Thales

CipherTrust Transparent Encryption Cryptographic Module

Table of Contents

1 General 5
1.1 Overview 5
1.2 Security Levels 5
2 Cryptographic Module Specification 6
2.1 Description 6
2.2 Tested and Vendor Affirmed Module Version and Identification 7
2.3 Excluded Components 8
2.4 Modes of Operation 8
2.5 Algorithms 9
2.6 Security Function Implementations 10
2.7 Algorithm Specific Information 10
2.8 RBG and Entropy 10
2.9 Key Generation 10
2.10 Key Establishment 11
2.11 Industry Protocols 11
3 Cryptographic Module Interfaces 12
3.1 Ports and Interfaces 12
4 Roles, Services, and Authentication 13
4.1 Authentication Methods 13
4.2 Roles 13
4.3 Approved Services 13
4.4 Non-Approved Services 15
4.5 External Software/Firmware Loaded 15
5 Software/Firmware Security 16
5.1 Integrity Techniques 16
5.2 Initiate on Demand 16
6 Operational Environment 17
6.1 Operational Environment Type and Requirements 17
7 Physical Security 18
8 Non-Invasive Security 19
9 Sensitive Security Parameters Management 20
9.1 Storage Areas 20
9.2 SSP Input-Output Methods 20
9.3 SSP Zeroization Methods 20
9.4 SSPs 20
9.5 Additional Information 21
10 Self-Tests 22

10.1 Pre-Operational Self-Tests 22
10.2 Conditional Self-Tests 22
10.3 Periodic Self-Test Information 23
10.4 Error States 24
11 Life-Cycle Assurance 25
11.1 Installation, Initialization, and Startup Procedures 25
11.2 Administrator Guidance 25
11.3 Non-Administrator Guidance 25
11.4 End of Life 25
12 Mitigation of Other Attacks 26

List of Tables

Table 1: Security Levels 5
Table 2: Tested Module Identification – Software, Firmware, Hybrid (Executable Code Sets) 7
Table 3: Tested Module Identification – Hybrid Disjoint Hardware 7
Table 4: Tested Operational Environments - Software, Firmware, Hybrid 8
Table 5: Modes List and Description 8
Table 6: Approved Algorithms 9
Table 7: Non-Approved, Not Allowed Algorithms 9
Table 8: Security Function Implementations 10
Table 9: Ports and Interfaces 12
Table 10: Roles 13
Table 11: Approved Services 14
Table 12: Non-Approved Services 15
Table 13: Storage Areas 20
Table 14: SSP Input-Output Methods 20
Table 15: SSP Zeroization Methods 20
Table 16: SSP Table 1 20
Table 17: SSP Table 2 21
Table 18: Pre-Operational Self-Tests 22
Table 19: Conditional Self-Tests 22
Table 20: Pre-Operational Periodic Information 23
Table 21: Conditional Periodic Information 24
Table 22: Error States 24

List of Figures

Figure 1: Block Diagram 7
Figure 2: Cryptographic Hardware for Intel Processor 18
Figure 3: Intel® Xeon® Gold 5318N as found in PowerEdge R750xs 18
Figure 4: Cryptographic Hardware for IBM Processor 18
Figure 5: IBM Power9 Processor as found in IBM Power9 Server 18

1 General

1.1 Overview

This document is the non-proprietary FIPS 140-3 Security Policy for the CipherTrust Transparent Encryption Cryptographic Module Version 3.0.1. This Security Policy describes the security services provided by the module and describes how the module meets the requirements of FIPS 140-3 (Federal Information Processing Standards 140-3) for an overall Security Level 1 implementation.

1.2 Security Levels

Section	Security Level
1	1
2	1
3	1
4	1
5	1
6	1
7	1
8	N/A
9	1
10	1
11	1
12	N/A

Table 1: Security Levels

2 Cryptographic Module Specification

2.1 Description

Purpose and Use:

The CipherTrust Transparent Encryption Cryptographic Module is a component within the CipherTrust Transparent Encryption solution, which in turn is part of the CipherTrust Data Security Platform solution. This solution provides data protection and access control functionality and is intended to be used as part of the CipherTrust Transparent Encryption solution.

The CipherTrust Transparent Encryption Cryptographic Module is a loadable kernel module for Windows, Linux and IBM AIX. It is also loadable in the user space for Linux. The module communicates with the SecFS, a secure layer that sits over the filesystem to secure files and directories and to enforce the access and encryption policy. The policy specifies the key to be used by the cryptographic module when writing data to or reading data from the disk. The CipherTrust Transparent Encryption Cryptographic Library provides the cryptographic algorithms used by the module.

Module Type: Software-hybrid

Module Embodiment: MultiChipStand

Cryptographic Boundary:

The CipherTrust Transparent Encryption Cryptographic Module includes the CipherTrust Transparent Encryption Cryptographic Library. Cryptographic keys are provided to the library by other modules or applications and are used within the library.

The cryptographic boundary of the module is the software library and CPU as shown in Figure 1. The diagram shows plaintext and ciphertext for encryption and decryption operations, and the request and response for PBKDF requests.

Tested Operational Environment's Physical Perimeter (TOEPP):

Figure 1 shows the location of the cryptographic module with respect to the TOEPP. The TOEPP includes the CipherTrust Transparent Encryption application.

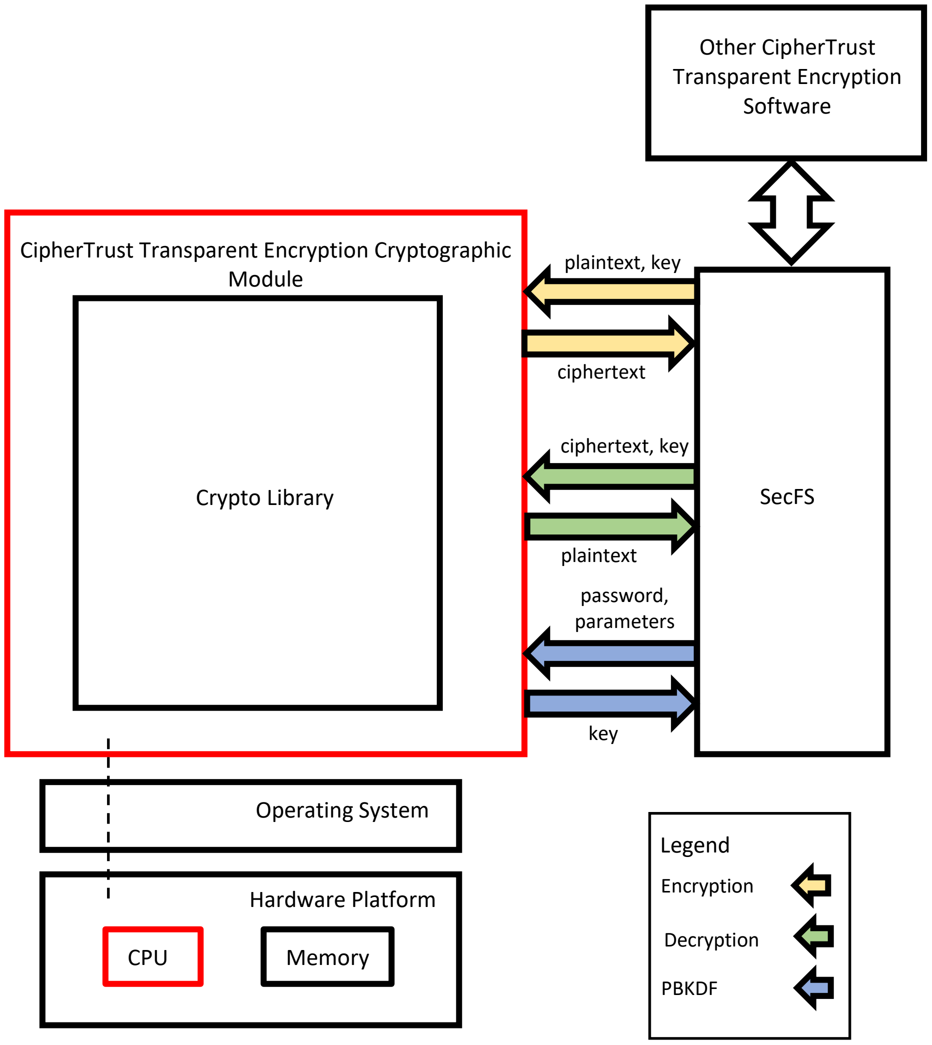

Figure 1: Block Diagram

2.2 Tested and Vendor Affirmed Module Version and Identification

Tested Module Identification – Software, Firmware, Hybrid (Executable Code Sets):

Package or File Name	Software/ Firmware Version	Features	Integrity Test
seccrypto.ko (kernel space)	3.0.1		HMAC-SHA2-384
libvorcrypto.so (user space)	3.0.1		HMAC-SHA2-384
vmcrypto.sys	3.0.1		HMAC-SHA2-384
seccrypto	3.0.1		HMAC-SHA2-384

Table 2: Tested Module Identification – Software, Firmware, Hybrid (Executable Code Sets)

Tested Module Identification – Hybrid Disjoint Hardware:

Model and/or Part Number	Hardware Version	Firmware Version	Processors	Features
-	Intel® Xeon® Gold 5318N			
-	IBM Power9			

Table 3: Tested Module Identification – Hybrid Disjoint Hardware

Tested Operational Environments - Software, Firmware, Hybrid:

Operating System	Hardware Platform	Processors	PAA/PAI	Hypervisor or Host OS	Version(s)
Red Hat Enterprise Linux 8.6 (Kernel Space)	PowerEdge R750xs	Intel® Xeon® Gold 5318N	Yes	VMware ESXi 7.0 Update 3	3.0.1
Red Hat Enterprise Linux 8.6 (User Space)	PowerEdge R750xs	Intel® Xeon® Gold 5318N	Yes	VMware ESXi 7.0 Update 3	3.0.1
Windows Server 2019	PowerEdge R750xs	Intel® Xeon® Gold 5318N	Yes	VMware ESXi 7.0 Update 3	3.0.1
AIX 7.3	IBM Power9 Server (9009-42A)	IBM Power9	Yes		3.0.1

Table 4: Tested Operational Environments - Software, Firmware, Hybrid

Vendor-Affirmed Operational Environments - Software, Firmware, Hybrid:

N/A for this module.

2.3 Excluded Components

None.

2.4 Modes of Operation

Modes List and Description:

Mode Name	Description	Type	Status Indicator
Approved Mode	The module is operating in the Approved mode	Approved	From Service (is_fips=1)
Non-Approved Mode	The module is operating in the non-Approved mode.	Non-Approved	From Service (is_fips=0 or no indicator)

Table 5: Modes List and Description

The module is acting in an approved mode of operation when the module provides the services described in the Approved Services table. These services use the security functions listed in the Approved Algorithm table in an approved manner. The security service indicator provides confirmation that an approved service has been provided.

When a service listed in the Non-Approved Services table is used, the module is not acting in an approved mode of operation.

2.5 Algorithms

Approved Algorithms:

Algorithm	CAVP Cert	Properties	Reference
AES-CBC	A3609	Direction - Decrypt, Encrypt Key Length - 128, 256	SP 800-38A
AES-CBC-CS1	A3609	Direction - decrypt, encrypt Key Length - 128, 256	SP 800-38A
AES-XTS Testing Revision 2.0	A3609	Direction - Decrypt, Encrypt Key Length - 256	SP 800-38E
HMAC-SHA2-256	A3609	Key Length - Key Length: 256-448 Increment 8	FIPS 198-1
HMAC-SHA2-384	A3609	Key Length - Key Length: 256-448 Increment 8	FIPS 198-1
PBKDF	A3609	Iteration Count - Iteration Count: 90000-120000 Increment 1000 Password Length - Password Length: 96-128 Increment 1	SP 800-132
SHA2-256	A3609	-	FIPS 180-4
SHA2-384	A3609	-	FIPS 180-4

Table 6: Approved Algorithms

Vendor-Affirmed Algorithms:

N/A for this module.

There are no vendor-affirmed algorithms in this module.

Non-Approved, Allowed Algorithms:

N/A for this module.

There are no non-approved, allowed algorithms in this module.

Non-Approved, Allowed Algorithms with No Security Claimed:

N/A for this module.

There are no non-approved, allowed algorithms with no security claimed in this module.

Non-Approved, Not Allowed Algorithms:

Name	Use and Function
Diffie-Hellman Key Agreement	Used to create keys used to obfuscate the communications between the kernel and the user space.
RSA	Used for key wrapping as part of the key exchange protocol between the key manager and the CipherTrust Transparent Encryption agent.
ARIA	Used for data encryption. Encryption operations using ARIA do not provide the FIPS indicator in the return.

Table 7: Non-Approved, Not Allowed Algorithms

Non-approved algorithms are not used in an approved mode of operation. An approved mode of operation is used when using an approved service (listed in the Approved Services Table with an approved algorithm listed in the Approved Algorithms table). The return code 'is_fips=1' indicates that the cryptographic operation has been performed in an approved mode of operation. If the return code is absent, or is 'is_fips=0', the service was not performed in an approved mode of operation.

2.6 Security Function Implementations

Name	Type	Description	Properties	Algorithms
Encrypt/Decrypt 1	BC-UnAuth	Encryption/ Decryption of data		AES-CBC Key Length: 128
Encrypt/Decrypt 2	BC-UnAuth	Encryption/ Decryption of data		AES-CBC Key Length: 256
Encrypt/Decrypt 3	BC-UnAuth	Encryption/ Decryption of data		AES-CBC-CS1 Key Length: 128
Encrypt/Decrypt 4	BC-UnAuth	Encryption/ Decryption of data		AES-CBC-CS1 Key Length: 256
Encrypt/Decrypt 5	BC-UnAuth	Encryption/ Decryption of data		AES-XTS Testing Revision 2.0 Key Length: 256
Create MAC 1	MAC	Creates an HMAC		HMAC-SHA2-256 Key Length: 256
Create MAC 2	MAC	Creates an HMAC		HMAC-SHA2-384 Key Length: 384
Verify Software Integrity	MAC	Verifies an HMAC		HMAC-SHA2-384 Key Length: 384
Create Hash 1	SHA	Creates a hash		SHA2-256 Hash size: 256
Create Hash 2	SHA	Creates a hash		SHA2-384 Hash size: 384
PBKDF	PBKDF	Derives a key		PBKDF

Table 8: Security Function Implementations

2.7 Algorithm Specific Information

PBKDF is implemented in accordance with SP 800-132, option 1a. The module includes only the functionality to receive the input to the PBKDF function (password, password length, salt, salt length, iterations, key length) and provide the output (key). The PBKDF function has been tested based on the minimum and maximums noted in the Approved Algorithms table. The minimum recommended password length is 96 bytes. This is consistent with a 96-character password. SP 800-132, Appendix A recommends a minimum password length of 10 characters. This range exceeds the recommendation. Assuming that the password is made up of upper-case letters, lower case letters and numeric characters, the probability of guessing a random 96-character password in a single attempt is one in 62^{96}. The lower bound of 90,000 iterations is deemed sufficient since the derived key never leaves the CipherTrust Transparent Encryption Agent (the Agent). The derived key is used to protect configuration files and keys used by the Agent.

Note that in accordance with SP800-132, keys derived from passwords using PBKDF may only be used in storage applications. Although usage of keys resulting from use of the PBKDF function is outside of the scope of this cryptographic module, it should be noted that the Agent in which these keys are used provides data-at-rest encryption for storage applications.

2.8 RBG and Entropy

N/A for this module.

This module does not include an entropy source.

2.9 Key Generation

2.10 Key Establishment

This module does not provide key establishment functions in the approved mode of operation.

2.11 Industry Protocols

Not applicable.

3 Cryptographic Module Interfaces

3.1 Ports and Interfaces

Physical Port	**Logical Interface(s)**	**Data That Passes**
N/A	Data Input	Application Programming Interface (API) entry point data input stack parameters
N/A	Data Output	API entry point data output stack parameters
N/A	Control Input	API entry point and corresponding stack parameters
N/A	Control Output	API entry point return values and stack parameters
N/A	Status Output	API entry point return values and status stack parameters

Table 9: Ports and Interfaces

4 Roles, Services, and Authentication

4.1 Authentication Methods

N/A for this module.

4.2 Roles

Name	Type	Operator Type	Authentication Methods
Crypto Officer	Role	Crypto Officer	None

Table 10: Roles

The Crypto Officer role is implicitly assumed by the entity that can access the interfaces to the cryptographic module. This entity accesses the module exclusively via API calls. Each process or thread accessing the module is logically separated by the operating system into independent contexts of execution.

4.3 Approved Services

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
Encrypt Data	Service to encrypt supplied data.	is_fips=1	• key • IV (16 bytes) • flag to indicate encrypt • data to be encrypted • data length • data segment index number • data segment size	• encrypted data • return code	Encrypt/Decrypt 1 Encrypt/Decrypt 2 Encrypt/Decrypt 3 Encrypt/Decrypt 4 Encrypt/Decrypt 5	Crypto Officer - Data Encryption Key: W,E
Decrypt Data	Service to decrypt supplied data.	is_fips=1	• key • IV (16 bytes) • flag to indicate decrypt • data to be decrypted • data length • data segment index number • data segment size	• decrypted data • return code	Encrypt/Decrypt 1 Encrypt/Decrypt 2 Encrypt/Decrypt 3 Encrypt/Decrypt 4 Encrypt/Decrypt 5	Crypto Officer - Data Encryption Key: W,E
MAC Generate	Service to generate a MAC over supplied data.	is_fips=1	• data • HMAC Key • key length • message buffer • message length • SHA type (length)	• message digest • message digest length	Create MAC 1 Create MAC 2	Crypto Officer - HMAC Key: W,E
Verify Software Integrity	Service to verify the integrity of	is_fips=1	• software	• message digest • message digest length	Verify Software Integrity	Crypto Officer

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
	the module software		• Software Integrity HMAC Key • key length • message buffer • message length • SHA type (length) • known MAC	• Success or Failure		
Calculate Hash	Service to create a hash over supplied data.	is_fips=1	• supplied data • data length • hash type (256, 384)	• hash result	Create Hash 1 Create Hash 2	Crypto Officer
Key Derivation	Password-based key derivation function (PBKDF)	is_fips=1	• password • password length • salt • salt length • iteration number • derived key length	• Derived Key	PBKDF	Crypto Officer - Derived Key: R
Key Destroy	Service to zeroize keys.	is_fips=1	Command	Status	None	Crypto Officer - HMAC Key: Z - Data Encryption Key: Z - Derived Key: Z
Show Module Version	Shows the module version.	N/A	const char *vds_crypto_version(void)	Module version	None	Crypto Officer
Shows Status	Shows the module status.	N/A	Command	Module status	None	Crypto Officer
Perform Self-Tests	Performs module self-tests via module restart.	N/A	Command	Status	None	Crypto Officer

Table 11: Approved Services

4.4 Non-Approved Services

Name	Description	Algorithms	Role
DH Key Agreement	Used to create a key, which is then provided to SecFS for use outside of the module.	Diffie-Hellman Key Agreement	Crypto Officer
RSA Key Wrap	Used to encrypt a provided key and pass the result to SecFS.	RSA	Crypto Officer
Non-Approved Encrypt Data	Used to encrypt supplied data.	ARIA	Crypto Officer
Non-Approved Decrypt Data	Used to decrypt supplied data.	ARIA	Crypto Officer

Table 12: Non-Approved Services

4.5 External Software/Firmware Loaded

Not applicable.

5 Software/Firmware Security

5.1 Integrity Techniques

The module checks the integrity of its object code when it is initialized. The module performs an HMAC-SHA2-384 of itself when it is loaded into the kernel or user space. Following the HMAC Known Answer Test (KAT), this HMAC value is compared to the HMAC-SHA2-384 digest generated during build time. If the results are not the same, an error message is written to the output interface, and the module ceases operation. The HMAC key used for this function is embedded in the module in plaintext.

At the completion of this test, temporary values are zeroized.

5.2 Initiate on Demand

Restarting the CipherTrust Transparent Encryption Agent will cause the integrity check to be rerun.

6 Operational Environment

6.1 Operational Environment Type and Requirements

Type of Operational Environment: Modifiable

How Requirements are Satisfied:

The CipherTrust Transparent Encryption Cryptographic Module exists as software executing in a commercially available operating system. The operating system (Windows, Linux or AIX) provides process isolation and CPU scheduling to ensure that programs do not interfere with each other. This ensures that the cryptographic module has control over its own SSPs while the cryptographic process is in use.

Within the cryptographic module, each binary is launched into the address space of a process. Each instance of the module is run strictly inside the process space of the module. All processes spawned by the cryptographic module are owned by the module.

The single operator for a given instance of the module is the combined identities associated with the processes containing the module. The operating system and hardware enforce the process isolation including memory isolation, where keys and intermediate key data are temporarily stored.

When running in User Space, the writable memory areas of the module are accessible only to the module. Data and stack segments are accessible only to the processes containing the module.

When running in Kernel space, access is restricted to only kernel modules. It is the user's responsibility to allow only trusted modules to be loaded into the kernel.
The operating system is responsible for multitasking operations so that other processes cannot access the address space of the processes containing the module.

The module does not require a specific configuration of the target operating systems to enforce the controls described above.

7 Physical Security

The module is a software-hybrid module that operates on a multi-chip standalone platform, which conforms to the level 1 requirements for physical security.

Figure 2: Cryptographic Hardware for Intel Processor

Figure 3: Intel® Xeon® Gold 5318N as found in PowerEdge R750xs

Figure 4: Cryptographic Hardware for IBM Processor

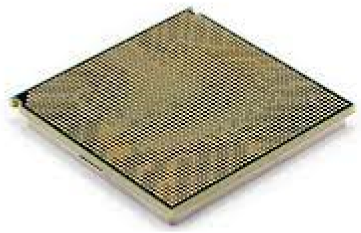

Figure 5: IBM Power9 Processor as found in IBM Power9 Server

8 Non-Invasive Security

Not applicable. The module does not claim Non-Invasive Security.

9 Sensitive Security Parameters Management

9.1 Storage Areas

Storage Area Name	Description	Persistence Type
Memory	Working memory used by the module	Dynamic

Table 13: Storage Areas

9.2 SSP Input-Output Methods

Name	From	To	Format Type	Distribution Type	Entry Type	SFI or Algorithm
Key Import	SecFS	CipherTrust Transparent Encryption Cryptographic Module	Plaintext	Automated	Electronic	
Key Export	CipherTrust Transparent Encryption Cryptographic Module	SecFS	Plaintext	Automated	Electronic	

Table 14: SSP Input-Output Methods

9.3 SSP Zeroization Methods

Zeroization Method	Description	Rationale	Operator Initiation
Key Destroy	Keys are overwritten with zeros.	Keys are not stored by the module. They are zeroized at the end of each cryptographic operation.	N/A. Keys are zeroized at the end of each cryptographic operation.

Table 15: SSP Zeroization Methods

9.4 SSPs

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
HMAC Key	Key used with MAC Generate.	256 bit, 384 bit - 256 bit, 384 bit	HMAC Key - CSP			
Data Encryption Key	AES key used for Encrypt Data and Decrypt Data services.	128, 256 bit AES-CBC 256 bit AES-XTS 128, 256 bit AES-CBC-CS1 - 128, 256 bit	Symmetric Key - CSP			
Derived Key	Used outside of the module to encrypt configuration and other information used by a storage application.	384 bit - 384 bit	Symmetric Key - CSP	PBKDF		

Table 16: SSP Table 1

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
HMAC Key	Key Import	Memory:Plaintext	Until the end of the cryptographic operation.	Key Destroy	
Data Encryption Key	Key Import	Memory:Plaintext	Until the end of the cryptographic operation.	Key Destroy	
Derived Key	Key Export			Key Destroy	

Table 17: SSP Table 2

9.5 Additional Information

Keys are obtained from an external FIPS-validated module. No entropy input is claimed for the CipherTrust Transparent Encryption Cryptographic Module.

10 Self-Tests

10.1 Pre-Operational Self-Tests

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details
HMAC-SHA2-384 (A3609)	Key length: 384 bits	Integrity Test	SW/FW Integrity	Logged: "FIPS <testname> Test failed" or "FIPS Tests passed"	Keyed Hash

Table 18: Pre-Operational Self-Tests

The module performs a pre-operational self-test to verify the integrity of the software. On power-up, the module runs the conditional self-tests (including the KAT for HMAC-SHA2-384) and then runs the HMAC-SHA2-384 integrity test.

Data input/output and any data processing are inhibited while the test is in progress. When all of the pre-operational tests have run to completion, one or more messages are written to the log. If all tests pass, a single "FIPS Tests passed" message is written to the log. When all tests pass, the module is capable of being operated in an approved mode of operation.

10.2 Conditional Self-Tests

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
AES-CBC (A3609)	Key Length: 128 bits	KAT	CAST	Logged: "FIPS <testname> Test failed" or "FIPS Tests passed"	Encrypt, Decrypt	Power-up
AES-CBC (A3609)	Key Length: 256 bits	KAT	CAST	Logged: "FIPS <testname> Test failed" or "FIPS Tests passed"	Encrypt, Decrypt	Power-up
AES-CBC-CS1 (A3609)	Key Length: 128 bits	KAT	CAST	Logged: "FIPS <testname> Test failed" or "FIPS Tests passed"	Encrypt, Decrypt	Power-up
AES-XTS Testing Revision 2.0 (A3609)	Key Length: 256 bits	KAT	CAST	Logged: "FIPS <testname> Test failed" or "FIPS Tests passed"	Encrypt, Decrypt	Power-up
HMAC-SHA2-256 (A3609)	Key length: 256 bits	KAT	CAST	Logged: "FIPS <testname> Test failed" or "FIPS Tests passed"	Keyed Hash	Power-up
HMAC-SHA2-384 (A3609)	Key length: 384 bits	KAT	CAST	Logged: "FIPS <testname> Test failed" or "FIPS Tests passed"	Keyed Hash	Power-up
PBKDF (A3609)	Key length: 384 bits	KAT	CAST	Logged: "FIPS <testname> Test failed" or "FIPS Tests passed"	Key Derivation	Power-up
SHA2-256 (A3609)	Hash length: 256 bits	KAT	CAST	Logged: "FIPS <testname> Test failed" or "FIPS Tests passed"	Hash	Power-up
SHA2-384 (A3609)	Hash length: 384 bits	KAT	CAST	Logged: "FIPS <testname> Test failed" or "FIPS Tests passed"	Hash	Power-up

Table 19: Conditional Self-Tests

The module performs conditional self-tests to confirm the proper operation of the cryptographic functions used in the software. These are run at power-up and may be run on demand by restarting the module.

The Known Answer Tests (KATs) perform a byte-by-byte comparison of the result with a known answer, and abort on the first mismatch. Keys are destroyed upon completion of the KAT.

Data input/output and any data processing are inhibited while the tests are in progress. If any test fails, an error status message is sent to the log and the module ceases operation. When all of the known answer tests and the pre-operational test have run to completion, one or more messages are written to the log. In the case of a failed test, the log will indicate which test failed with a "`FIPS <testname> Test failed`" message. If all tests pass, a single "`FIPS Tests passed`" message is written to the log. When all the conditional tests and the software integrity test pass, the module is capable of being operated in an approved mode of operation.

10.3 Periodic Self-Test Information

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
HMAC-SHA2-384 (A3609)	Integrity Test	SW/FW Integrity	Defined by Crypto Officer	The software integrity test may be run manually (on demand) by restarting the module.

Table 20: Pre-Operational Periodic Information

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
AES-CBC (A3609)	KAT	CAST	Defined by Crypto Officer	The known answer tests may be run manually (on demand) by restarting the module.
AES-CBC (A3609)	KAT	CAST	Defined by Crypto Officer	The known answer tests may be run manually (on demand) by restarting the module.
AES-CBC-CS1 (A3609)	KAT	CAST	Defined by Crypto Officer	The known answer tests may be run manually (on demand) by restarting the module.
AES-XTS Testing Revision 2.0 (A3609)	KAT	CAST	Defined by Crypto Officer	The known answer tests may be run manually (on demand) by restarting the module.
HMAC-SHA2-256 (A3609)	KAT	CAST	Defined by Crypto Officer	The known answer tests may be run manually (on

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
				demand) by restarting the module.
HMAC-SHA2-384 (A3609)	KAT	CAST	Defined by Crypto Officer	The known answer tests may be run manually (on demand) by restarting the module.
PBKDF (A3609)	KAT	CAST	Defined by Crypto Officer	The known answer tests may be run manually (on demand) by restarting the module.
SHA2-256 (A3609)	KAT	CAST	Defined by Crypto Officer	The known answer tests may be run manually (on demand) by restarting the module.
SHA2-384 (A3609)	KAT	CAST	Defined by Crypto Officer	The known answer tests may be run manually (on demand) by restarting the module.

Table 21: Conditional Periodic Information

The self-tests may be run on demand by restarting the module.

10.4 Error States

Name	Description	Conditions	Recovery Method	Indicator
Error	FIPS Algorithm Known Answer Test/Integrity test failed.	Module enters the error state on condition that a self-test failure has occurred.	Reinstallation of module	Error message is written to the log.

Table 22: Error States

Operation of the module will cease if there is a failure of a KAT. An operator may confirm success or failure of the KATs by viewing the syslog messages and examining the ‘FIPS Tests passed’ or ‘FIPS <testname> Test failed’ message. The log files are found:

- On Linux
  - in /var/log/messages
- On Windows
  - in the system event viewer under the Vormetric group
- On AIX
  - /opt/vormetric/DataSecurityExpert/agent/secfs/tmp/secfs.log

11 Life-Cycle Assurance

11.1 Installation, Initialization, and Startup Procedures

The module is started by starting the CipherTrust Transparent Encryption Agent software.

A call can be made to the "const char *vds_crypto_version(void);" API. The return string is the version of the cryptographic module software.

11.2 Administrator Guidance

Installation of the CTE Agent software is performed in accordance with the appropriate guidance:

- CTE Agent for DSM Windows Quick Start Guide
- CTE Agent for DSM Linux Quick Start Guide
- CTE Agent for AIX with DSM Installation & Configuration Guide
- CTE Agent for CipherTrust Manager Windows Quick Start Guide
- CTE Agent for CipherTrust Manager Linux Quick Start Guide
- CTE Agent for AIX with CipherTrust Manager Installation & Configuration Guide

11.3 Non-Administrator Guidance

In addition to the direct guidance provided in this security policy, CipherTrust Transparent Encryption user guidance is available in an online manual, which can be accessed at www.thalesdocs.com.

11.4 End of Life

The cryptographic module does not provide persistent storage of SSPs. SSPs are held in volatile memory and are zeroized following provision of the requested cryptographic service. It is recommended that the CTE agent application be uninstalled at the end of life of the module. This will result in the uninstallation of the cryptographic module and erasure of the HMAC Key used with the software integrity check.

12 Mitigation of Other Attacks

Not applicable. The module does not claim Mitigation of Other Attacks.