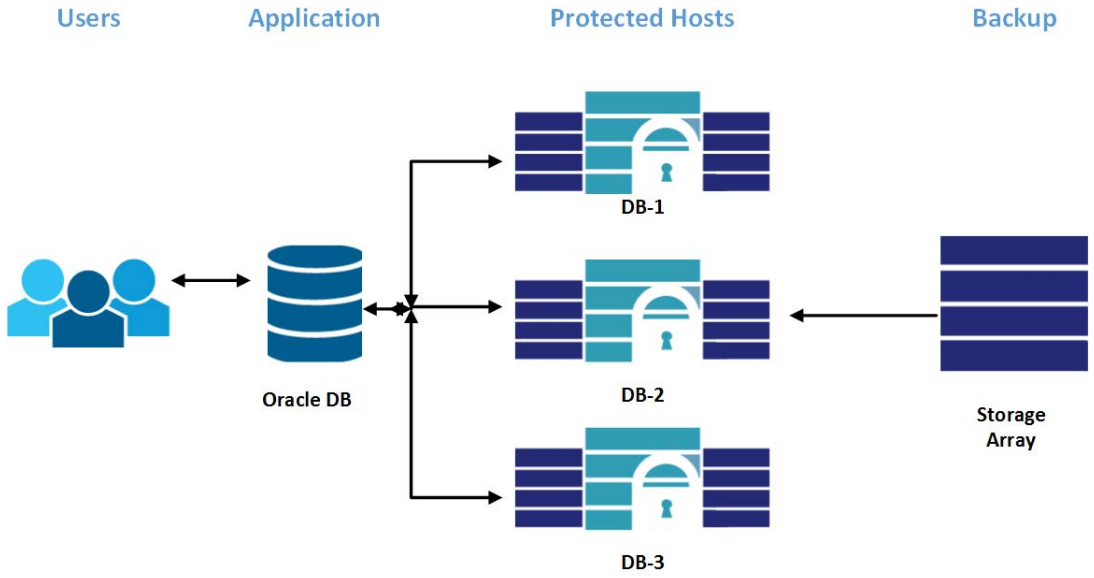
Restore encryption method

In this method, your sensitive data is backed up to a storage device. To encrypt the data:

1. Block access to the directory or device that will be encrypted.
2. Create a GuardPoint with an encryption policy on the directory or block device that will hold the protected data.
3. Restore the data from the backup device to the GuardPoint. As data is written into the GuardPoint, it is encrypted.
4. Replace the encryption policy with a production policy or the initial test policy.
5. Open access to the now-protected directory or block device.

In the following example, users access Oracle databases on the protected host.

Figure 8: Users Accessing a restored Oracle DB



To encrypt `\DB-3`:

1. Block user access to it.
2. Create an encryption GuardPoint on `\DB-3.....`
3. Restore the backup data from the backup media to `\DB-3`.
4. Restore access to the directory.

This method requires no extra disk space. The speed of this method depends on the speed of the restoration device.

Copy encryption method

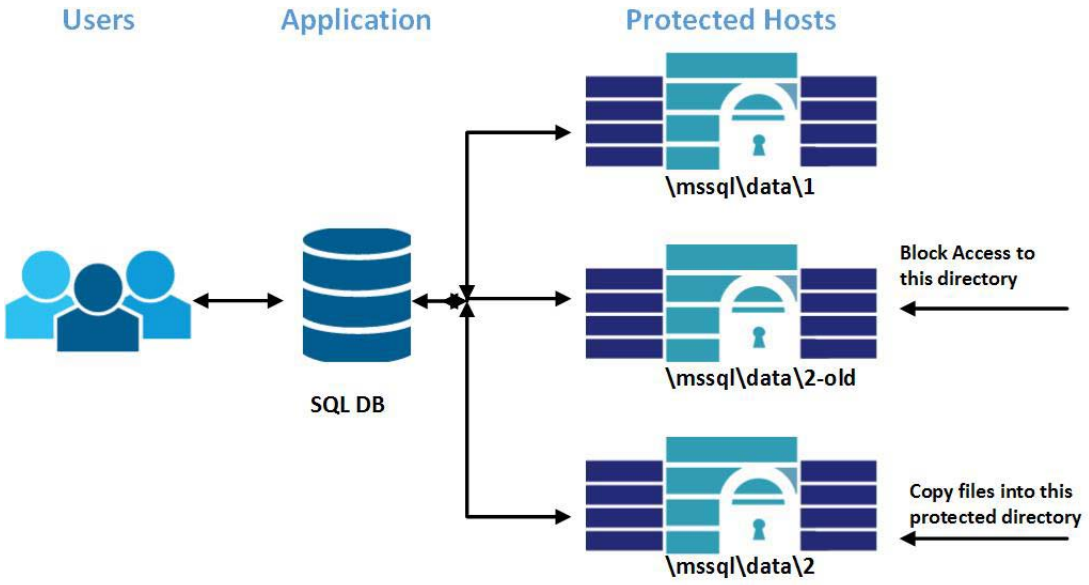
In this method, you encrypt clear data by copying it into a GuardPoint with an encryption policy. This method is generally faster than the restore encryption method. If the data you copy to the GuardPoint is on the same drive and volume as the GuardPoint, this method is comparable in speed to `dataxform`, which is about 2-4 GB per minute. If the data to be encrypted is accessed from a slower disk or a different volume, the encryption will be slightly slower.

Following is an example of the Copy Encryption process:

1. Block all access to the directory containing the data to be encrypted.
2. Rename that directory (example: from `\mssql\data\2` to `\mssql\data\2-OLD`).
3. Create a new directory for your sensitive data with the original directory path (`\mssql\data\2`) and block access to it.
4. Create a GuardPoint with an encryption policy on that directory.
5. Copy the sensitive data into the GuardPoint. Data in the GuardPoint is encrypted.
6. Replace the encryption policy with a production policy or the initial test policy.
7. Allow access to the new directory.

This method requires additional disk space at least as large as `\mssql\data\2-OLD`. The speed of the method depends on the speed of the copy.

Figure 9: Users using the copy method



dataxform encryption method

In this method, you encrypt data in place using the `dataxform` tool. In general, this method is the fastest. To generate an estimated time for encryption see "To estimate a dataxform runtime period" on page 50.

Following is an overview of the dataxform method:

1. Block all access to the directory containing the sensitive data.
2. Create a `dataxform` policy for the GuardPoint on this directory.
3. Run `dataxform` on the directory. After completion, the data in the GuardPoint is encrypted.
4. Remove the `dataxform` policy on the GuardPoint and replace it with a production policy.
5. Open access to the directory.

How to decide what method to use

Some general rules:

- If the data you are encrypting is in a block device, you must use the Copy or Restore encryption method.
- If you can copy the data into an encrypting GuardPoint on the same disk and same volume, the Copy method is as fast as the `dataxform` method.
- If you can copy the data into an encrypting GuardPoint on the same disk, but different volume, or SAN or NAS, you can use the Copy method but it is slightly slower.
- The Copy method also requires disk space that is twice the size of the data you are encrypting.
- If the data you are encrypting comes from a backup device that is not a disk, for example, a storage array, you can use the Restore method.

Using the Copy or Restore encryption method on file systems

This section describes how to use the Copy or Restore encryption method on file systems.

Warning! If you apply an encryption GuardPoint to a folder containing files, those files remain unencrypted. If you try to access those files, they are then encrypted. If you attempt a write, you can potentially corrupt

parts or the entire file. The only method to access those files in an unencrypted state is to disable or remove the GuardPoint.

Prerequisites

1. Verify that there is a good backup of the data to be encrypted. This step is vital.
2. Stop ALL access and services to the data to be encrypted. Make sure no processes, services, or users are currently accessing the data.
3. Make sure you have enough empty storage space to copy the data.
4. Make sure that you have a VTE production policy.

Encrypting GuardPoint data using the Copy or Restore encryption method

1. Log on to the Management Console as an administrator of type *All*, or as a Security Administrator with *Key* and *Policy* roles. Switch to the domain containing the host you want to protect.
2. Create an encryption policy to encrypt data copied into the GuardPoint.

   a. Create an encryption key.

   b. Using that encryption key, create a policy with a single rule:

   Action: all_ops **Effect**: permit apply_key
3. Under **Key Selection Rule** select the encryption key that you just created.
4. Create an empty directory for the protected data.
5. Apply the encryption policy to the empty directory. This is the new GuardPoint.
6. Block data access to the old operational directory.
7. Copy or restore the data from the old operational directory to the newly created GuardPoint. This data is encrypted.
8. Disable the encryption policy from the new GuardPoint and apply your production policy. Your data is now fully encrypted and you can redirect access to the GuardPoint.
9. Start application testing and inform application teams that systems are ready for use. Everything should work exactly as before except that now the data is encrypted. Monitor the situation with your users.

Using the Copy or Restore encryption method on block devices

The process for using the Copy or Restore encryption method on block devices and raw disks is much the same as with file systems.

Prerequisites

1. Verify that there is a good backup of the data.
2. The block device receiving the protected data must be new or clean as all existing data will be unusable.
3. Stop ALL services and access to the block device to be encrypted.
4. You will need a VTE production policy.

Information for encrypting block devices

- The Oracle DBA defines a disk group with `secvm` disks/devices. Then `secvm` communicates with the DSMDBA and updates the disk group information. Oracle ASM provides the mapping of secvm devices to physical disks/devices.
- All databases (table space) in a disk group must be encrypted. Do not mix encrypted and non-encrypted databases in a single disk group. Non-encrypted databases must be kept in separate disk groups that are not protected by a File System agent.
- You can configure the same GuardPoints in both the Edit Host window and the Edit Host **Group** window, and apply different policies to those GuardPoints. If you want to apply GuardPoints globally to a set of hosts, configure the GuardPoints in a host group. If you plan to configure the host in a host group, do not configure GuardPoints at the host level.
- Partitions are identified by their device name. Device names for partitions vary between platforms.

Encrypting data on a block device

1. Log in to the Management Console as an administrator of type *All*, or as a Security Administrator with *Key* and *Policy* roles.

2. Switch to the domain containing the host you want to protect.
3. Create an encryption policy with the following rule:

   Action: all_ops **Effect**: permit apply_key
4. Under **Key Selection Rule** select an appropriate encryption key.
5. Click **Hosts > Hosts** in the Management Console to open the *Hosts* window.
6. Click on the protected host name containing the block device where you will create the GuardPoints.
7. Click the **Guard FS** tab. The host's GuardPoints, if any, are display in the Protected Path.
8. Click **Guard** to open the **Guard File System** panel.
   - For **Policy**, choose an appropriate encryption policy. In the example, the name of the encryption policy is `basic-access-policy`.
   - For **Type**, choose the device type that fits the OS and Storage system.
     - For Windows choose **Raw or Block Device (Auto Guard)**.
     - For Linux/UNIX choose **Raw or Block Device (Auto Guard)**.
     - Select **Raw or Block Device (Manual Guard)** for raw devices that are to be manually guarded and unguarded in order to failover to a different node in a cluster. See "Automatic and Manual GuardPoints" on page 77 for details on this issue.
   - For **Path**, enter the GuardPoint folder. For example, */dev/sda1*
   - You can browse, but it should show **block** devices. Click a partition name to select it.

     Inactive partitions are displayed. Open partitions are not displayed, nor are currently guarded partitions.

NOTE: Third-party applications can open raw devices in obscure ways, and may cause the Remote File Browser to ignore and not display supposedly inactive devices. For example, inactive raw devices in the Oracle DBCA disk discovery path are not displayed in the Remote File Browser, even when the devices are not assigned to a disk group. If /dev/sd* is configured in the DBCA disk discovery path and DBCA is running, inactive /dev/sd* devices are not displayed in the browser. This is because the devices are kept open by the Oracle process. To get around this

problem, close dbca and open the browser again. The devices are free, displayed in the browser, and available for selection

9. Click **OK** to apply the policy to the GuardPoint. The *Edit Host* panel opens.

   A red status indicator means that the policy has not yet started to encrypt the data. Click **Refresh** until the *Status* turns green. This may take up to 30 seconds. The policy is then activated and the GuardPoint is protected.

10. Repeat this process for each block device that you want to protect.
11. Stop ALL services and access to the block device with the active data.
12. Using the appropriate method, copy or restore the data into the newly created devices so that VTE encrypts it.
13. Disable the encryption policy (if necessary) from the new GuardPoint and apply your production policy.
14. Start all services and restore access to the data that is now fully encrypted.

    Make sure that the applications and services access the newly created Vormetric protected hosts. Vormetric encrypted raw or block protected hosts are accessed using the directory:
    /dev/secvm/dev/xxxxx where *xxxxx* is the original device name.
15. Start application testing and inform application teams that systems are ready for use. Everything should work exactly as before; however, monitor the situation with your users.

Using dataxform to encrypt your data

The `dataxform` utility is an executable that encrypts data-in-place in a GuardPoint. This section describes how to use `dataxform` for initial data encryption. For more details on `dataxform` and its capabilities see "DataXform Reference" on page 49 and "Overview of the dataxform Utility " on page 13.

NOTE: `dataxform` does not work on block devices.

The process for using `dataxform` for initial data encryption is as follows:

- "Create dataxform policy."

- "Apply the dataxform policy to the GuardPoint." The GuardPoint contains the files to be encrypted.
- "Execute dataxform to start data encryption in the GuardPoint."
- "Remove the dataxform policy and apply production policy."Remove `dataxform` policy and apply the production policy.

Notes and Limitations

- For detailed `dataxform` information, see "dataxform man page" on page 82 and "Overview of the dataxform Utility " on page 13.
- Guarding and transforming linked files is potentially dangerous. See "Checking for Hard-Link Files Inside the GuardPoint with dataxform" on page 57.
- You can execute multiple `dataxform` sessions on the same host as long as each `dataxform` session is executed in a different GuardPoint. If you execute multiple instances of `dataxform` manually, the instances run in parallel. If you execute multiple instances of `dataxform` automatically (see "Automatic Data Transformation " on page 22 and "To run automatic data transformation" on page 53), the instances run consecutively.

NOTE: `dataxform` is optimized to run as fast as possible, tuning itself to the computer automatically. To run more than one instance simultaneously, you may need to reduce to number of execution threads allocated to each instance. See "dataxform man page" on page 82.

Prerequisites

1. If you use Microsoft Volume Shadow Services (VSS) files and use `dataxform` to change the encryption keys, then you must make a VSS shadow copy before and after running `dataxform`. See "Running dataxform and VSS" on page 21.
2. Verify that there is a good backup of the data to be encrypted. This step is vital.
3. Stop ALL services and access to the data to be encrypted. Make sure no processes, services, or users are currently accessing the data.Create a VTE production policy. (See the *DSM Administrator Guide*)

Create dataxform policy

The `dataxform` initial encryption policy is required when you run the `dataxform` executable on a directory. After `dataxform` encrypts the files, you remove the `dataxform` initial encryption policy and add a production policy to the GuardPoint.

1. Log in to the Management Console as a Security Administrator with *Key* and *Policy* roles or as an administrator of type *All*. Switch to the domain containing the host you want to protect.
2. Create an encryption key.
3. Create an initial encryption policy:
   a. Click **Policies > Manage Policies** to display the Policies window. The Policies window lists policies available to this domain.
   b. Click **Add Online Policy**. The Add Online Policy window opens.
   c. For Policy Type, select **Standard**.
   d. Enter a name and optional description for your policy. In this example, we used `dataxform1`.
   e. Click **Add** in the Security Rules panel. The Add Security Rule window opens.
   f. Select **Action**. The Select Action window opens.
   g. Select **key_op** and click **Selection Action**. The Add Security Rule window returns with **key_op** in the **Action** field.
   h. Select **Effect** to display the Select Effect window.
   i. Select **Permit** and **Apply Key**, then click **Select Effect**.
   j. Click **OK**. The Add Policy window opens with the Security Rule added. The Key Selections Rules panel and the Data Transformation Rules Panel also display.
   k. Click **Add** in the Key Selection Rules panel. The Add Key Rule window opens.
   l. Click **Key**. The Select Symmetric Key window opens.
   m. Select **clear_key** and click **Select Key**.
   n. The Add Key Rule window opens with `clear_key`. Click **OK**. The Add Policy window opens with the Security Rule added.
   o. Click **Add** in the Data Transformation Rules panel. The Add Key Rule window opens.
   p. Click **Key**. The Select Symmetric Key window opens.
   q. Select the key you created in Step 2.
   r. Click **Select Key**. The Add Key Rule window opens with the key name entered.

s. Click **OK**. The Add Online Policy window opens with the Data Transformation Rule added.

4. Click **OK**. The `dataxform` policy is added to the Policies *window*.

Apply the dataxform policy to the GuardPoint

Once you create the `dataxform` policy, you must apply it to the directory containing the data for encryption. When a policy is applied to a directory, the directory is called a **GuardPoint**.

In the following instructions, you apply the dataxform1 policy to a directory on one of your protected hosts. In the following steps, we will use an example directory called `/vipdata` to demonstrate the procedure.

1. Click **Hosts > Hosts** in the Management Console. The *Hosts* window opens.
2. Click on the protected host name in blue that contains the directory with the files to encrypt. The **Edit Host** screen opens.
3. Click the **Guard FS** tab. Existing GuardPoints display. Click **Guard**.
4. For **Policy**, choose **dataxform1**.
5. For ***Type***, select **Directory (Auto Guard)**.

   Select **Directory (Manual Guard)** for Linux/UNIX file system directories that must be manually guarded and unguarded in order to failover to a different node in a cluster. See "Automatic and Manual GuardPoints" on page 77 for details on this issue.
6. For **Path**, enter the directory for the files to be encrypted.

   Optionally, click **Browse** to browse and highlight the GuardPoint directory.

NOTE: Browse will not work if the host was registered with *One-way Communication*.

7. Click **OK** to apply the policy to the GuardPoint.

   The *Edit Host* panel opens. A red status indicator means that the policy hasn't taken effect.
8. Click **Refresh** until the *Status* turns green. This may take up to 30 seconds. The policy is now activated and the GuardPoint is ready to run the `dataxform` executable.

Execute dataxform to start data encryption in the GuardPoint

The following instructions are for running `dataxform` on a GuardPoint for the initial encryption.

1. Disable existing active GuardPoints before manually running `dataxform`. Encrypt the data. For each GuardPoint run `dataxform`. Example:

   ```
   # dataxform --rekey --print_stat --gp <directory>
   ```

   See "dataxform man page" on page 82 for all the features and options.

   Example:

   ```
   # dataxform --rekey --print_stat --preserve_modified_time --gp
   /opt/apps/dx2
   ```

   System Response

   ```
   Checking if data transform is supported for guardpoint
   /opt/apps/dx2
   Data transformation is supported on /opt/apps/dx2
   About to perform the requested data transform operation
   -- Be sure to back up your data
   -- Do not access files in the guard point during the transform
   process
   -- Please do not attempt to terminate the application


   Scan found 10005 files (273 KB) in 5 directories for guard point
   /opt/apps/dx2
   The current operation took 0 hours, 0 minutes and 1 seconds
   Transformed 10010 files (273 KB) of 10005 files (273 KB) for
   guard point /opt/apps/dx2
   The current operation took 0 hours, 0 minutes and 25 seconds
   Data transform skipped some files
   The file /opt/apps/dx2/hardlinkedfileLocal01 was skipped. It
   was an additional hard link
   The file /opt/apps/dx2/filenothere03 was skipped. It was a soft
   link
   The file /opt/apps/dx2/filenothere02 was skipped. It was a soft
   link
   The file /opt/apps/dx2/filenothere01 was skipped. It was a soft
   link
   The file /opt/apps/dx2/hardlinkedfileLocal02 was skipped. It
   was an additional hard link
   Number of additional hard links skipped: 2
   Number of soft links skipped: 3
   ```

```
Missing 1 references to hard link /opt/apps/dx2/hardlinkfile01
Missing 1 references to hard link /opt/apps/dx2/hardlinkfile02
Missing 1 references to hard link /opt/apps/dx2/hardlinkfile03
The data transform operation took 0 hours, 0 minutes and 25
seconds
Data transform for guard point /opt/apps/dx2 finished but 5
files were skipped
#
```

- View the `dataxform` run results in the local log, `/var/log/vormetric/vordxf _path_usr.log`, or in the Logs window.
- View the list of files that were not transformed in `/var/log/vormetric/dataxform_status_skip-_path.log`.

NOTE: Low-power systems can run out of memory while running `dataxform`. If entries like `"[VMD] [ERROR] [1933564] [DXF4328E] Kernel component gave unexpected status 4."` and `"[VMD] [ERROR] [3670108] [DXF4300E] Out of Memory"` are sent to the system messages file, lower the `--thd` parameter value. It takes longer to run, but `dataxform` uses less memory and completes successfully.

2. Read the `dataxform` command line messages as it encrypts files. Specifically note messages that list files or folders that are skipped and the reasons why. The `dataxform` log file also contains this information. Use it to identify failed transformations (see "Monitoring dataxform" on page 59.
3. If a `dataxform` fails during transformation, you can usually rerun it and it resumes transformation beginning with the next file.The risk is that all files that were in-progress during the transformation may not be completely transformed at the point of failure. In this case you will have to restore it from your backup and transform it again.

NOTE: If `dataxform` fails, do not clean up the GuardPoint or remove the `dataxform` status files. If you run `dataxform` again in the same GuardPoint, `dataxform` will use these files and resume operations where it left off.

4. After you have verified that the encryption is successful, clean up the `dataxform` session files before you run `dataxform` again on the same GuardPoint. See "Cleaning up a previous dataxform session" on page 56.

Remove the dataxform policy and apply production policy

After running `dataxform`, the data in the target folder is encrypted. The next step is to remove the `dataxform` policy and apply your production policy. If you do not have a production policy, the DSM *Administrator Guide* describes how to create one.

1. In the Management Console, click **Hosts >** ***Host Name*** **> Guard FS**.
2. Select your initial encryption policy and click **Unguard**.
3. Click **Guard** and apply the production/standard policy to the GuardPoint that you just encrypted.
   - For **Policy**, choose the production policy. In this example, the name of the production policy is `basic-access-policy`.
   - For **Type**, use **Directory (Auto Guard)**.

     Select **Directory (Manual Guard)** for Linux/UNIX file system directories that are to be manually guarded and unguarded in order to failover to a different node in a cluster. See "Automatic and Manual GuardPoints" on page 77 for details on this issue.
   - For **Path**, enter the GuardPoint folder. For example, */vipdata* for Linux/UNIX hosts or *c:\vipdata* for Windows hosts.
   - Optionally, click **Browse** to browse and highlight the GuardPoint directory.

NOTE: Browse does not work if the host was registered with *One-way Communication*.

4. Click **OK** to apply the production policy to the GuardPoint. The *Edit Host* panel opens.

   A red status indicator means that the policy has not taken effect. Click **Refresh** until the *Status* turns green. This may take up to 30 seconds. The policy is now activated and the GuardPoint is protected.
5. Start all services and restore access to the data that is now encrypted.
6. Inform application teams that systems are ready for use. Everything should work exactly as before; however, monitor the situation with your users.

3 Rekeying

This chapter describes how to rekey your existing GuardPoints. It consists of the following sections:

Rekeying Overview

Data encryption keys are the keys used to encrypt data in a GuardPoint. *Rekeying*, also called *key rotation*, is the process of changing the encryption key used to encrypt your GuardPoint data. Changing GuardPoint encryption keys increases security and is required in some organizations. Best practices covered in the National Institute of Standards and Technology (NIST) Special Publication 800-57 dictate that encryption keys should be rotated periodically to ensure the security of data from compromise. This security, however, comes at a cost – namely, downtime to perform the re-encryption of the data.

There are two methods for rekeying:

- **Using the `dataxform` utility**

  Data is encrypted in-place using the `dataxform` utility. This method is fast and easy, but requires total and exclusive access to the GuardPoint. All users and applications are blocked from accessing the data until `dataxform` is finished executing. See "Rekeying with dataxform" on page 44.

- **Manual copying**

  A GuardPoint is created on a new directory or device and guarded using a new encryption key. Then the data is copied from the original GuardPoint to the new GuardPoint. During the copy, data is decrypted with the original key and encrypted with the new key when it is placed in the new GuardPoint. This method does not require exclusive access to the original GuardPoint, but requires extra storage space of at least the amount of data to be copied. It also requires more steps to ensure that the newly re-encrypted data is both protected by a policy using the new key and known to users or applications that require access to the data. See "Rekeying using manual copy method" on page 47.

Rekeying with `dataxform`

Rekeying with `dataxform` requires that you change the current key of the production policy on your GuardPoint to a new key.

Notes and Limitations

- For detailed `dataxform` information, see "dataxform man page" on page 82 and "Overview of the dataxform Utility " on page 13.
- Guarding and transforming linked files is potentially dangerous. See "Checking for Hard-Link Files Inside the GuardPoint with dataxform" on page 57.
- If you use Microsoft Volume Shadow Services (VSS) files and you use `dataxform` to change the encryption keys, then you must make a VSS shadow copy before and after running `dataxform`. See "Running dataxform and VSS" on page 21.
- You can execute multiple `dataxform` sessions on the same host as long as each `dataxform` session is executed in a different GuardPoint. If you execute multiple instances of `dataxform` manually, the instances run in parallel. If you execute multiple instances of `dataxform` automatically (see "Automatic Data Transformation " on page 22 and "To run automatic data transformation" on page 53), the instances run consecutively.

NOTE: `dataxform` is optimized to run as fast as possible, tuning itself to the computer automatically. To run more than one instance at once, you may need to reduce to number of execution threads allocated to each instance. See "dataxform man page" on page 82.

- If you change the encryption key on particular production policy, and if another GuardPoint on another host uses the same production policy, then that GuardPoint's data will be unreadable because it still uses the old key. To avoid this:
  - Transform all of the data in all of the GuardPoints on all of the Hosts that use the same production policy with the new key
  - Change the name of the production policy used on the GuardPoint on which you ran `dataxform`. The policy will then only apply to that GuardPoint and not the other GuardPoints using the policy of the original name.

To rekey with `dataxform`

The following steps describe the process of transforming data from one encryption key to another encryption key using the `dataxform` utility.

1. If `dataxform` was run on this GuardPoint previously, you must clean up those `dataxform` sessions. See "Cleaning up a previous dataxform session" on page 56.
2. Log into the DSM Management Console.
3. In the **Keys** menu, create a new key for re-encrypting data.
4. Create a `dataxform` policy that specifies the following:
   a. **Action**: key_op
   b. **Effect**: apply_key, permit
   c. **Key Selection Rules** tab key: The original key currently in use.
   d. **Data Transformation Rules** tab key: The new key.
5. Block all access to data in the GuardPoint that is to be re-encrypted.
6. In the Management Console, click the **Hosts** tab, select the ***<hostname>***, and click the **Guard FS** tab.
7. Disable the production policy on the GuardPoint.
8. Add a new GuardPoint to the host with the same directory as the original GuardPoint, applying the `dataxform` policy to the GuardPoint.

NOTE: At this point, all access to the GuardPoint is denied except for `dataxform`.

9. From the command line on the protected host, run `dataxform` as `root` or admin user with at least the `--rekey` and `--gp` options.

   Example

   ```
   # dataxform --rekey --print_stat --preserve_modified_time --gp
   /opt/apps/dx2
   ```

   System Response

   ```
   Checking if data transform is supported for guard point
   /opt/apps/dx2
   Data transformation is supported on /opt/apps/dx2
   About to perform the requested data transform operation
   -- Be sure to back up your data
   -- Do not access files in the guard point during the transform
   ```

```
process
-- Please do not attempt to terminate the application
Scan found 10005 files (273 KB) in 5 directories for guard point
/opt/apps/dx2
The current operation took 0 hours, 0 minutes and 1 seconds
Transformed 10010 files (273 KB) of 10005 files (273 KB) for
guard point /opt/apps/dx2
The current operation took 0 hours, 0 minutes and 25 seconds
Data transform skipped some files
The file /opt/apps/dx2/hardlinkedfileLocal01 was skipped. It
was an additional hard link
The file /opt/apps/dx2/filenothere03 was skipped. It was a soft
link
The file /opt/apps/dx2/filenothere02 was skipped. It was a soft
link
The file /opt/apps/dx2/filenothere01 was skipped. It was a soft
link
The file /opt/apps/dx2/hardlinkedfileLocal02 was skipped. It
was an additional hard link
Number of additional hard links skipped: 2
Number of soft links skipped: 3
Missing 1 references to hard link /opt/apps/dx2/hardlinkfile01
Missing 1 references to hard link /opt/apps/dx2/hardlinkfile02
Missing 1 references to hard link /opt/apps/dx2/hardlinkfile03
The data transform operation took 0 hours, 0 minutes and 25
seconds
Data transform for guard point /opt/apps/dx2 finished but 5
files were skipped
#
```

- View the `dataxform` run results in the local log, `/var/log/vormetric/vordxf _path_usr.log`, or in the *Logs* window.
- View the list of files that were not transformed in `/var/log/vormetric/dataxform_status_skip-_path.log`.

NOTE: If `dataxform` fails, do not clean up the GuardPoint or remove the `dataxform` status files. If you run `dataxform` again in the same GuardPoint, `dataxform` will use these files to resume operation where it had left off.

NOTE: Low-power systems can run out of memory while running `dataxform`. If entries like `"[VMD] [ERROR] [1933564] [DXF4328E] Kernel component`

`gave unexpected status 4." and "[VMD] [ERROR] [3670108] [DXF4300E] Out of Memory"` are sent to the system messages file, lower the `--thd` parameter value. It will take longer to run, but `dataxform` will use less memory and complete successfully.

10. On the production policy, change the key to use the new key instead of the original key.
11. Delete the transformation policy and re-enable the production policy that now has the new key.
12. Verify proper access to the data.

Rekeying using manual copy method

This section describes how to rekey your GuardPoint using the copy method.

- If you change the encryption key on a production policy, and if another GuardPoint on another host uses the same production policy, then that GuardPoint's data becomes unreadable because the data is still encrypted/decrypted with the old key. To avoid this:
  - Transform all of the data in all of the GuardPoints on all of the Hosts that use the same production policy with the new key
  - Change the name of the production policy used on the GuardPoint on which you ran `dataxform`. The policy will then only apply to that GuardPoint and not the other GuardPoints using the policy of the original name.

1. Identify a location where VTE can create a GuardPoint and has enough space to hold the data for rekeying.
2. Log into the Management Console.
3. In the **Keys** menu, create a new key for re-encrypting data.
4. Create a *transformation policy* that specifies the following:

   Action: all_op

   Effect: apply_key, permit

   Key Selection Rules key: The new key
5. On the Management Console, click **Hosts** > ***hostname*** > **Guard FS**

6. Add a new GuardPoint that uses the new production policy on the protected host location you identified in Step #1.
7. Copy or move the data from the original directory to the new GuardPoint.

   The data in the new directory is rekeyed.

NOTE: Use of a multi-threaded copy command, like the `xcopy` (Windows only), is highly recommended to minimize the transfer time.

Direct all applications and users to use the new data location, or change the name of the new GuardPoint directory to the name of the original directory.

1. If you direct all applications and users to use the new data location, make sure that they use the production policy that contains the new encryption key.
2. If you change the name of the new GuardPoint directory to the name of the original directory use the following instructions:

   a. In the Management Console, disable the original GuardPoint.

   b. Unguard the newly created GuardPoint with the rekeyed data.

   c. On the protected host, rename the original directory to a temporary name.

   d. Rename the newly created directory to the original directory name.

   e. In the Management Console, modify the original policy to use the newly created key.

   f. Enable the original GuardPoint with the original policy. This puts the original policy in effect with the new key on the newly transformed data.
3. Verify that the rekeyed data is accessible to users.

DataXform Reference

A

This appendix consists of the following sections:

Common dataxform examples

To display dataxform version information

```
# dataxform --version
```

System Response

```
Build version: 5.2.2
Build ID: 850
Build Date: 2014-121-22 09:22:38 (PST)
```

To verify that a directory guarded with a rekey policy can be rekeyed with dataxform

In the following example, the first GuardPoints have a standard policy. In the second, the GuardPoint has a rekey policy:

```
# dataxform --rekey_supported --gp /opt/apps/apps1/doc
```

System Response

```
Checking if data transform is supported for guard point
/opt/apps/apps1/doc
The kernel component doesn't support data transform on
/opt/apps/apps1/doc
Verify this is a guard point with valid data transformation
policy, and check the system log files for any other problems.
It may be due to one or more of following reasons; 1.policy has
no valid key rule(s), and/or 2. policy has no key_op rule,
and/or 3. policy has valid permit rule(s), and/or 4. policy
rule that contains key_op in the action field also specifies
other actions.
```

```
# dataxform --rekey_supported --gp /opt/apps/dx2
```

System Response

```
Checking if data transform is supported for guard point
/opt/apps/dx2
Data transformation is supported on /opt/apps/dx2
```

NOTE: You can also get the message "`not a guard point or there is no data transformation rule`" when an administrator is inside the GuardPoint or accessing files in the GuardPoint. Check that no one is in the GuardPoint and that a rekey policy is applied to the GuardPoint. If a GuardPoint does not qualify for rekeying, check that a key is configured in the *Data Transformation Rules* tab of the assigned policy in the Management Console.

To estimate a dataxform runtime period

Rekeying the files in a GuardPoint can take a long time if there are thousands of files to be rekeyed. The `--deep_scan` argument on the `dataxform` command simulates and estimate how long a `dataxform` session will take on a specified GuardPoint. Enter all the arguments that you would normally use for the `dataxform` session, and assign the rekey policy to the GuardPoint, so `dataxform` accurately simulates the actual rekey process.

NOTE: `--deep_scan` is CPU and I/O intensive. Expect a drop in system performance while running `--deep_scan`. It can take a long time to complete--enough so that for very small file systems, you are better off running `dataxform`

directly rather than trying to estimate how long it will take with the `--deep_scan` option.

The following shows a simple `dataxform` command line session with the `--deep_scan` argument.

```
# dataxform --deep_scan --gp /opt/apps/dx4
```

System Response

```
Checking if data transform is supported for guard point
/opt/apps/dx4
Data transformation is supported on /opt/apps/dx4
About to perform a deep scan of the guard point
-- To enable consistent results, do not access the guard point
during
   the scan
-- Transformation simulations will be performed, which may take
some time
Do you wish to continue (y/n)?y


Status information for directories in guard point
Directory /opt/apps/dx4/ab_dir/ contains 40 files (1 KB)
Directory /opt/apps/dx4/ac_dir/ contains 40 files (1 KB)
Directory /opt/apps/dx4/aa_dir/ contains 40 files (1 KB)
Directory /opt/apps/dx4/ad_dir/ contains 37 files (1 KB)
Directory /opt/apps/dx4/ contains 46 files (915 bytes)
Scan found 203 files (5 KB) in 5 directories for guard point
/opt/apps/dx4
Estimated data transformation time for /opt/apps/dx4 is 0h 0m
6s
-- This is an estimated time using actual data
   transformations on test files
-- The actual transformation time may be more or less than this
estimate
#
```

The second line of the `--deep_scan` output in the example above is "`Data transformation is supported on` *gp*". This line is displayed when the directory being scanned is an active GuardPoint with a rekey policy. This line is not displayed when the directory being scanned is a regular directory, a disabled GuardPoint, or an active GuardPoint with a regular, non-rekey policy.

The `--deep_scan` results are also echoed to `/var/log/vormetric/vordxf-`*`_path_usr`*`.log` and forwarded to the *Logs* windows.

To manually run dataxform on a specific set of files

Use the following procedure to manually execute `dataxform` on a specific set of files in a GuardPoint.

1. Back up the data in the GuardPoint.
2. If specific files are to be encrypted, create a file list.

   A file list is a text file that consists of the full path name of each file to be transformed. Enter one file path per line. If a file list is not specified, `dataxform` will rekey all the files in the GuardPoint.
3. Log on to the Management Console as an administrator of type Security Administrator with `Host` role permissions or type All.

NOTE: Existing active GuardPoints must be disabled before running a manual data transformation.

4. For an existing GuardPoint, disable it. For new GuardPoints, go to step 5.

   a. Open the *Guard FS* tab of the host with the GuardPoint to be transformed. The applied policies and GuardPoints of the host are displayed.

   b. Disable the GuardPoint that is currently in effect. Select the **Select** check box for the GuardPoint and click **Disable**.

   c. Confirm that the GuardPoint is disabled:

   - For Linux and UNIX systems: execute the `secfsd -status guard` command repeatedly until the GuardPoint is no longer displayed.
   - For Windows systems: on the task bar, right-click the Vormetric Tray Icon and click **View > File System > GuardPoints** until the GuardPoint is no longer displayed.
5. Create a `dataxform` *policy* and apply it to the now disabled or newly created GuardPoint. The `dataxform` policy specifies the following:

   Action: key_op
   Effect: apply_key, permit
   Key Selection Rules key: The original key currently in use. Use clear_key if unencrypted.
   Data Transformation Rules key: The new key. Use clear_key if decrypting.
6. Confirm that the GuardPoint is re-enabled:

- For Linux and UNIX systems: execute the `secfsd -status guard` command repeatedly until the GuardPoint is displayed.
- For Windows systems: On the task bar, right-click the Vormetric Tray Icon and click View > File System > GuardPoints until the GuardPoint is displayed.

7. Execute the `dataxform` command with the desired options on the host system. For example:

   `#dataxform --rekey_list --file_list dx_fileList.txt`

System Response

`--gp /home/apps/apps1/data --dir_recovery /root`

`--dir_recovery` allows you to specify where `dataxform` status files are placed.

8. (Optional) Monitor `dataxform` progress on the host system.

   `# tail -f /var/log/vormetric/vordxf_`*`path_usr`*`.log`

9. Wait until `dataxform` completes.
10. Disable or delete the `dataxform` policy and replace with a production policy. Reboot the host if you cannot disable or delete the rekey policy

Caution: Do not apply a policy that is configured for encryption to a directory that contains unencrypted files because, when `apply_key` is configured, the unencrypted files are encrypted when they are accessed. The data will be unusable if read and corrupted if saved.

To run automatic data transformation

You can automatically transform your GuardPoint data by creating a `dataxform` policy and placing a file, `dataxform_auto_config`, in the top-level directory of the GuardPoint.

The `dataxform_auto_config` file consists of one or two lines. The first line is mandatory. It specifies an internally used version number. Currently, the internal version number is `1`. Set the first line to "`version=1`". The second line is optional. The second line lists additional `dataxform` parameters. By default, `dataxform` executes the `--rekey` and `--gp` options. These options rekey all the files in the GuardPoint. Do not enter the `--rekey_list`, `--file_list`, or `--gp` options in

the `dataxform_auto_config` file. An example `dataxform_auto_config` file is shown below:

```
version=1
--thd 4 --preserve_modified_time
```

Automatic data transformation notes and limitations

- (Windows only) UNIX and Windows users can run both manual and automatic `dataxform`. However, after Windows users run automatic `dataxform`, the folder for the GuardPoint remains in a “busy” state and it cannot be unguarded. Windows users must reboot the system after performing automatic `dataxform` to reset the state of the GuardPoint and unguard the folder.
- (Windows only) Do not run automatic `dataxform` in GuardPoints that are on network drives. Automatic `dataxform` runs as the local “system” account. This account operates only on the file system and cannot operate across the network. To transform files on a network drive, log on to the system as an administrator with privileges to access the network drive and to modify the GuardPoint to be transformed, and then run `dataxform` manually.
- Low-power systems can run out of memory while running `dataxform`. If entries like "`[VMD] [ERROR] [1933564] [DXF4328E] Kernel component gave unexpected status 4.`" and "`[VMD] [ERROR] [3670108] [DXF4300E] Out of Memory`" are sent to the system messages file, lower the `--thd` parameter value. It will take longer to run, but `dataxform` will use less memory and should complete successfully.
- *Automatic* `dataxform` processes directories and files according to the native sort order of the system. Automatic `dataxform` is aware of the files currently being processed. If the automatic `dataxform` process is interrupted, you can restart it and it will resume from roughly where it had left off. It will resume within a range of files generally equal to the number of open threads that were running, using the files listed in `./dataxform_status-`*`_gp`* and `./dataxform_status-alt-`*`_gp`*, and based on the system sort order. We strongly recommend that no one access the GuardPoint during data transformation because depending where `dataxform` is in the list of directories and files to process, the directories and files that you create can be skipped, and changes that you make to the files can go unnoticed. You will then have to manually transform the new or modified files. Additionally, if you do not configure manual `dataxform` properly, it is easy to rekey a file twice, thus corrupting that file. Automatic `dataxform`, on the other hand, is easier to recover.

To run automatic dataxform

1. Back up and block all access to the GuardPoint.
2. Create a `dataxform_auto_config` file.
3. Log on to the Management Console as an administrator of type Security Administrator with `Host` role permissions or type All.
4. Open the *Guard FS* tab of the host with the GuardPoint to be transformed. The applied policies and GuardPoints of the host are displayed.
5. Disable the GuardPoint that is currently in effect. Select the **Select** option for the GuardPoint. Click **Disable**.
6. (Optional) Enter the `df` command on the host system repeatedly until the secfs mount for the GuardPoint is no longer displayed, or execute the "`secfsd -status guard`" command repeatedly until the GuardPoint is no longer displayed.
7. Copy the `dataxform_auto_config` file into the GuardPoint. Data transformation should start within seconds.
8. Apply the `dataxform` policy to the now disabled GuardPoint.

   (Optional) Execute the "`secfsd -status guard`" command repeatedly on the host system until the GuardPoint and rekey policy are displayed. You can also keep clicking the **Refresh** button in the *Edit Host* window, *Guard FS* tab, until the green status ball is displayed.
9. (Optional) Monitor `dataxform` progress on the host system.

   `# tail -f /var/log/vormetric/vordxf_`*`path_usr`*`.log`
10. Check log files to verify successful `dataxform` completion.

    The `/var/log/vormetric/vordxf_`*`path_usr`*`.log` file lists the success or failure of `dataxform`, the files that were affected, and the actions taken. Refer to "Using dataxform_status* Files" on page 62 for details.

    Check the rekey status in the *Logs* window.
11. Disable or delete the `dataxform` policy. Reboot the host if you cannot disable or delete the rekey policy.
12. Delete the `dataxform_auto_config` file from the GuardPoint.

    If you do not delete the `dataxform_auto_config` file, the next time you apply a rekey policy to the GuardPoint, data transformation will begin immediately. It is better to copy the file into the GuardPoint when you are ready.
13. Apply a production policy to the GuardPoint. If the `dataxform` policy used an encryption key, be sure to use the same key in the production policy.

Caution: Do not apply a policy that is configured for encryption to a directory that contains unencrypted files because, when `apply_key` is configured, the unencrypted files are encrypted when they are accessed. The data will be unusable if read and corrupted if saved.

Cleaning up a previous dataxform session

When `dataxform` execution completes (or aborts), it leaves behind status files that it uses to regulate subsequent executions. Whenever the `dataxform` starts, it looks for these files and if it finds them, either resumes running where it left off, or displays an informative message and exits without transforming any files. The exiting prevents `dataxform` from running repeatedly. Prior to running `dataxform`, a protected host administrator must execute the utility's "cleanup" function to eliminate status files from previous runs. If a transformation fails, the protected host administrator must repair the problem, complete transformation, and *then* execute the cleanup function.

Warning! NEVER run a cleanup operation after a partial transformation, as you will be unable to resume transforming the remaining files.

When `dataxform` is run, it checks the status in the `dataxform_status-`*gp* file. If the status is "done", `dataxform` exits with "`data transform status for` ***GuardPoint_dir*`: previous attempt completed`" error message. Remove the status file and other configuration files with the `--cleanup` option.

The command syntax for cleaning a previous `dataxform` session is:

```
# dataxform --cleanup --gp gp
```

For example:

```
# dataxform --cleanup --nq --gp /opt/apps/dx2
```

System Response

```
About to remove the data transformation status files
#
```

The `--cleanup` operation does not return a message indicating a successful completion. If there are no errors, the cleanup operation completed successfully, and you can now run `dataxform` on the GuardPoint.

Checking for Hard-Link Files Inside the GuardPoint with dataxform

Guarding and transforming linked files is potentially dangerous. Links require special consideration. Depending on how policies and keys are applied to both the link file/directory and the source file/directory, there is the potential for policy conflicts and key mismatches. A key mismatch will corrupt data files when they are saved. See also "**dataxform** and Linked Files " on page 20 for more information.

With regard to links:

- There is only one instance of a file; where, with links, it can have multiple names. The instance is transformed when the first name for it is encountered. Subsequent names are recognized as aliases of the already transformed instance and are ignored.
- Never make a links to sources outside of the GuardPoint. If a file in a GuardPoint is linked to a file in another GuardPoint, and encryption is applied, cross-changes to the same file from different GuardPoints will use different keys and corrupt files.
- If a file in a GuardPoint is a link to a file that is not in any GuardPoint, then only standard operating system access controls are applied to the file outside the GuardPoint. That means that root, and possibly others, can access and modify the file instance without an encryption key and without restriction. Only the file instance in the GuardPoint should be accessed, otherwise the key will not be used to decrypt/encrypt the instance. If the instance is changed and saved outside of the GuardPoint, it will be corrupted.
- Security rules for link files inside the GuardPoint must be intelligently written to anticipate how files are accessed and how keys are applied. For example, you can create a file named `a.foo` and create a link to it named `a.bar`. If you write a security rule that applies one key to `*.foo` and another key to `*.bar`, the key applied depends upon how the file is accessed. Again, this is an opportunity for cross-changes and file corruption.

A GuardPoint should be inactive when you transform its contents. Make sure that linked files are also inactive.

The `--check_links` example below detected links. It found three hard links to files outside the GuardPoint and two to files inside the GuardPoint. The `dataxform` output includes the full path names of the hard-link files detected in the GuardPoint.

```
# dataxform --check_links --gp /opt/apps/dx2
```

System Response

```
Status information for directories in guard point
Directory /opt/apps/dx2/ab_dir/ contains 2002 files (57 KB)
Directory /opt/apps/dx2/ac_dir/ contains 2000 files (57 KB)
Directory /opt/apps/dx2/ad_dir/ contains 2000 files (57 KB)
Directory /opt/apps/dx2/aa_dir/ contains 2000 files (57 KB)
Directory /opt/apps/dx2/ contains 2003 files (43 KB)
Scan found 10005 files (273 KB) in 5 directories for guard
point /opt/apps/dx2
Scanning for hard links in directory /opt/apps/dx2
Found 1 references to hard link but expected 2
Hardlink reference 1 /opt/apps/dx2/hardlinkfile01
Found 1 references to hard link but expected 2
Hardlink reference 1 /opt/apps/dx2/hardlinkfile02
Found 1 references to hard link but expected 2
Hardlink reference 1 /opt/apps/dx2/hardlinkfile03
Found 2 references to hard link
Hardlink reference 1 /opt/apps/dx2/hardlinkedfileLocal02
Hardlink reference 2 /opt/apps/dx2/ab_dir/linkedfile02
Found 2 references to hard link
Hardlink reference 1 /opt/apps/dx2/hardlinkedfileLocal01
Hardlink reference 2 /opt/apps/dx2/ab_dir/linkedfile01
#
```

Note the lines that read:

```
Found 1 references to hard link but expected 2
```

Because `dataxform` “expected” more links than it found, a likely culprit is that the file inside the GuardPoint is linked to a file outside the GuardPoint. We strongly recommend that links do not go outside the GuardPoint because policy and key application can become unpredictable and corrupt files.

Also, note the lines that read:

```
Found 2 references to hard link
```

These lines indicate that the link file and the source file both reside inside the GuardPoint.

Monitoring dataxform

Dataxform activity is recorded in three places:

- The *Message Logs* window
- `/var/log/vormetric/vordxf_`*`path_usr`*`.log`
- `/var/log/vormetric/dataxform_status-`*`gp`* and `dataxform_status_alt-`*`gp`* files

Using the Message Log window

During a rekey operation, the DSM logs both the current encryption key and the new encryption key. Transformed files are listed four times because multiple operations occur during a rekey operation (the logging level is set to `INFO` and `audit` is enabled). This can result in an extremely large number of log entries. In the example below, a GuardPoint is opened with the `clear_key` key and saved with the `aes128` key.

Ensure that there are no errors in the DSM log that have to do with `dataxform` or DXF. Look for errors that contain strings like “denied” and “failed”. For example,

[DXF4376E] Data transform in guard point /opt/apps/dx9 failed for 6 files

[DXF4271E] Number of files in error due to a signal stopping the dataxform: 6

The example errors indicate that `dataxform` had been interrupted as it was actively transforming six files.

The `dataxform` messages indicate that `dataxform` is supported, `dataxform` detected a number of files, it transformed that same number of files, and then completed successfully.

If errors are generated, check the `dataxform_status_skip-`*`_gp`* file for a list of the files that were in the process of being transformed, but are now in an unknown state.

Using vordxf_* Files.

Data transformation activity is recorded and prepared for transmission to the DSM. One file that records transformation activity is updated dynamically and should be checked first should any problems arise. You may want to use the "`tail -f`" command on the file to monitor the data transformation process in real-time.

The data transformation log file resides in `/var/log/vormetric` on the host system.

The base name of the data transformation log file is `vordxf_`*`path_usr`*`.log`, where *path* is the underscore-separated (_) full path to the GuardPoint directory. *usr* is the name of the user executing `dataxform`. Only the UNIX `root` user or the Windows `Administrator` user can run `dataxform`. For example, `vordxf-_opt_apps_dx9_root.log`.

Always wait for the "`Data transform for guard point` *path* `completed ...`" message before making any changes to the GuardPoint.

```
...
2011-01-17 12:45:55.846 [VMD] [INFO ] [15792] [DXF4344I] Data
transform version 4.4.1.0
2011-01-17 12:45:55.857 [VMD] [INFO ] [15792] [DXF4429I] Data
transformation is supported on /opt/apps/dx9
2011-01-17 12:45:55.886 [VMD] [INFO ] [15792] [DXF4378I] Scan
found 63 files (1 KB) in 5 directories for guard point
/opt/apps/dx9
2011-01-17 12:45:55.889 [VMD] [INFO ] [15792] [DXF4366I] The
current operation took 0 hours, 0 minutes and 0 seconds
2011-01-17 12:45:56.220 [VMD] [INFO ] [15792] [DXF4380I]
Transformed 69 files (1 KB) of 63 files (1 KB) for guard point
/opt/apps/dx9
2011-01-17 12:45:56.224 [VMD] [INFO ] [15792] [DXF4366I] The
current operation took 0 hours, 0 minutes and 1 seconds
2011-01-17 12:45:57.242 [VMD] [INFO ] [15792] [DXF4371I] Data
transform skipped some files
2011-01-17 12:45:57.255 [VMD] [INFO ] [15792] [DXF4201I] The
file /opt/apps/dx9/filenothere01 was skipped. It was a soft
link
2011-01-17 12:45:57.283 [VMD] [INFO ] [15792] [DXF4201I] The
file /opt/apps/dx9/filenothere02 was skipped. It was a soft
link
2011-01-17 12:45:57.286 [VMD] [INFO ] [15792] [DXF4201I] The
file /opt/apps/dx9/filenothere03 was skipped. It was a soft
link
```

```
2011-01-17 12:45:57.289 [VMD] [INFO ] [15792] [DXF4200I] The
file /opt/apps/dx9/hardlink01 was skipped. It was an additional
hard link
2011-01-17 12:45:57.292 [VMD] [INFO ] [15792] [DXF4200I] The
file /opt/apps/dx9/hardlink02 was skipped. It was an additional
hard link
2011-01-17 12:45:57.299 [VMD] [INFO ] [15792] [DXF4200I] The
file /opt/apps/dx9/hardlink03 was skipped. It was an additional
hard link
2011-01-17 12:45:57.302 [VMD] [INFO ] [15792] [DXF4257I]
Number of additional hard links skipped: 3
2011-01-17 12:45:57.307 [VMD] [INFO ] [15792] [DXF4258I]
Number of soft links skipped: 3
2011-01-17 12:45:57.312 [VMD] [INFO ] [15792] [DXF4365I] The
data transform operation took 0 hours, 0 minutes and 2 seconds
2011-01-17 12:45:57.316 [VMD] [INFO ] [15792] [DXF4363I] Data
transform for guard point /opt/apps/dx9 finished but 6 files
were skipped
```

`dataxform` logging is tied to VMD logging. The size of the `vordxf_`*`path_usr`*`.log` file is set by the **Maximum File Size** text-entry box on the *FS Log* tab of the Management Console. The default maximum `vordxf_`*`path_usr`*`.log` file is 1 MB.

The following `vordxf_`*`path_usr`*`.log` excerpt shows a `dataxform` session that scans the directory and subdirectories for files to transform, performs a policy check with the DSM, and successfully transforms all but three files.

```
2011-01-10 14:37:03.970 [VMD] [INFO ] [26158] [DXF4344I] Data
transform version 4.4.1.0
2011-01-10 14:37:03.997 [VMD] [INFO ] [26158] [DXF4345I] Data
transform version 4.4.1.0, using policy aes128_to_clear_dx on
guard point /opt/apps/lib/dx4
2011-01-10 14:37:04.155 [VMD] [INFO ] [26158] [DXF4429I] Data
transformation is supported on /opt/apps/lib/dx4
2011-01-10 14:37:06.564 [VMD] [INFO ] [26158] [DXF4378I] Scan
found 10000 files (273 KB) in 5 directories for guard point
/opt/apps/lib/dx4
2011-01-10 14:39:25.797 [VMD] [INFO ] [26158] [DXF4380I]
Transformed 10003 files (273 KB) of 10000 files (273 KB) for
guard point /opt/apps/lib/dx4
2011-01-10 14:39:35.949 [VMD] [INFO ] [26158] [DXF4371I] Data
transform skipped some files
2011-01-10 14:39:35.980 [VMD] [INFO ] [26158] [DXF4201I] The
file /opt/apps/lib/dx4/filenothere01 was skipped. It was a soft
link
2011-01-10 14:39:35.986 [VMD] [INFO ] [26158] [DXF4201I] The
```

```
file /opt/apps/lib/dx4/filenothere02 was skipped. It was a soft
link
2011-01-10 14:39:35.989 [VMD] [INFO ] [26158] [DXF4201I] The
file /opt/apps/lib/dx4/filenothere03 was skipped. It was a soft
link
2011-01-10 14:39:36.000 [VMD] [INFO ] [26158] [DXF4258I]
Number of soft links skipped: 3
2011-01-10 14:39:36.050 [VMD] [INFO ] [26158] [DXF4363I] Data
transform for guard point /opt/apps/lib/dx4 finished but 3
files were skipped
Check the status files in /var/log/vormetric, or the status
file location you configured, for dataxform_status_skip-_gp
and dataxform_status_error-_gp files. The following is the
dataxform_status_skip-_gp file for the preceding example.
These are link files and link file cannot be transformed.
```

```
# cat dataxform_status_skip-_opt_apps_lib_dx4
```

System Response

```
Skipped, the name refers to a
softlink : /opt/apps/lib/dx4/filenothere01
Skipped, the name refers to a
softlink : /opt/apps/lib/dx4/filenothere02
Skipped, the name refers to a softlink :
/opt/apps/lib/dx4/filenothere03
#
```

An error status file was not generated. The `dataxform_status_error-_gp` file is generated when `dataxform` fails to complete. It did complete successfully, so no error status file is generated. The error status file consists of the full path of each file that was being processed when `dataxform` failed. The files are not necessarily corrupt. You must check each file in the list to determine its current status.

Using dataxform_status* Files

Several `dataxform_status_*` files are used to run `dataxform`. One file you generate. The other files are generated by `dataxform` to track the `dataxform` session. Each `dataxform_status` file contains specialized records of the last or current `dataxform` session. The files are placed in `/var/log/vormetric` by default, but you can specify an alternate location using the `--dir_recovery` argument to `dataxform`. The default location on some Windows systems is `\Documents and Settings\All Users\Application Data\Vormetric\DataSecurityExpert\Agent\logs`. (On Windows Vista and

Windows Server 2008, the status file is written to `\ProgramData\Vormetric\DataSecurityExpert\agent\log`.)

The files are:

```
dataxform_status-_gp
dataxform_status_alt-_gp
dataxform_status_error-_gp
dataxform_status_skip-_gp
```

where, *gp* is the full path to the GuardPoint. The slash (`/`) and backslash (`\`) path delimiters are replaced with underscores (_). For example,

```
# pwd
```

System Response

```
/var/log/vormetric
```

```
# ls dataxform_status*
```

System Response

```
dataxform_status_alt-_opt_apps_lib_dx1
dataxform_status_error-_opt_apps_lib_dx1
dataxform_status_skip-_opt_apps_lib_dx1
dataxform_status-_opt_apps_lib_dx1
dataxform_status-_opt_apps_lib_dx2_aa_dir
#
```

where, `/opt/apps/lib/dx1` and `/opt/apps/lib/dx2/aa_dir` are the actual GuardPoint paths.

On Windows, `dataxform_status-_gp` and `dataxform_status_alt-_gp` include the partition letter in the GuardPoint path. In the example below, the GuardPoint is on the `C:` partition:

```
dataxform_status-C_users_bubba_lib_dx1
dataxform_status-alt-C_users_bubba_lib_dx1
```

The format and use of each file are described below.

`dataxform_status-_gp` and **`dataxform_status_alt-_gp`** are used together to monitor data file processing. These status files indicate the threads that are started, the threads that are running, their status, and sequence.

For a `dataxform` failure or interruption, the **`dataxform_status-_gp`** and **`dataxform_status_alt-_gp`** files can be used to guesstimate the files that have been completed and the files that still must be transformed. Some manual verification will be needed to verify precisely the files that have and have not been

transformed. You can use this information to create a file list and manually resume the `dataxform` session, or, you can just leave these status files in place and resume automatic `dataxform`. Automatic `dataxform` will determine where to resume.

The structure of **`dataxform_status-`*`_gp`*** and **`dataxform_status_alt-`*`_gp`*** files is:

```
version=n
status=stat
operation=action
current=file
n in-progress files
seqno=n
hmac=<xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx>
```

where:

n is an internally used version number. It has no user value. Ignore it.

status is the current `dataxform` processing status. Status can be done, `in-progress`, `stopped`, or undone. An `in-progress` status indicates that `dataxform` is still processing data files. Do not work in, or access, the GuardPoint while `dataxform` is still `in-progress`. Otherwise, you risk corrupting data. A `stopped` status indicates that the last `dataxform` session was interrupted before it could complete. The done status means that `dataxform` completed. Note that, though `dataxform` completes, not all files are necessarily transformed. Check the log files for data files that have not been transformed. undone indicates that rekey from a file list had been performed.

action is the operation specified on the `dataxform` command line. It can be either `rekey` or `rekey_list`. Automatic `dataxform` is implicitly configured to operate with `rekey`.

file is the full path name of the file currently being processed. When `status` is done, the `current` parameter is blank. In every other case, `current` will be set to some value, such as the default unset value `-1`.

n is the total number of files being concurrently processed by `dataxform`. Each file is transformed as a separate sub-process, or thread. The number of sub-processes is hard-coded and varies by platform. On Windows it is 8 and on Solaris it is 28. You cannot configure the number of sub-processes. The *n* `in-progress files` parameter is set to zero when `dataxform` completes.

seqno is the sequence number. There are two status files, **`dataxform_status-`*`_gp`*** and **`dataxform_status_alt-`*`_gp`***. The `dataxform` utility writes status information to one file. When a sub-process

completes and a new data file is opened for transformation, the current status file is closed, and the other status file is opened with the new data file as the `current=`*file* file. The `seqno` number increments each time `dataxform` begins to process the next data file. If a `dataxform` session fails or is interrupted, check the `seqno` number. Use the status file with the higher `seqno` number to determine the files that were being processed when `dataxform` stopped. The status file with the lower `seqno` number indicates the last fully processed data file in the `current=`*file* parameter. The status file with the higher `seqno` number indicates the last opened data file in the `current=`*file* parameter. Most likely the last opened data file was not successfully transformed.

<*xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx*> This is a check value used to ensure the integrity of the status files, and can be ignored by the user.

By default, the log files are generated by `dataxform` during manual and automatic data transformation and placed in `/var/log/vormetric` on UNIX systems and `\Documents and Settings\All Users\Application Data\Vormetric\DataSecurityExpert\Agent\log` on some Windows systems. (On Windows Vista and Windows Server 2008, the status file is written to `\ProgramData\Vormetric\DataSecurityExpert\agent\log`. An alternate location can be specified using the `--dir_recovery` argument to `dataxform`.

A successful `dataxform` status file looks like:

```
# cat ./dataxform_status-_opt_apps_lib_dx2_aa_dir
```

System Response

```
version=5
status=done
operation=rekey
current=
0 in-progress files\
seqno=2004
hmac=6BI53B320C455A45E013AC21F353CE0BBCF159440B2B32F3F3
000FBB0BA7B3C5267D32D60385C786FEDA8C59D57F1C44
5588A9FCAB56CC642E8B2E5601D09CA7E9F6AFFA886ADR6
2A8FE559CF
C89B9F55G
3E65F4
I8FD574
#
```

Note that the `status` is done, there is no `current` file, and there are 0 files being processed. The `seqno` number is the number of files in the GuardPoint. There is no corresponding `dataxform_status_alt-_gp` file because the

`dataxform_status_alt-_gp` file is deleted when `dataxform` completes successfully. Also, look for a `dataxform_status_skip-_gp` file to see what files in the GuardPoint, if any, were detected but not transformed.

The `dataxform_status-_gp` and `dataxform_status_alt-_gp` files for a `dataxform` session are shown below.

```
# cat dataxform_status-_Guard
```

System Response

```
version=5
status=in-progress
operation=rekey
current=48986 /Guard/dir_47/boot/grub/stage2
8 in-progress files
48881 /Guard/dir_47/boot/grub/reiserfs_stage1_5
48936 /Guard/dir_47/boot/grub/fat_stage1_5
48333 /Guard/dir_47/boot/initramfs-2.6.32-
71.el6.x86_64.img
48758 /Guard/dir_47/boot/vmlinuz-2.6.32-71.el6.x86_64
46899 /Guard/dir_5/boot/initramfs-2.6.32-
71.el6.x86_64.img
48821 /Guard/dir_47/boot/config-2.6.32-71.el6.x86_64
48986 /Guard/dir_47/boot/grub/stage2
47316 /Guard/dir_5/boot/vmlinuz-2.6.32-71.el6.x86_64
seqno=738
hmac=2ED53B320C455A45E013AC21F353CE0BBCF159440B2B32F3F3
000FBB0AD9B3C5267D32D60385C786FEDA8C59D57F1C52
7788A9FCAB56CC642E8B2E5601D09CA7E9F6AFFA886ADE1
2A8FE559FC
C89B9F55E
3E65F5
D8FD544
```

```
# cat dataxform_status_alt-_Guard
```

System Response

```
version=5
status=in-progress
operation=rekey
current=49031 /Guard/dir_47/boot/grub/ufs2_stage1_5
8 in-progress files
49031 /Guard/dir_47/boot/grub/ufs2_stage1_5
48936 /Guard/dir_47/boot/grub/fat_stage1_5
48333 /Guard/dir_47/boot/initramfs-2.6.32-
71.el6.x86_64.img
```

```
48758 /Guard/dir_47/boot/vmlinuz-2.6.32-71.el6.x86_64
46899 /Guard/dir_5/boot/initramfs-2.6.32-
71.el6.x86_64.img
48821 /Guard/dir_47/boot/config-2.6.32-71.el6.x86_64
48986 /Guard/dir_47/boot/grub/stage2
47316 /Guard/dir_5/boot/vmlinuz-2.6.32-71.el6.x86_64
seqno=739
hmac=E113351BDB7497AAD150DEC57953DB0E57401404F564733E71
02A612158F65D7ACC2E1D9A75CB94ED91751397CDFD1CF
EE3483B58DFD6DD4DC722D
0CD47F89890DFA572DE7F3E4E9F
4

0A2
E78F4
F8
7B
DC8D603
```

The number that precedes every file path in the status file is the offset into the `dataxform` file list. It is used to quickly locate entries if `dataxform` needs to be restarted. This number of can be ignored.

Note the `seqno` numbers in both status files. `dataxform_status_alt-_Guard` has the higher `seqno` number so it is the true snapshot of what `dataxform` was doing while the files were captured or `dataxform` failed. The `dataxform` utility was running 8 sub-processes.

The other status file, `dataxform_status-_Guard`, is older because it has the lower `seqno` number. The `current=`*file* parameter shows the last loaded file only and it shows that the previous `current` file was `stage2`. Note that `stage2` is still listed in the alternate file, `dataxform_status-_Guard`. This indicates that `stage2`, like all the files listed in the alternate file, was still being processed when `dataxform` stopped.

The final line with the prefix `hmac=` is a check sum field. This is used to verify that the file content is intact should the transform be interrupted and need to be restarted.

All data in the status files after this line are 'noise'. This occurs for efficiency reasons. The `dataxform` program writes the status files using a simple block write of all the status information into the existing file, overwriting any entries already present. If, as often happens, the existing status file is larger than the data being written, then some of the stale entries may be visible after the final checksum line. The additional effort of removing this stale data for each file transformation would

add a significant additional amount of time to the overall transformation, so this 'noise' is simply left and can be ignored.

Using dataxform_auto_config

The `dataxform_auto_config` file is user-generated and consists of one or two lines and is placed in the GuardPoint to initiate automatic data transformation. Usually, it contains just one line, "`version=1`". The version is an internally used number that currently must be set to `1`. File usage is described in "To run automatic data transformation" on page 53.

```
# cat dataxform_auto_config
```

System Response

```
version=1
#
```

NOTE: When you assign a rekey policy to a GuardPoint, a message like the following is issued in the `secfsd.log` file every 20 seconds until a `dataxform_auto_config` file is placed in the GuardPoint, manual `dataxform` is run on the GuardPoint, or a non-rekey policy is applied to the GuardPoint: `/opt/vormetric/DataSecurityExpert/agent/secfs/.sec/current/bin/secfsd[29252]: Information: Mon Mar 28 11:54:37 2011 do_dataxform: no auto config file exists; not starting dataxform`. This information is not displayed in the *Logs* window.

Using dataxform_auto_lock

The `dataxform_auto_lock` file is a two-line file generated by `dataxform` and is placed in the GuardPoint during manual and automatic data transformation. The first line is the name of the host system on which `dataxform` is being executed. The second line indicates the `dataxform` status. Status is "in-progress", "stopped", or "done". A stopped status indicates that the last `dataxform` session was stopped before it reached completion. An example of a successful `dataxform` run is:

```
# cat dataxform_auto_lock
```

System Response

```
solaris120
done
#
```

If a `dataxform_auto_lock` file is present in a GuardPoint, and/or other `dataxform` configuration files are in `/var/log/vormetric` (except for `dataxform_auto_config`) `dataxform` will exit with an error. You cannot perform a new data transformation in the Guardpoint unless you delete the `dataxform_auto_lock` file and the `dataxform_*` files in `/var/log/vormetric`. The recommended way to remove old status files is to run `dataxform` with the `--cleanup` argument.

Warning! Do not use `--cleanup` if a previous `dataxform` session did not complete successfully (see "Recovering a Failed dataxform Session" on page 69).

Example of using the `--cleanup` argument to remove old status files:

```
# cat /opt/apps/lib/dx1/ad_dir/dataxform_auto_lock
```

System Response

```
solaris120
done
```

```
# dataxform --cleanup --nq --gp /opt/apps/lib/dx1/ad_dir
```

System Response

```
About to remove the data transformation status files
#
```

You can now run `dataxform` again and successfully rekey files in the GuardPoint.

Recovering a Failed dataxform Session

A `dataxform` session can fail for different reasons: `dataxform` can be canceled, the process is killed, the system crashes, and so on. The recovery process is similar

for manual and automatic `dataxform`, however, for incomplete automatic `dataxform` sessions, see "Restarting an incomplete automatic dataxform session".

Restarting an incomplete automatic dataxform session

Although unlikely, if an automatic `dataxform` session fails to complete, your intervention may be required to allow it to resume.

In the case of a system restart--for example after a power failure--the automatic session will be resumed when the GuardPoint is mounted during the boot sequence. However, if the `dataxform` session terminated unexpectedly (for example, if the session had been killed), then the system will not restart it automatically, and the GuardPoint status needs to be reset so that another automatic session will be initiated.

The following steps reset the GuardPoint status:

1. Verify that the session really has terminated unexpectedly by verifying that no existing `dataxform` process is running. Use the standard system tools (`ps` or task manager).
2. From the Management Console, disable the GuardPoint, then wait until the GuardPoint status on the agent no longer lists the GuardPoint.
3. From the Management Console, enable the GuardPoint.

An automatic `dataxform` session will start soon after this, and the session will continue from where it previously stopped.

To recover from a failed dataxform session

The following is an example of how to recover from a failed `dataxform` session. The GuardPoint in this example is `/opt/apps/dx9`

1. A manual `dataxform` session is run, then canceled using Ctrl-c:

```
# dataxform --rekey --nq --print_stat --preserve_modified_time
--gp /opt/apps/dx9
```

System Response

```
Checking if data transform is supported for guard point
/opt/apps/dx9
Data transformation is supported on /opt/apps/dx9
About to perform the requested data transform operation
-- Be sure to back up your data
```

```
-- Do not access files in the guard point during the transform
process
-- Please do not attempt to terminate the application
Scan found 10003 files (273 KB) in 5 directories for guard point
/opt/apps/dx9
The current operation took 0 hours, 0 minutes and 2 seconds
Shutting down data transform: received fatal signal 2
Transformed 1126 files (24 KB) of 10003 files (273 KB) for guard
point /opt/apps/dx9
The current operation took 0 hours, 0 minutes and 7 seconds
Data transform got errors on some files
The file /opt/apps/dx9/datafile931 could not be transformed, a
signal stopped the data transform
The file /opt/apps/dx9/datafile1102 could not be transformed, a
signal stopped the data transform
The file /opt/apps/dx9/datafile1121 could not be transformed, a
signal stopped the data transform
The file /opt/apps/dx9/datafile1100 could not be transformed, a
signal stopped the data transform
The file /opt/apps/dx9/datafile1120 could not be transformed, a
signal stopped the data transform
The file /opt/apps/dx9/datafile1132 could not be transformed, a
signal stopped the data transform
Number of files in error due to a signal stopping the dataxform: 6
The data transform operation took 0 hours, 0 minutes and 7 seconds
Could not complete data transform for guard point /opt/apps/dx9,
data transform was interrupted by a signal
Data transform for guard point /opt/apps/dx9 finished but 6 files
were not processed due to errors
#
```

The `dataxform_status-_opt_apps_dx9`, `dataxform_status_error-_opt_apps_dx9`, and `dataxform_dir_list-_opt_apps_dx9` files are created in the log directory, or the GuardPoint, depending on the command line options. By default, all status and log files go to `/var/log/vormetric`. If `--status_gp` was included on the command line, the status files go in the GuardPoint.

The dataxform session log file, `/var/log/vormetric/vordxf-_opt_apps_dx9_root.log`, is updated. It displays the same basic information as displayed on the terminal screen.

2. The `dataxform` messages indicate that the session had been interrupted and that a number of files (6) are in an unknown state due to the interruption.
3. You may optionally use the `--recovery` option to generate a set of files that track the progress `dataxform` made in the GuardPoint (see later). However this step is not required and can be deferred until after a new `dataxform` session has been run to transform the remaining files.

4. Resume the data transformation run, using the same command as you used to start it before.

```
# dataxform --rekey --nq --print_stat --preserve_modified_time
--gp /opt/apps/dx9
```

System Response

```
Checking if /opt/apps/dx9 is a guard point with a rekey
policy applied /opt/apps/dx9 is a guard point with a
rekey policy applied
Automatic data transform status for /opt/apps/dx9:
previous attempt did not complete
Note: data from a previous dataxform run is being used
About to perform the requested data transform operation
-- Be sure to back up your data
-- Please do not attempt to terminate the application
Scan found 10003 files (273 KB) in 5 directories for
guard point /opt/apps/dx9
Transformed 10001 files (273 KB) of 10003 files (273 KB)
for guard point /opt/apps/dx9
Data transform got errors on some files
The file /opt/apps/dx9/datafile931 could not be
transformed, it was in progress in the previous data
transform
The file /opt/apps/dx9/datafile1102 could not be
transformed, it was in progress in the previous data
transform
The file /opt/apps/dx9/datafile1121 could not be
transformed, it was in progress in the previous data
transform
The file /opt/apps/dx9/datafile1100 could not be
transformed, it was in progress in the previous data
transform
The file /opt/apps/dx9/datafile1120 could not be
transformed, it was in progress in the previous data
transform
The file /opt/apps/dx9/datafile1132 could not be
transformed, it was in progress in the previous data
transform
The data transform operation took 0 hours, 0 minutes and
7 seconds
Could not complete data transform for guard point
/opt/apps/dx9, data transform was interrupted by a
signal
Data transform for guard point /opt/apps/dx9 finished
```

```
but 6 files were not processed due to errors
#
```

5. If for some reason this session also did not complete, performing the same operation again should continue from where the previous session ended. The status records are accumulated across each session.
6. Use the `--recovery` option to generate a set of files that track the progress `dataxform` made in the GuardPoint. Use these files later to determine which files in the GuardPoint have not been transformed.

Warning! Do not run the `--recovery` option more than once, as a second run may overwrite vital information required to recover any files that did not transform correctly.

```
# dataxform --recovery --gp /opt/apps/dx9
```

System Response

```
Note: data from a previous dataxform run is being used
Number of files previously in error due to a signal stopping the
dataxform: 6
Scan found 10010 files (273 KB) in 6 directories for guard point
/opt/apps/dx9
Generating list of files previously transformed on /opt/apps/dx9
Data transform got errors on some files
The file /opt/apps/dx9/datafile931 was previously in error. A signal
stopped the dataxform
The file /opt/apps/dx9/datafile1102 was previously in error. A
signal stopped the dataxform
The file /opt/apps/dx9/datafile1121 was previously in error. A
signal stopped the dataxform
The file /opt/apps/dx9/datafile1100 was previously in error. A
signal stopped the dataxform
The file /opt/apps/dx9/datafile1120 was previously in error. A
signal stopped the dataxform
The file /opt/apps/dx9/datafile1132 was previously in error. A
signal stopped the dataxform
Number of files in error due to a signal stopping the dataxform: 6


The dataxform_files_todo-_opt_apps_dx9 and dataxform_files_done-
_opt_apps_dx9 are created, and the
```

```
/var/log/vormetric/dataxform_status-_opt_apps_dx9 file may be
updated.
```

```
# cat dataxform_status-_opt_apps_dx9
```

System Response

```
version=5
status=done
operation=rekey
current=
0 in-progress files
seqno=2
hmac=5449453CBAEFCC01EC02542650D8C1040D762D213829F8B3CF967DC578320A
475F705C11D3BAF74D588630CE8078AF46
#
```

This file indicates that the `dataxform` session had completed.

7. The `/var/log/vormetric/vordxf-_opt_apps_dx9_root.log` file is updated. It displays the same basic information displayed on the terminal screen.
8. The primary files of interest are `dataxform_files_todo-_opt_apps_dx9`, `dataxform_files_done-_opt_apps_dx9`, and `dataxform_status_error-_opt_apps_dx9`.

   `dataxform_files_todo-_opt_apps_dx9` lists the files that `dataxform` had not touched and are yet to be transformed, and any files for which a transform was attempted but for some reason failed.

   `dataxform_files_done-_opt_apps_dx9` lists the files that `dataxform` rekeyed successfully. Leave these files alone.

   `dataxform_status_error-_opt_apps_dx9` lists the files that were being processed at the time an error occurred—for example, when `dataxform` was interrupted in the above example. These are the files that have to be checked individually to determine if they had been processed. They may or may not have been completely processed.

   ```
   # cat dataxform_status_error-_opt_apps_dx9
   ```

System Response

```
Error, was in progress during a previous session:
/opt/apps/dx9/datafile931
Error, was in progress during a previous session:
/opt/apps/dx9/datafile1102
Error, was in progress during a previous session
/opt/apps/dx9/datafile1121
Error, was in progress during a previous session:
```

```
/opt/apps/dx9/datafile1100
Error, was in progress during a previous session:
/opt/apps/dx9/datafile1120
Error, was in progress during a previous session:
/opt/apps/dx9/datafile1132
#
```

Note that if more than one session restart was required to complete the `dataxform` run, there may be repeated entries in the above list.

```
# cat dataxform_files_todo-_opt_apps_dx9
```

System Response

```
/opt/apps/dx9/datafile931
/opt/apps/dx9/datafile1100
/opt/apps/dx9/datafile1102
/opt/apps/dx9/datafile1120
/opt/apps/dx9/datafile1121
/opt/apps/dx9/datafile1132
#
```

```
# head -12 dataxform_files_done-_opt_apps_dx9
```

System Response

```
/opt/apps/dx9/datafile1
/opt/apps/dx9/datafile2
/opt/apps/dx9/datafile3
/opt/apps/dx9/datafile4
/opt/apps/dx9/datafile5
/opt/apps/dx9/datafile6
/opt/apps/dx9/datafile7
/opt/apps/dx9/datafile8
/opt/apps/dx9/datafile9
/opt/apps/dx9/datafile10
/opt/apps/dx9/datafile11
/opt/apps/dx9/datafile12
#
```

9. Disable the GuardPoint with the rekey policy through the Management Console. Do not re-apply the regular policy because you are not able to use the in the GuardPoint. A proper rekey policy restricts access to all the files in the GuardPoint. You have to disable the rekey policy so you can access all the files in the GuardPoint and determine their transformation status.
10. At this point, you should restore the files that were named in the error listing above from a backup, and run a transformation session for just these files, using the "`todo`" list (`dataxform_files_todo-_opt_apps_dx9`) to select which files are to be

transformed. However, depending on the exact nature of the error reported, some entries may need to be removed from the "`todo`" list, for example, if somehow a file no longer exists, then attempting to transform it again will obviously not work.

11. If you are running automatic `dataxform`, remove the `dataxform_auto_conf` file from the GuardPoint.
12. Re-apply the rekey policy to the GuardPoint, but first be sure that your current directory is not the GuardPoint.
13. Verify that the policy has been successfully re-applied.

```
# secfsd -status guard
```

System Response

```
GuardPoint Policy Type ConfigState Status Reason
---------- ------ ---- ----------- ------ ------
/opt/apps/dx1 allowAllOps_fs local guarded guarded N/A
/opt/apps/dx3 denyAllOps_fs local guarded guarded N/A
/opt/apps/dx4 allowAllOps_fs local guarded guarded N/A
/dev/dsk/c0t0d0s7 allowAllOps_rd rawdevice guarded guarded N/A
/opt/apps/dx9 clear_to_aes128_dx local guarded guarded N/A
#
```

14. Run `dataxform` on just the files in the `todo` list. For example:

```
# dataxform --rekey_list --file_list
var/log/vormetric/dataxform_files_todo-_opt_apps_dx9 --nq --
print_stat --preserve_modified_time --gp /opt/apps/dx9
```

System Response

```
Checking if data transform is supported for guard point
/opt/apps/dx9
Data transformation is supported on /opt/apps/dx9
Previous status information does not relate to a --rekey_file
operation.
About to perform the requested data transform operation
-- Be sure to back up your data
-- Please do not access files in the guard point during the
transform process
-- Please do not attempt to terminate the application
Starting data transform of /opt/apps/dx9 for files listed in
/var/log/vormetric/dataxform_files_todo-_opt_apps_dx9
```

```
The data transform operation took 0 hours, 1 minutes and 20 seconds
bash-3.00#
```

In this case the `dataxform` completes successfully. The `error` file was removed because no errors were encountered in the rekey process.

15. Disable the rekey policy.
16. If you are satisfied with the successful completion, clean up the files left by the rekey process.

    ```
    # dataxform --cleanup --gp /opt/apps/dx9
    ```

System Response

```
About to remove the data transformation status files
Do you wish to continue (y/n)? y
#
```

17. Apply a regular policy that uses the applied key.
18. Check that the access controls and encryption keys configured in the policy are working as expected.

Automatic and Manual GuardPoints

A File System Agent GuardPoint is usually applied immediately after it is configured in the Management Console. This is called an Automatic GuardPoint. However, GuardPoints can also be applied later on a host system. This is called a Manual GuardPoint.

NOTE: Manual GuardPoints can be applied to UNIX platforms only.

When would you want to apply the GuardPoint later? Consider the case of a 2-node protected host cluster configured as active/passive in a cluster environment, such (Veritas Cluster Server (VCS), IBM PowerHA (formerly HACMP), or Microsoft Cluster Server (MSCS). There are two nodes, one which is currently active and the other that is currently inactive. Both nodes are locked. You apply GuardPoint protection to active nodes only. You should never apply a GuardPoint to a passive node. If the active node develops a problem and tries to switch over to the inactive node, the cluster process will fail to switch over because the mirror directory on the inactive node is currently mounted on the active node. The solution is for the cluster

process to unmount (for example, unguard) the currently active node, place it in an inactive state, place the old inactive node in an active state, and then mount (for example, guard) the mirror directory on the newly active node. Inappropriate switching on any AIX system can spawn messages like:

- "invalid GuardPoint"
- "The directory is not on cluster file system partition shared across nodes"
- "secfsd Failed to unguard <dirpath> - will retry later"
- "Agent is calling clean for resource <resource name> because the resource became OFFLINE unexpectedly, on its own."

Generally, when you get messages like these, check that only active nodes are properly guarded.

Automatic and Manual GuardPoints are set in the *Edit Host* window, *Guard File System* sub-window.

The GuardPoint type is usually set to `Directory (Auto Guard)` for file system based directories and to `Raw or Block Device (Auto Guard)` when applying GuardPoint protection to raw or block devices. When an auto GuardPoint is applied, regardless if it is a file system directory or a raw device, the change is pushed to the host system, and the GuardPoint is applied immediately. This is evident by using the `df` command to display `secfs` mounts (for example, GuardPoints) or `secfsd` to display the GuardPoints themselves. The `secfsd` output shows a guard type of `local` for directories configured with `Directory (Auto Guard)`.

Example

```
# df
```

System Response

```
Filesystem                      1K-
blocks     Used  Available Use%  Mounted on
/dev/mapper/VolGroup00-LogVol00  40123784  11352236
  26733380  30%  /
/dev/sda1                          101086     14590
81277  16%  /boot
none                               254492         0     254492   0%
  /dev/shm
/opt/vormetric/DataSecurityExpert/agent/secfs/.sec
                                 40123784  11352236   26733380  30%
/opt/vormetric/DataSecurityExpert/agent/secfs/.sec
/opt/apps/apps1/tmp              40123784  11352236   26733380  30%
```

```
 /opt/apps/apps1/tmp
/opt/apps/apps1/lib              40123784  11352236   26733380  30%
 /opt/apps/apps1/lib
/opt/apps/apps1/doc              40123784  11352236   26733380  30%
 /opt/apps/apps1/doc
```

```
# secfsd -status guard
```

```
GuardPoint           Policy
                    Type ConfigState  Status Reason
----------           ------                    ----    --------   ---
---     ---
/opt/apps/apps1/tmp  allowAllOps_fs           local
guarded    guarded    N/A
/opt/apps/apps1/lib  allowAllRootUsers_fs     local
guarded    guarded    N/A
/opt/apps/apps1/doc  allowAllOps-winusers1_fs local
guarded    guarded    N/A
#
```

When a manual GuardPoint is applied, regardless if it is a file system directory or a raw device, the change is pushed to the host system only. The host is aware of the GuardPoint but the host does not mount it. This is indicated in the `Type` column of the "`secfsd -status guard`" output. For example, the GuardPoint `/opt/apps/apps2/bin` has been configured with `Directory (Manual Guard)` so the guard type is set to "`manual`".

```
# secfsd -status guard
```

```
GuardPoint           Policy
                   Type   ConfigState  Status       Reason
----------           ------                    ----   --------     --
----        ---
/opt/apps/apps1/tmp  allowAllOps_fs
          local  guarded      guarded       N/A
/opt/apps/apps1/lib  allowAllRootUsers_fs     local  guarded      g
uarded       N/A
/opt/apps/apps1/doc  allowAllOps-winusers1_fs
local  guarded      guarded       N/A
/opt/apps/apps2/bin  HR_policy01
            manual unguarded    not guarded  Inactive
#
```

Note the `Type` value. A `Type` of `manual` indicates a manual GuardPoint. A `Type of local` indicates an automatic GuardPoint.

A manually applied GuardPoint retains a yellow triangle status (Pending) in the Management Console until the GuardPoint is applied on the host. After the

GuardPoint is applied on the host, and the host communicates the change to the server, the status changes to a green ball (Normal). It returns to the yellow triangle when the GuardPoint is manually unguarded.

Use the `secfsd` command to guard and unguard `Directory (Manual Guard)` and `Raw or Block Device (Manual Guard)` GuardPoints. The `secfsd` syntax is:

```
secfsd -guard path
secfsd -unguard path
```

NOTE: In zone-based File System Agent deployments, such as Solaris Zones, always specify paths relative to the global zone, never the local zone. Also, you must guard and unguard manual GuardPoints in the global zone.

For example, to manually guard and unguard a file system directory:

1. Configure a GuardPoint with the type `Directory (Manual Guard)`.
2. Log onto the protected host with File System Agent as the root user.
3. Wait until the configuration change is downloaded to the protected host.

   You can run the status command until you see the manual GuardPoint. For example:

   ```
   # secfsd -status guard
   ```

System Response

```
GuardPoint        Policy          Type   ConfigState  Status
      Reason
----------       ------         ----   -----------  ------         -
-----
/opt/apps/etc     allowAllOps_fs  manual unguarded    not guarded
     N/A
/opt/apps/lib/dx3 allowAllOps_fs  local  guarded      guarded
     N/A
#
```

4. Enable the GuardPoint.

   ```
   # secfsd -guard /opt/apps/apps2/bin
   ```

System Response

```
secfsd: Guard initiated
#
```

The GuardPoint is active and the policy is enforced.

5. Disable the GuardPoint.

```
# secfsd -unguard /opt/apps/apps2/bin
```

System Response

```
secfsd: Unguard initiated
#
```

Running dataxform in an Oracle database on an HP-UX system

Attempts to run automatic `dataxform` in an Oracle 9i database on HP-UX 11iv1 can fail to transform every file if the number of threads is not enough for the number of files to be transformed.

When a set of files is passed to `dataxform`, `dataxform` is triggered and creates a user thread for each file. Each user thread does a `REKEY ioctl` with the filename and passes it to `secfs2`. Multiple kernel threads can also be created per file, depending on the size of the file. This can exhaust the system resources. The function that generates kernel threads can fail and return `ENOMEM`, which means that to create the thread, it would have to exceed the `nkthread` kernel tunable limit. This condition is encountered only if `freekthread_cnt` becomes 0, in which case new threads must be allocated. Thread size and number are set in `dataxform` by including the `--mt`, `--thd`, and `--buf_size` options.

This is an HP-UX system-specific issue. It depends entirely on the HP-UX kernel configurable parameters and the system hardware specifications. Tune the system per Oracle documentation so that Oracle can create and run databases. Check that there are enough kernel threads available before starting `dataxform` (for example, inside `REKEY_CHECK_SUPPORT ioctl`). For `dataxform` to work on HP-UX, kernel tunable "nkthread" should be set to a higher value. The current recommended values for Oracle are:

```
nkthread >= 6000
nproc >= 4000
```

This issue applies to automatic `dataxform` only.

dataxform man page

Location

`dataxform` is located at:

- (UNIX) `/opt/vormetric/DataSecurityExpert/agent/vmd/bin/dataxform`

  Add the path for the directory that contains `dataxform` to the user $PATH environment variable so you can run `dataxform` without having to specify the full path. Also, a symbolic link named `/usr/bin/dataxform`. is created during File System Agent installation and `/usr/bin` is usually defined in the $PATH variable.
- (Windows) `C:\Program Files\Vormetric\DataSecurityExpert\Agent\vmd\bin\dataxform.exe`

Syntax

Some of the more common uses are:

```
# dataxform --version

dataxform --cleanup --nq --gp /opt/apps/dx2

dataxform --recovery [--file_list file] --gp dir
          [--dir_recovery path][--nq]

dataxform --rekey_supported -gp dir

dataxform --rekey --gp dir [--nq]

dataxform --rekey_list --file_list file --gp dir

dataxform --scan --gp dir

dataxform --deep_scan --gp dir

dataxform --check_links --gp dir
```

Option	Description (Parameter)

--buf_size	Sets the size of the kernel buffers that are allocated to run `dataxform`. Buffer sizes range between 4 and 128 KB. The default buffer size is 128 KB. The selected default has been empirically determined to be the safest and most efficient. We strongly recommend that you do not change the default value. (integer between 4 and 128, inclusive)
--check_links	Lists the subdirectories in the GuardPoint or regular directory, the number of files in the top directory and in each subdirectory, and the total number of files and directories. The disk usage for each directory, and total disk usage, are also listed. In addition, this argument scans for hard links in the GuardPoint or regular directory. Use the `--gp` option to specify the GuardPoint or regular directory to scan. No files are re-keyed. This operation generates only a list of hard-link files. (None)
--cleanup	Deletes the status files that were generated by a previous `dataxform` session and that prevents you from running another `dataxform` session. It removes the `dataxform_auto_lock`, `dataxform files`, and `dataxform_status` files from a GuardPoint. It also deletes the `dataxform_status-`*gp* file from `/var/log/vormetric`. If those files are detected by `dataxform`, `dataxform` will abort with the message: "`Automatic data transform status for` *gp*`: previous attempt completed.`" Use `--cleanup` to remove these files, or remove them manually, after a `dataxform` session. Include the `--gp` option to specify the GuardPoint. If the `--dir_recovery` argument was used to output the `dataxform_*` files to a different location, be sure to include `--dir_recovery` and the full path to the alternate directory when you run cleanup. If you are running automatic `dataxform`, disable or remove the GuardPoint before using the `--cleanup` argument. Enable or reapply the GuardPoint afterwards. You must do this because once automatic `dataxform` completes it is done. It no longer checks the GuardPoint for a `dataxform_auto_config` file. When the GuardPoint is enabled or reapplied, `dataxform` is reactivated and searches for the file. (None)
--deep_scan	Lists the subdirectories in the GuardPoint or regular directory, the number of files in the top directory and in each subdirectory, and the total number of files and directories. The disk usage for each directory, and total disk usage, are also listed. In addition, this argument creates a set of simulation files of various sizes in the GuardPoint, and uses these to estimate how long it would take to rekey the actual directory or GuardPoint. See also "To estimate a dataxform runtime period" on page 50. (None)

--dir_recovery	Allows you to specify where `dataxform` status files are placed. The status files are output in one of three places: 1) Without any arguments, the status files are placed in `/var/log/vormetric` on a UNIX system or the `log` folder on a Windows system. Windows log folders are located at: (Windows Server 2003) C:\Documents and Settings\All Users\Application Data\Vormetric\DataSecurityExpert\agent\log (Windows Vista and Windows Server 2008) `C:\ProgramData\Vormetric\DataSecurityExpert\agent\log` (Windows XP) C:\Documents and Settings\All Users.WINDOWS\Application Data\Vormetric\DataSecurityExpert\agent\log 2) With the `--status_gp` argument, the status files are placed in the GuardPoint. 3) With the `--dir_recovery` argument, the status files are placed in a user-specified location. The name of the status files created by `--dir_recovery` are `dataxform_status-_gp` and `dataxform_status-alt_gp`, where *gp* is the underscore-separated path to the GuardPoint. For example, if the path to the GuardPoint is `/home/apps/lib/dx1`, then a status file name would be `dataxform_status-_home_apps_lib_dx1`. The `dataxform` file, `dataxform_auto_lock`, is always written to the GuardPoint. (directory name)
--encrypt_sparse_file_holes	Checks for "holes" in sparse files and fills and encrypts them. The `dataxform` utility processes each rekey block in 1k "chunks" (the rekey block is on a 1K boundary and multiples of 1K in size). Each "chunk" consisting of all zeros is considered a "hole". By default, holes on UNIX systems are not rekeyed or written, thus keeping the file size small. It is quicker and more efficient to process every block of Windows data as regular, non-sparse data. The default on UNIX systems is: `--preserve_sparse_files`. Windows users should always include: `--encrypt_sparse_file_holes` on the `dataxform` command line. (None)
--file_list	Used with `--recovery` to specify the output file name, or with `--rekey_list`, to specify the input file name. When used with `--recovery`, the output file name is automatically appended with "`_done`". `--file_list` takes one argument, ***`file`***. (file name)
--gp	Specifies the full path to the GuardPoint directory to process. (GuardPoint)
--mt	The `--mt` option sets the maximum number of threads allowed to transform one file. Valid values are between 1 and 16. We recommend that you do not change the default values for "Sparse Files". To increase dataxform performance, you may want to start with the default value and gradually increase the maximum threads value. See also "--thd" on page 86. (integer between 1and 16 inclusive)

--nq	The `--nq` ("no queries") option does rekeying without prompts. Without this option, `dataxform` prompts you to verify that you want to continue with the rekey operation. (None)
--preserve_access_time	Directs `dataxform` to leave the last-accessed time of files intact during the rekey process. This option is useful when the last-accessed time is used to trigger file backups. (None)
--preserve_modified_time	Directs `dataxform` to leave the last-modified time of files intact during the rekey process. This option is useful when the last-modified time is used to trigger file backups. (None)
--preserve_sparse_files	Checks for "holes" in sparse files and preserves them in the rekeyed file. The `dataxform` utility processes each rekey block in 1k "chunks" (the rekey block is on a 1K boundary and multiples of 1K in size). Each "chunk" consisting of all zeros is considered a "hole". Holes are not rekeyed nor written, therefore creating a sparse file, which can be considerably smaller in size. Sparse files are preserved by default on UNIX systems. This `dataxform` option is provided only for backward compatibility. When run on a Windows host, include `--encrypt_sparse_file_holes` on the `dataxform` command line. See "--encrypt_sparse_file_holes". (None)
--print_stat	Displays the time it takes `dataxform` to complete each phase of the transformation process. (None)
--recovery	Generates the files needed to complete an interrupted `dataxform` session. The generated files are `dataxform_files_done-`*_path* and `dataxform_files_todo-`*_path*, and they are placed in `/var/log/vormetric` on UNIX systems. Use the `--gp` option to specify the GuardPoint. (None)
--rekey	The `––rekey` option rekeys all the files in the GuardPoint specified by the `--gp` option. (None)
--rekey_list	Same as `--rekey`, except that `dataxform` transforms only the files in the file specified in the `--file_list` option. This option is typically used to recover a failed `dataxform` session. Use the `--gp` option to specify the GuardPoint. (None)

--rekey_supported	Checks the specified directory to determine if it is a valid GuardPoint, if a rekey policy is currently applied, and if anyone is currently accessing the directory. The `--gp` option specifies the GuardPoint to check. No files are re-keyed. This operation indicates only if the specified GuardPoint is ready to be rekeyed. (None)
--scan	Lists the subdirectories in the GuardPoint or regular directory, the number of files in the top directory and in each subdirectory, and the total number of files and directories. The disk usage for each directory, and total disk usage, are also listed. When run on a GuardPoint that had already been transformed but not cleaned, `--scan` also returns the number of hard links and soft links that were skipped in the previous `dataxform` session. No files are re-keyed. Use the `--gp` option to specify the GuardPoint. (None)
--status_gp	Causes `dataxform` to put the status, run, and error files in the GuardPoint rather than in the log directory. See also "Using dataxform_status* Files" on page 62. (None)
--status_interval	Sets the time interval (in seconds) at which data transformation status messages are sent to the DSM. The default is `300` seconds (5 minutes). (seconds)
--thd	Sets the number of threads that `dataxform` can use to transform files. A "thread" equates to a file. The more threads specified, the more files that are transformed concurrently. You may process up to 32 files concurrently. Threads are numbered 0 through 31, but the values you enter are 1 through 32. 0 indicates all 32 threads. The default number of threads is 4 times the number of CPUs (4 * #CPUs). If messages like "access denied" are displayed or files are being skipped, try reducing the number of threads. If the errors still occur after the number of threads is set to 1, the errors are not due to `dataxform` processes colliding. Most likely there is something wrong with the files or policy permissions. The default value has been empirically determined to be the safest and most efficient. We strongly recommend that you do not increase the default value. See also "--mt" on page 84. (integer between 1and 32, inclusive)
--version	Displays `dataxform` version information. (None)

Unencrypting Data

There are two methods for unencrypting data, also known as, migrating the data back to clear_key:

Copy method

1. Stop all applications accessing the GuardPoint.
2. Log on as a user who can see clear text for all files in the GuardPoint.
3. Copy all files from the GuardPoint to a new directory that is not guarded.
4. In the DSM, unguard the GuardPoint. Make sure that it does not display in the DSM.
5. Delete all files in the original directory.
6. Copy all files from the unguarded/clear text directory to the original directory.
7. Start any application once all files have finished copying.

NOTE: Do not rename directories. This will not result in unencrypting directories.

Dataxform method (Dataxform guard point method)

1. Stop any application accessing the GuardPoint.
2. In the DSM, clone the original policy that you used to encrypt the data.
3. In the DSM, reverse the keys. Put the original key in for the key_selection rule and the clear_key in for the transformation rule
4. In the DSM, guard with that policy.

NOTE: Make sure the status is green before you guard with that policy.

5. Run dataxform rekey on the GuardPoint:

```
# dataxform --rekey --gp /<dirName> --preserve_modified_time
```

Example

```
# dataxform --rekey --gp /DataSecurity/mydir --
preserve_modified_time
```

6. Run dataxform cleanup on the GuardPoint:

   ```
   # dataxform --cleanup --gp /<dirName>
   ```

   Example

   ```
   # dataxform --cleanup --gp /DataSecurity/myDir
   ```

7. In the DSM, unguard the Guard Point.

   Data should now be in clear text which means that it's unencrypted.

GLOSSARY

B

access control
The ability of Vormetric Transparent Encryption (VTE) to control access to data on protected hosts. Access can be limited by user, process (executable), action (for example read, write, rename, and so on), and time period. Access limitations can be applied to files, directories, or entire disks.

admin administrator
The default DSM administrator created when you install the DSM. `admin` has DSM System Administrator privileges and cannot be deleted.

Administrative Domain
(domains). A protected host or group of protected hosts on which an DSM administrator can perform security tasks such as setting policies. Only DSM administrators assigned to a domain can perform security tasks on the protected hosts in that domain. The type of VTE tasks that can be performed depends on the type of administrator. See also **"local domain"**.

administrator
See **"DSM Administrator and types"**.

Agent utilities
A set of utilities installed with the VTE agents and run on protected hosts. These utilities provide a variety of useful functions such as gathering protected host and agent configuration data, registering agents on the DSM, and encrypting data on the protected host.

All Administrator, **Administrator of type All**
The DSM Administrator with the privileges of all three administrator types: *System*, *Domain* and *Security*.

appliance
The DSM server. Often referred to as a *DSM hardware appliance*, which is a hardened DSM server provided by Vormetric, or as a *DSM virtual appliance*, which is the software version of the DSM to be deployed by the customers as a virtual machine.

asymmetric key cryptography
See *public key cryptographic algorithm.*

asymmetric key pair
A public key and its corresponding private key used with a public key algorithm. Also called a key pair.

authentication
A process that establishes the origin of information, or determines the legitimacy of an entity's identity.

authorization
Access privileges granted to an entity that convey an “official” sanction to perform a security function or activity.

block devices
Devices that move data in and out by buffering in the form of blocks for each input/output operation.

catch-all rule
The last policy rule that applies to any GuardPoint access attempt that did not fit any of the other rules in the policy.

certification authority or CA
A trusted third party that issues digital certificates that allow a person, computer, or organization to exchange information over the Internet using the public key infrastructure. A digital certificate provides identifying information, cannot be forged, and can be verified because it was issued by an official trusted agency. The certificate contains the name of the certificate holder, a serial number, expiration dates, a copy of the certificate holder's public key (used for encrypting messages and digital signatures) and the digital signature of the certificate-issuing authority (CA) so that a recipient can verify that the certificate is real. This allows others to rely upon signatures or assertions made by the private key that corresponds to the public key that is certified. The CA must be trusted by both the owner of the certificate and the party relying upon the certificate.

challenge-response
When a protected host is disconnected from the DSM, the GuardPoint data is not accessible to users. Challenge-response is a password-based procedure that allows users to gain access to their GuardPoint data during disconnection. Users run a utility, `vmsec challenge`, a seemingly random string (the challenge) is displayed. The user calls this in to their DSM Security administrator. The administrator returns a counter-string (the response) that the host user must enter to decrypt guarded data.

Character device
See ***“raw device.”***

ciphertext
Data in its encrypted form. Ciphertext is the result of encryption performed on plaintext using an algorithm, called a cipher.

cleartext or plaintext
Data in its unencrypted form.

cryptographic algorithm
A computational procedure that takes variable inputs, including a cryptographic key, and produces ciphertext output. Also called a cipher. Examples of cryptographic algorithms include AES, ARIA, and DES.

cryptographic key
See "**encryption key**."

cryptographic signature
See "**signing files**."

Database Encryption Key (DEK)
A key generated by Microsoft SQL when TDE is enabled.

Data Security Manager (DSM)
Sometimes called the *Security Server* or *appliance*. A Vormetric server that acts as the central repository and manager of encryption keys and security policies. Receives instructions and configuration from administrators through a GUI-based interface called the *Management Console*. Passes and receives information to and from VTE Agents. Available as a complete hardened hardware system (*DSM Appliance*) or as software solution installed on a UNIX box (*software-only DSM*).

dataxform
A utility to encrypt data in a directory. Short for "data transform."

DB2
A relational model database server developed by IBM.

Decryption
The process of changing ciphertext into plaintext using a cryptographic algorithm and key.

Digital signature
A cryptographic transformation of data that provides the services of origin authentication, data integrity, and signer non-repudiation.

domains
See *administrative domains*.

Domain Administrator
The second-level DSM administrator created by a DSM *System Administrator*. The DSM *Domain Administrator* creates and assigns DSM *Security Administrators* to domains and assigns them their security "**roles**". See "**DSM Administrator and types**".

Domain and Security Administrator
A hybrid DSM administrator who is has the privileges of a DSM Domain Administrator and Security Administrator.

DSM
See *"**Data Security Manager (DSM)."***

DSM Administrator and types
Specialized system security administrators who can access the Vormetric DSM Management Console. There are five types of DSM administrators:

DSM System Administrator - Creates/removes other DSM administrators of any type, changes their passwords, creates/removes, domains, assigns a Domain Administrator to each domain. Cannot do any security procedures in any domain.

Domain Administrator - Adds/removes DSM Security Administrators to domains, and assign roles to each one. Cannot remove domains and cannot do any of the domain security roles.

Security Administrator - Performs the data protection work specified by their roles. Different roles enable them to create policies, configure hosts, audit data usage patterns, apply GuardPoints, and so on.

Domain and Security Administrator - Can do the tasks of DSM Domain and Security Administrators.

All - Can do the tasks of all three of the DSM administrative types

DSM Automation Utilities
Also called VMSSC. A set of command line utilities that is downloaded and installed separately on the protected host or any networked machine. These utilities can be used by advanced users to automate DSM processes that would normally be done with the Management Console. See the *DSM Automation Reference* for complete details.

DSM CLI
A command line interface executed on the DSM to configure the DSM network and perform other system-level tasks. See the *DSM Command Line Interface* documentation

DSM CLI Administrator
A user who can access the DSM CLI. DSM CLI Administrators are actual system users with real UNIX login accounts. They perform tasks to setup and operate the DSM installation. They do not have access to the Management Console.

DSM database
A database associated with the DMS containing the names of protected hosts, policies, GuardPoints, settings, and so on.

DSM System Administrator
The highest level of DSM administrator. This administrator creates/removes other DSM administrators of any type, creates/removes domains, and assigns a Domain Administrator to each domain. The DSM System Administrator cannot perform any security procedures in any domain or system. This administrator is not related to computer or network system administrators.

EKM
See "**Extensible Key Management (EKM)**."

Encryption
The process of changing plaintext into ciphertext using a cryptographic algorithm and key.

encryption agent
See *Vormetric Transparent Encryption agent*.

encryption key
A piece of information used in conjunction with a cryptographic algorithm that transforms plaintext into ciphertext, or vice versa during decryption. Can also be used to encrypt digital signatures or encryption keys themselves. An entity with knowledge of the key can reproduce or reverse the operation, while an entity without knowledge of the key cannot. Any VDS policy that encrypts GuardPoint data requires an encryption key.

Extensible Key Management (EKM)
An API library specification provided by Microsoft that defines a software framework that allows hardware security module (HSM) providers to integrate their product with the Microsoft SQL Server.

failover DSM
A secondary DSM that assumes the policy and key management load when a protected host cannot connect to the primary DSM or when a protected host is specifically assigned to the failover DSM. A failover DSM is almost identical to the primary DSM, having the same keys, policies, protected hosts, and so on.

FF1
See "Format Preserving Encryption (FPE)".

FF3
See "Format Preserving Encryption (FPE)".

file signing
See *signing files*.

File Key Encryption Key (FKEK)
The key used to encrypt the file encryption key that is used to encrypt on-disk data, also known as a wrapper key.

FKEK
See "File Key Encryption Key (FKEK)"

File System Agent
A Vormetric software agent that resides on a host machine and allows administrators to control encryption of, and access to, the files, directories and executables on that host system. For example, administrators can restrict access to specific files and directories to specific users at specific times using specific executables. Files and directories can be fully encrypted, while the file metadata (for example, the file names) remain in cleartext. Also called the "**VTE Agent**".

Format Preserving Encryption (FPE)
An encryption algorithm that preserves both the formatting and length of the data being encrypted. Examples of such algorithms used by Vormetric include FF1 and FF3, both of which are approved by NIST. Vormetric's **FPE tokenization format** uses the FF3 algorithm.

FQDN
Fully qualified domain name. A domain name that specifies its exact location in the tree hierarchy of the Domain Name Server (DNS). For example: `example.vormetric.com`.

GPFS
General Parallel File System is a high-performance shared-disk clustered file system developed by IBM.

GuardPoint
A location in the file system hierarchy, usually a directory, where everything underneath has a Vormetric data protection policy applied to it. The File System Agent intercepts any attempt to access anything in the GuardPoint and uses policies obtained from the DSM to grant or deny the access attempt. Usually, depending on the policies, data copied into a GuardPoint is encrypted, and only authorized users can decrypt and use that GuardPoint data.

Hardware Security Module or HSM
A tamper-resistant hardware device that stores keys and provides stringent access control. It also provides a random number generator to generate keys. The DSM Appliance can come with an embedded Hardware Security Module.

host locks
Two Management Console options, **FS Agent Locked** and **System Locked,** that are used to protect the File System Agent and certain system files. File System Agent protection includes preventing some changes to the File System Agent installation directory and preventing the unauthorized termination of File System Agent processes.

host password
This is not a regular login or user password. This is the password entered by a host system user to unlock a GuardPoint when there is no DSM connection. This password decrypts cached keys when the DSM is not accessible. The host must also be configured with **Cached on Host** keys. See "**challenge-response**".

initial test policy
A first data security policy applied to a GuardPoint that is used to gather directory access information so DSM Security Administrators can create a permanent operational policy. The initial test policy encrypts all data written into the GuardPoint; decrypts GuardPoint data for any user who access it; audits and creates log messages for every GuardPoint access; reduces log message "noise" so you can analyze the messages that are important to you for tuning this policy; is run in the "**Learn Mode**" which does not actually deny user access, but allows you to record GuardPoint accesses.

After enough data is collected, the DSM Security Administrator can modify the initial test policy into an operational policy.

Key Agent
A Vormetric agent that provides an API library supporting a subset of the PKCS#11 standard for key management and cryptographic operations. It is required for the following products: Vormetric Key Management (VKM), Vormetric Tokenization, Vormetric Application Encryption (VAE), Vormetric Cloud Encryption Gateway (VCEG). Sometimes called the *VAE Agent*.

key group
A key group is a collection of asymmetric keys that are applied as a single unit to a policy.

key management
The management of cryptographic keys and other related security objects (for example, passwords) during their entire life cycle, including their generation, storage, establishment, entry and output, and destruction.

key template
A template that lets you quickly add agent keys or third-party vault keys by specifying a template with predefined attributes. You can define specific attributes in a template, then you can call up the template to add a key with those attributes.

key shares
When data is backed up or exported from VTE (for example, symmetric keys or DSM database backups), they can be encrypted in a wrapper key needed to restore the exported data on the new machine. Wrapper keys can be split and distributed to multiple individuals. Each split piece of the wrapper key is called a *key share*. Decrypting the data requires that some specified number of the individuals that received key shares contribute their key share to decrypt the data.

key wrapping
A class of symmetric encryption algorithms designed to encapsulate (encrypt) cryptographic key material. The key wrap algorithms are intended for applications such as protecting keys while in untrusted storage or transmitting keys over untrusted communications networks. Wrapper keys can be broken up into *key shares*, which are pieces of a wrapper key. Key shares are divided amongst two or more *custodians* such that each custodian must contribute their key share in order to assemble a complete wrapper key.

Key Vault
A Vormetric product that provides passive key vaulting. It securely stores symmetric and asymmetric encryption keys from any application and tracks key expiration dates.

KMIP
Key Management Interoperability Protocol. A protocol for communication between enterprise key management systems and encryption systems. A KMIP-enabled device or client software can communicate with the DSM to manage encrypted keys.

Learn Mode
A DSM operational mode in which all actions that would have been denied are instead permitted. This permits a policy to be tested without actually denying access to resources. In the Learn Mode, all GuardPoint access attempts that would have been denied are instead permitted. These GuardPoint accesses are logged to assist in tuning and troubleshooting policies.

Live Data Transformation (LDT)
A separately licensed feature of Vormetric Transparent Encryption (VTE) that allows you to transform (encrypt or decrypt) or rekey GuardPoint data without blocking use or application access to that data.

local domain
A DSM domain in which DSM administration is restricted to Domain Administrators or Security Administrators assigned to that domain. To access a local domain in the Management Console, a DSM administrator must specify their local domain upon login.

Management Console
The graphical user interface (GUI) to the DSM.

Master encryption key (MEK)
The encryption key for Oracle Database used to encrypt secondary data encryption keys used for column encryption and tablespace encryption. Master encryption keys are part of the Oracle Advanced Security Transparent Data Encryption (TDE) two-tier key architecture.

MEK
See *Master encryption key.*

Microsoft SQL Server
A relational database server, developed by Microsoft.

Microsoft SQL Transparent Data Encryption (MS-SQL TDE)
Microsoft SQL Server native encryption for columns and tables.

multi-factor authentication
An authentication algorithm that requires at least two of the three following authentication factors:
1) something the user knows (for example, password); 2) something the user has (example: RSA SecurID); and 3) something the user is (example: fingerprint). VTE implements an optional form of multi-factor authentication for Management Console users by requiring DSM administrators to enter the token code displayed on an RSA SecurID, along with the administrator name each time the administrator logs on to the Management Console.

multitenancy
A VTE feature that enables the creation of multiple local domains within a single DSM. A local domain is a DSM domain in which DSM administration is restricted to Domain Administrators or Security Administrators assigned to that domain. This allows Cloud Service Providers to provide their customers with VTE administrative domains over which the customer has total control of data security. No other administrators, including CSP administrators, have access to VTE security in a local domain.

offline policy
Policies for Database Backup Agents. *Online policies* are for the File System Agent.

one-way communication
A VTE feature for an environment where the DSM cannot establish a connection to the agent, but the agent can establish a connection to the DSM. For example, the protected host is behind a NAT so protected host ports are not directly visible from the DSM, or the protected host is behind a firewall that prohibits incoming connections, or the protected host does not have a fixed IP address as in the cloud. When an agent is registered with one-way communication, changes made for that protected host on the DSM are not pushed to the protected host, rather as the protected host polls the DSM it will retrieve the change.

online policies
Policies for the File System Agent. *Offline policies* are for Database Backup Agents.

policy
A set of security access and encryption rules that specify who can access which files with what executable during what times, and whether or not those files are encrypted. Policies are created by DSM Security Administrators, stored in the DSM, and implemented on protected hosts by a File system Agent. See “**rule (for policies)**”.

policy tuning
The process of creating a simple Learn Mode policy that allows any protected host user to access a GuardPoint; to examine who accesses the GuardPoint, what executables they use, and what actions they require; and to modify the policy such that it allows the right people, using the right executable, performing the right action to do their job, and prevent anyone else from inappropriate access.

process set
A list of processes that can be used by the users in a user set associated with a policy rule.

protected host
A host on which a VTE Agent is installed to protect that host's data.

public key cryptographic algorithm, public key infrastructure
A cryptographic system requiring two keys, one to lock or encrypt the plaintext, and one to unlock or decrypt the ciphertext. Neither key can do both functions. One key is published (*public key*) and the other is kept private (*private key*). If the lock/encryption key is the one published, the system enables private communication from the public to the unlocking key's owner. If the unlock/decryption key is the one published, then the system serves as a signature verifier of documents locked by the owner of the private key. Also called asymmetric key cryptography.

raw device
A type of block device that performs input/output operations without caching or buffering. This results in more direct access.

register host
The process of enabling communication between a protected host and the DSM. Registration happens during agent installation. Before registration can happen, the host must be added to the DSM database.

rekeying
The process of changing the encryption keys used to encrypt data. Changing keys enhances data security and is a requirement to maintain compliance with some data security guidelines and regulations. Also called *key rotation*.

roles
A set of Management Console permissions assigned to DSM Security Administrators by DSM Domain Administrators. There are five roles: *Audit* (can generate and view logging data for file accesses), *key* (can create, edit, and delete keys), *Policy* (can create, edit, and delete policies), *Host* (can configure, modify, and delete protected hosts and protected host groups), and *Challenge & Response* (can generate a temporary password to give to a protected host user to decrypt cached encryption keys when connection to the DSM is broken).

RSA SecurID
A hardware authentication token that is assigned to a computer user and that generates an authentication code at fixed intervals (usually 60 seconds). In addition to entering a static password, Management Console administrators can be required to input an 8-digit number that is provided by an external electronic device or software.

rule (for policies)
Every time a user or application tries to access a GuardPoint file, the access attempt passes through each rule of the policy until it finds a rule where all the criteria are met. When a rule matches, the *effect* associated with that rule is enforced. A rule consists of five access criteria and an effect. The criteria are Resource (the file/directories accessed), User (the user or groups attempting access), Process (the executable used to access the data), When (the time range when access is attempted) and Action (the type of action attempted on the data, for example read. write, rename and so on). *Effect* can be permit or deny access, decrypt data access, and audit access attempt. See *policy*.

secfs
1) The File System Agent initialization script. 2) An acronym for Vormetric Secure File System agent. It generally refers to the kernel module that handles policies (locks, protected host settings, logging preferences) and keys, and enforces data security protection.

secvm
A proprietary device driver that supports GuardPoint protection to raw devices. `secvm` is inserted in between the device driver and the device itself.

Security Administrator
The third-level DSM administrator who does most of data protection work like creating policies, configuring protected hosts, auditing data usage patterns, applying GuardPoints and other duties. The privileges of each Security Administrator is specified by the roles assigned to them by the Domain Administrator. See *roles*. See "**DSM Administrator and types**".

Security Server
See "**DSM**".

separation of duties
A method of increasing data security by creating customized DSM administrator roles for individual DSM administrators such that no one administrator has complete access to all encryption keys in all domains of all files.

signing files
File signing is a method that VTE uses to check the integrity of executables and applications before they are allowed to access GuardPoint data. If file signing is initiated in the Management Console, the File System Agent calculates the cryptographic signatures of the executables that are eligible to access GuardPoint data. A tampered executable, such as a Trojan application, malicious code, or

rogue process, with a missing or mismatched signature, is denied access. Also called *cryptographic signatures*.

Suite B mode
A set of publicly available cryptographic algorithms approved by the United States National Security Agency (NSA). These algorithms enhance security by adding up to 384-bit encryption to the communication between the Web browser and the DSM, the DSM and Agent, and between DSMs in HA environments.

Symmetric-key algorithm
Cryptographic algorithms that use the same cryptographic keys for both encryption of plaintext and decryption of ciphertext. The keys, in practice, represent a shared secret between two or more parties that can be used to maintain a private information link. This requirement that both parties have access to the secret key is one of the main drawbacks of symmetric key encryption, in comparison to public-key encryption.

System Administrator (DSM)
See "**DSM Administrator and types**".

Transparent Data Encryption (TDE)
A technology used by both Microsoft and Oracle to encrypt database content. TDE offers encryption at a column, table, and tablespace level. TDE solves the problem of protecting data at rest, encrypting databases both on the hard drive and consequently on backup media.

user set
A named list of users on which a policy rule applies.

VAE Agent
See "**Key Agent**".

VDE Agent
Vormetric agent installed on a protected host to implement disk encryption. See *Vormetric Disk Encryption (VDE)*.

vmd
Acronym for Vormetric Daemon, vmd is a process that supports communication between the DSM and kernel module.

VMSSC or Vormetric Security Server Command Line Interface
See *DSM Automation Utilities*.

Vormetric Application Encryption (VAE)
A product that enables data encryption at the application level as opposed to the file level as is done with VTE.

Where VTE encrypts a file or directory, VAE can encrypt a column in a database or a field in an application. VAE is essentially an API library for key management and cryptographic operations based on PKCS#11. See the *Vormetric Application Encryption Installation and API Reference Guide*.

Vormetric Cloud Encryption Gateway (VCEG)
Vormetric product that safeguards files in cloud storage environments, including Amazon Simple Storage Service (Amazon S3) and Box. The cloud security gateway solution encrypts sensitive data before it is saved to the cloud storage environment, then decrypts data for approved users when it is removed from the cloud.

Vormetric Data Security Platform or VDS Platform
The technology platform upon which all other Vormetric products—Vormetric Transparent Encryption (VTE), Vormetric Application Encryption (VAE), Vormetric Key Management (VKM), Vormetric Cloud Encryption Gateway (VCEG), Vormetric Tokenization Server (VTS), Vormetric Key Management (VKM), and Vormetric Protection for Teradata Database—are based.

Vormetric Encryption Expert or VEE
Earlier name of the Vormetric Transparent Encryption (VTE) product. It may sometimes appear in the product GUI or installation scripts.

Vormetric Key Management (VKM)
Vormetric product that provides a standards-based platform for storing and managing encryption keys and certificates from disparate sources across the enterprise. This includes Vormetric encryption keys, 3rd-party software keys, KMIP device keys and so on.

Vormetric Protection for Teradata Database
Vormetric product that secures sensitive data in the Teradata environment.

Vormetric Security Intelligence
Vormetric product that provides support for Security Information and Event Management (SIEM) products such as ArcSight, Splunk and QRadar. Provides solutions that monitor real-time events and analyze long-term data to find anomalous usage patterns, qualify possible threats to reduce false positives, and alert organizations when needed. Documented in the VDS Platform Security Intelligence User Guide.

Vormetric Tokenization Server (VTS)
Vormetric product that replaces sensitive data in your database (up to 512 bytes) with unique identification symbols called tokens. Tokens retain the format of the original data while protecting it from theft or compromise.

Vormetric Transparent Encryption or VTE
Vormetric product that protects data-at-rest. Secures any database, file, or volume without changing the applications, infrastructure or user experience.

Vormetric Vault
A virtual vault to store 3rd-party encryption keys, certificates and other security objects.

VTE Agent
Vormetric agents that are installed on protected hosts to implement data protection. See “**File System Agent**”.

wrapper keys
See “**key wrapping**”.

WSDL
Web Services Description Language.