VMware ESXi 8.0 Update 3e Assurance Activity Report

Version:	1.1
Date:	2025-05-19
Status:	FINAL
Classification:	Public
Product:	VMware ESXi 8.0 Update 3e
Sponsor:	Broadcom
Evaluation Facility:	atsec information security
Validation ID:	11533
Validation Body:	NIAP CCEVS
Author(s):	Joachim Vandersmissen, Dick Sikkema, Hunter Barton
Quality Assurance:	Trang Huynh, Yi Mao, Evan Barnett

atsec information security corporation
4516 Seton Center Pkwy, Suite 250
Austin, TX 78759

Phone: +1 512-615-7300
Fax: +1 512-615-7301
www.atsec.com

Classification Note

Public Information (public)

This classification level is for information that may be made available to the general public. No specific security procedures are required to protect the confidentiality of this information. Information classified “public” may be freely distributed to anyone inside or outside of atsec.

Information with this classification shall be clearly marked “public”, except that it is not required to mark “public” on printed marketing material obviously intended for publication.

Revision History

Version	Date	Author(s)	Changes to Previous Revision
1.0	2025-04-14	Joachim Vandersmissen	First version
1.1	2025-05-19	Joachim Vandersmissen	Address validator feedback

Table of Contents

1 Evaluation Basis and Documents

This evaluation is based on the "Common Criteria for Information Technology Security Evaluation" CC:2022 revision 1 [CC], the "Common Methodology for Information Technology Security Evaluation" [CEM] and the following extended methodologies:

- "CC and CEM addenda - Exact Conformance, Selection-Based SFRs, Optional SFRs" [CCDB-2017-05-17];
- "Protection Profile for Virtualization Version 1.1" [PP_BASE_VIRTUALIZATION_V1.1];
- "Supporting Document: PP-Module for Server Virtualization Systems" [MOD_SV_V1.1]; and
- "Functional Package for Transport Layer Security (TLS) Version 1.1" [PKG_TLS_V1.1].

This evaluation claims exact compliance with the above PP-Configuration, PP, and PP-Modules.

The following scheme documents and interpretations have been considered:

- [CCEVS-LG]: "CCEVS Labgrams", version as of May 2025.
- [CCEVS-PL]: "CCEVS Scheme Policy Letters", version as of May 2025.
- [CCEVS-PUB]: "CCEVS Scheme Publications", version as of May 2025.
- [CCEVS-TD]: "Technical Decisions", version as of May 2025.

2 Evaluation Results

The evaluator work units have been performed, including: evaluator actions and analysis explicitly stated in the CEM; evaluator actions implicitly derived from developer action elements described in the CC Part 3; and evaluator confirmation that requirements for content and presentation of evidence elements described in the CC Part 3 have been met.

The evaluation was performed by informal analysis of the evidence provided by the sponsor.

2.1 CAVP Summary

The TOE uses the following cryptographic libraries:

- OpenSSL 3.x FIPS Provider 3.0.9 (OpenSSL)
- BoringCrypto Module 6.0 (BoringCrypto)
- VMkernel Cryptographic Module 2.0 (VMKCrypto)

The following tables summarize the Cryptographic Algorithm Validation Program (CAVP) certificates that apply to the TOE, including specific standards, options, and implementations for each algorithm of each SFR, and the applicable CAVP certificate for each.

Table 1: Mapping of SFRs to CAVP certificates (OpenSSL cryptographic library)

Cryptographic Service	Algorithm	Key Sizes	Standard	CAVP Cert.
FCS_CKM.1 Cryptographic Key Generation	ECC key pair generation	P-256, P-384, P-521 (256 to 521 bits)	FIPS 186-5	A5719
FCS_CKM.2 Cryptographic Key Distribution	ECC based key establishment	P-256, P-384, P-521 (256 to 521 bits)	SP 800-56Ar3	A5719
FCS_COP.1/Hash Cryptographic Operation (Hashing)	SHA-256, SHA-384	N/A	FIPS 180-4	A5719
FCS_COP.1/ KeyedHash Cryptographic Operation (Keyed Hash Algorithms)	HMAC-SHA-256, HMAC-SHA-384	256 and 384 bits	FIPS 198-1 FIPS 180-4	A5719
FCS_COP.1/Sig Cryptographic Operation (Signature Algorithms)	RSA	2048-bit or greater	FIPS 186-5	A5719
	ECDSA	P-256, P-384, P-521 (256 to 521 bits)	FIPS 186-5	A5719
FCS_COP.1/UDE Cryptographic Operation (Encryption/Decryption)	AES-GCM	128 and 256 bits	SP 800-38D	A5719
FCS_RBG_EXT.1 Random Bit Generation	CTR_DRBG	256 bits	SP 800-90Ar1	A5719

Table 2: Mapping of SFRs to CAVP certificates (BoringCrypto cryptographic library)

Cryptographic Service	Algorithm	Key sizes	Standard	CAVP cert.
FCS_CKM.1 Cryptographic Key Generation	ECC key pair generation	P-256, P-384, P-521 (256 to 521 bits)	FIPS 186-5	A4970
FCS_CKM.2 Cryptographic Key Distribution	ECC based key establishment	P-256, P-384, P-521 (256 to 521 bits)	SP 800-56Ar3	A4970
FCS_COP.1/Hash Cryptographic Operation (Hashing)	SHA-256, SHA-384	N/A	FIPS 180-4	A4970
FCS_COP.1/ KeyedHash Cryptographic Operation (Keyed Hash Algorithms)	HMAC-SHA-256, HMAC-SHA-384	256 and 384 bits	FIPS 198-1 FIPS 180-4	A4970
FCS_COP.1/Sig Cryptographic Operation (Signature Algorithms)	RSA	2048-bit or greater	FIPS 186-5	A4970
FCS_COP.1/UDE Cryptographic Operation (Encryption/Decryption)	AES-GCM	128 and 256 bits	SP 800-38D	A4970
FCS_RBG_EXT.1 Random Bit Generation	CTR_DRBG	256 bits	SP 800-90Ar1	A4970

Table 3: Mapping of SFRs to CAVP certificates (VMKCrypto cryptographic module)

Cryptographic Service	Algorithm	Key sizes	Standard	CAVP cert.
FCS_COP.1/Hash Cryptographic Operation (Hashing)	SHA-256	N/A	FIPS 180-4	A2792

2.2 Security Functional Requirements

2.2.1 Security audit (FAU)

2.2.1.1 Audit Data Generation (FAU_GEN.1)

TSS Assurance Activities

Assurance Activity AA-BVPP-FAU_GEN.1-ASE-01

The evaluator shall check the TSS and ensure that it lists all of the auditable events and provides a format for audit records. Each audit record format type shall be covered, along with a brief description of each field. The evaluator shall check to make sure that every audit event type mandated by the PP-Configuration is described in the TSS.

Summary

The evaluator verified that the “Security Audit” section in the TSS of [ST] provides information regarding the audit generation. It references Table 8 and Table 9 of [ST] and specifies the fields included with each audit record. The evaluator verified that Table 8 and Table 9 include every audit event type mandated by the PP-Configuration.

Guidance Assurance Activities

Assurance Activity AA-BVPP-FAU_GEN.1-AGD-01

The evaluator shall also make a determination of the administrative actions that are relevant in the context of this PP-Configuration. The evaluator shall examine the administrative guide and make a determination of which administrative commands, including subcommands, scripts, and configuration files, are related to the configuration (including enabling or disabling) of the mechanisms implemented in the TOE that are necessary to enforce the requirements specified in the PP and PP-Modules. The evaluator shall document the methodology or approach taken while determining which actions in the administrative guide are security-relevant with respect to this PP-Configuration.

Summary

The evaluator verified that the operational guidance section “Audit Configuration” supplies the reader with the knowledge to use local or remote auditing. Appendix A “Audit Information” describes the audit function. Section “Audit Record Format” describes the format of audit events in local and remote audit logs. “Audit Record Examples” section in the operational guidance provides an example audit log showing how the fields in this audit record map to the required fields specified in FAU_GEN.1. Lastly the section “Security-Relevant Audit Records” lists the events that must be audited per the claimed CC requirements along with sample records that show what an audit record for that event may look like.

Test Assurance Activities

Assurance Activity AA-BVPP-FAU_GEN.1-ATE-01

The evaluator shall test the TOE’s ability to correctly generate audit records by having the TOE generate audit records for the events listed and administrative actions. For administrative actions, the evaluator shall test that each action determined by the evaluator above to be security relevant in the context of this PP is auditable. When verifying the test results, the evaluator shall ensure the audit records generated during testing match the format specified in the administrative guide, and that the fields in each audit record have the proper entries.

Note that the testing here can be accomplished in conjunction with the testing of the security mechanisms directly.

Summary

The evaluator collected audit records in accordance with the instructions in the operational guidance, for each of the events listed in [ST]. The evaluator verified that the collected audit records are human readable and provide all information as specified in the operational guidance.

2.2.1.2 Audit Review (FAU_SAR.1)

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

Assurance Activity AA-BVPP-FAU_SAR.1-AGD-01

The evaluator shall review the operational guidance for the procedure on how to review the audit

records.

Summary

The evaluator verified that the “Viewing Audit Records” section of the operational guidance supplies the reader with the methods to view the audit records.

Test Assurance Activities

Assurance Activity AA-BVPP-FAU_SAR.1-ATE-01

The evaluator shall verify that the audit records provide all of the information specified in FAU_GEN.1 and that this information is suitable for human interpretation. The evaluation activity for this requirement is performed in conjunction with the evaluation activity for FAU_GEN.1.

Summary

This evaluation activity was performed in conjunction with the evaluation activity for FAU_GEN.1. Please refer to that evaluation activity for more information.

2.2.1.3 Protected Audit Trail Storage (FAU_STG.1)

TSS Assurance Activities

Assurance Activity AA-BVPP-FAU_STG.1-ASE-01

The evaluator shall ensure that the TSS describes how the audit records are protected from unauthorized modification or deletion. The evaluator shall ensure that the TSS describes the conditions that must be met for authorized deletion of audit records.

Summary

The evaluator verified that the “Security Audit” section in the TSS of [ST] describes that audit records are protected from unauthorized access (including modification / deletion) through file system permissions and management interface controls. There is no interface to delete audit records.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-BVPP-FAU_STG.1-ATE-01

The evaluator shall perform the following tests:

- Test 1: The evaluator shall access the audit trail as an unauthorized Administrator and attempt to modify and delete the audit records. The evaluator shall verify that these attempts fail.
- Test 2: The evaluator shall access the audit trail as an authorized Administrator and attempt to delete the audit records. The evaluator shall verify that these attempts succeed. The evaluator shall verify that only the records authorized for deletion are deleted.

Summary

The evaluator established that audit records are stored as regular files under the directory /scratch/auditLog/ on the ESXi host. These audit records cannot be accessed using the VIM API or ESXCLI. Then, the evaluator performed the following tests:

Test 1: Attempt to connect to the ESXi host as a non-administrator user using SSH. This attempt failed.

Test 2: Connect to the ESXi host as an administrator user using SSH. Then, the files in /scratch/auditLog were removed using standard Linux commands. The evaluator verified the deletion was successful.

2.2.1.4 Off-Loading of Audit Data (FAU_STG_EXT.1)

TSS Assurance Activities

Assurance Activity AA-BVPP-FAU_STG_EXT.1-ASE-01

FAU_STG_EXT.1.1: The evaluator shall examine the TSS to ensure it describes the means by which the audit data are transferred to the external audit server, and how the trusted channel is provided.

FAU_STG_EXT.1.2: The evaluator shall examine the TSS to ensure it describes what happens when the local audit data store is full.

Summary

The evaluator verified that the “Security Audit” section in the TSS of [ST] states that audit data can be transferred to a remote syslog server using TLS 1.2. This protocol provides a trusted channel as explained in the “Trusted Path/Channels” section in the TSS of [ST]. The evaluator also found that the “Security Audit” section describes how the TOE handles the maximum size of the local audit record storage. If the maximum capacity of a file is reached, a new file is created. If the maximum number of files is created, the TOE overwrites the first file and starts over.

Guidance Assurance Activities

Assurance Activity AA-BVPP-FAU_STG_EXT.1-AGD-01

FAU_STG_EXT.1.1: The evaluator shall examine the operational guidance to ensure it describes how to establish the trusted channel to the audit server, as well as describe any requirements on the audit server (particular audit server protocol, version of the protocol required, etc.), as well as configuration of the TOE needed to communicate with the audit server.

FAU_STG_EXT.1.2: The evaluator shall also examine the operational guidance to determine that it describes the relationship between the local audit data and the audit data that are sent to the audit log server. For example, when an audit event is generated, is it simultaneously sent to the external server and the local store, or is the local store used as a buffer and “cleared” periodically by sending the data to the audit server.

Summary

The evaluator verified that the “Configuring Remote Audit Server” section of the operational guidance describes how to establish the trusted channel to the audit server. This section of the AGD also describes the requirements on the audit server including the server protocol TLS 1.2 to be used as well as configuration of the TOE needed to communicate with the audit server. Lastly the “Audit Configuration” section of the operational guidance describes the relationship between the local audit data and the audit data that are sent to the audit log server such that a generated audit message is sent simultaneously to the local store and remote audit servers.

Test Assurance Activities

Assurance Activity AA-BVPP-FAU_STG_EXT.1-ATE-01

FAU_STG_EXT.1.1: Testing of the trusted channel mechanism is to be performed as specified in the evaluation activities for FTP_ITC_EXT.1.

The evaluator shall perform the following test for this requirement:

- Test 1: The evaluator shall establish a session between the TOE and the audit server according to the configuration guidance provided. The evaluator shall then examine the traffic that passes between the audit server and the TOE during several activities of the evaluator’s choice designed to generate audit data to be transferred to the audit server. The

evaluator shall observe that these data are not able to be viewed in the clear during this transfer, and that they are successfully received by the audit server. The evaluator shall record the particular software (name, version) used on the audit server during testing.

FAU_STG_EXT.1.2: The evaluator shall perform operations that generate audit data and verify that this data is stored locally. The evaluator shall perform operations that generate audit data until the local storage space is exceeded and verifies that the TOE complies with the behavior defined in the ST for FAU_STG_EXT.1.2.

Summary

To set up the audit server, the evaluator used a remote Linux server executing the rsyslog application. Then, the evaluator followed the instructions in the operational guidance to configure ESXi to send audit records to the rsyslog server. All packets were logged using tcpdump. The Host Client was used to perform various activities to stimulate the generation of audit records. The evaluator verified that the packet dump contains only encrypted TLS traffic, without any connection errors.

The generation of local audit records was also verified using the Host Client. Standard Linux commands were used to inspect the audit records in /scratch/auditLog/, allowing the evaluator to conclude that the oldest audit records are overwritten by newer audit records when the local storage space is full.

2.2.2 Cryptographic support (FCS)

2.2.2.1 Cryptographic Key Generation (FCS_CKM.1)

TSS Assurance Activities

Assurance Activity AA-BVPP-FCS_CKM.1-ASE-01

The evaluator shall ensure that the TSS identifies the key sizes supported by the TOE. If the ST specifies more than one scheme, the evaluator shall examine the TSS to verify that it identifies the usage for each scheme.

Summary

The evaluator verified that the “Cryptographic Support” section in the TSS of [ST] identifies the scheme supported by the TOE and its key size: ECC key pair generation (curves P-256, P-384 and P-521). Its usage is identified as ephemeral asymmetric key generation for TLS key exchange.

Guidance Assurance Activities

Assurance Activity AA-BVPP-FCS_CKM.1-AGD-01

The evaluator shall verify that the AGD guidance instructs the administrator how to configure the TOE to use the selected key generation schemes and key sizes for all uses defined in this PP.

Summary

The evaluator verified that the “Cryptographic Key Generation” section of the AGD guidance states that ephemeral keys generated for TLS sessions are generated using algorithms and key sizes (ECDSA curves P-256, P-384 and P-521) that depend on the negotiated TLS cipher suite. The TLS cipher suites must be configured as part of the TOE installation, specified in “To configure the allowed TLS ciphers” in the AGD guidance.

Test Assurance Activities

Assurance Activity AA-BVPP-FCS_CKM.1-ATE-01

[TD0874] Note: The following tests require the developer to provide access to a test platform that provides the evaluator with tools that are typically not found on factory products.

Key Generation for FIPS PUB 186-5 RSA Schemes

The evaluator shall verify the implementation of RSA Key Generation by the TOE using the Key Generation test. This test verifies the ability of the TSF to correctly produce values for the key components including the public verification exponent e, the private prime factors p and q, the public modulus n and the calculation of the private signature exponent d.

Key Pair generation specifies 5 ways (or methods) to generate the primes p and q. These include:

- Random Primes:
  - Provable primes
  - Probable primes
- Primes with Conditions:
  - Primes p1, p2, q1,q2, p and q shall all be provable primes
  - Primes p1, p2, q1, and q2 shall be provable primes and p and q shall be probable primes
  - Primes p1, p2, q1,q2, p and q shall all be probable primes

To test the key generation method for the Random Provable primes method and for all the Primes with Conditions methods, the evaluator shall seed the TSF key generation routine with sufficient data to deterministically generate the RSA key pair. This includes the random seeds, the public exponent of the RSA key, and the desired key length. For each key length supported, the evaluator shall have the TSF generate 25 key pairs. The evaluator shall verify the correctness of the TSF's implementation by comparing values generated by the TSF with those generated from a known good implementation.

Key Generation for Elliptic Curve Cryptography (ECC)

FIPS 186-5 ECC Key Generation Test

For each supported NIST curve (i.e., P-256, P-384 and P-521) the evaluator shall require the implementation under test (IUT) to generate 10 private/public key pairs. The private key shall be generated using an approved random bit generator (RBG). To determine correctness, the evaluator shall submit the generated key pairs to the public key verification (PKV) function of a known good implementation.

FIPS 186-5 Public Key Verification (PKV) Test

For each supported NIST curve (i.e., P-256, P-384 and P-521) the evaluator shall generate 10 private/public key pairs using the key generation function of a known good implementation and modify five of the public key values so that they are incorrect, leaving five values unchanged (i.e., correct). The evaluator shall obtain in response a set of 10 PASS/FAIL values.

Key Generation for Finite-Field Cryptography (FFC)

The evaluator shall verify the implementation of the Parameters Generation and the Key Generation for FFC by the TOE using the Parameter Generation and Key Generation test. This test verifies the ability of the TSF to correctly produce values for the field prime p, the cryptographic prime q (dividing p-1), the cryptographic group generator g, and the calculation of the private key x and public key y.

The Parameter generation specifies two ways (or methods) to generate the cryptographic prime q and the field prime p:

- Primes q and p shall both be provable primes
- Primes q and field prime p shall both be probable primes

and two ways to generate the cryptographic group generator g:

- Generator g constructed through a verifiable process
- Generator g constructed through an unverifiable process.

The Key generation specifies two ways to generate the private key x:

- len(q) bit output of RBG where $1^{g} x^{g}$ q-1
- len(q) + 64 bit output of RBG, followed by a mod q-1 operation where $1^{g} x^{g}$ q-1

The security strength of the RBG shall be at least that of the security offered by the FFC parameter set.

To test the cryptographic and field prime generation method for the provable primes method and the group generator g for a verifiable process, the evaluator shall seed the TSF parameter generation routine with sufficient data to deterministically generate the parameter set.

For each key length supported, the evaluator shall have the TSF generate 25 parameter sets and key pairs. The evaluator shall verify the correctness of the TSF's implementation by comparing values generated by the TSF with those generated from a known good implementation. Verification shall also confirm

- g != 0,1
- q divides p-1
- g^q mod p = 1
- g^x mod p = y

for each FFC parameter set and key pair.

Diffie-Hellman Group 14 and FFC Schemes using "safe-prime" groups

Testing for FFC Schemes using Diffie-Hellman group 14 and "safe-prime" groups is done as part of testing in FCS_CKM.2.1.

Summary

The evaluator verified that Automated Cryptographic Validation Testing System (ACVTS) is used to test all cryptographic algorithms in accordance with the CAVP test procedures. The results have been validated by the CAVP for the ECC and FFC key generation algorithms, and the certificate information is provided in Table 1 and Table 2.

2.2.2.2 Cryptographic Key Establishment (FCS_CKM.2)

TSS Assurance Activities

Assurance Activity AA-BVPP-FCS_CKM.2-ASE-01

The evaluator shall ensure that the supported key establishment schemes correspond to the key generation schemes identified in FCS_CKM.1.1. If the ST specifies more than one scheme, the evaluator shall examine the TSS to verify that it identifies the usage for each scheme.

Summary

The evaluator verified that the "Cryptographic Support" section in the TSS of [ST] identifies the key establishment scheme supported by the TOE: ECC based key establishment. This scheme corresponds to the ECC key pair generation scheme identified in FCS_CKM.1.1.

The usage for all key establishment schemes is identified as TLS key exchange.

Guidance Assurance Activities

Assurance Activity AA-BVPP-FCS_CKM.2-AGD-01

The evaluator shall verify that the AGD guidance instructs the administrator how to configure the TOE to use the selected key establishment schemes.

Summary

The evaluator verified that the "Cryptographic Key Establishment" section of the AGD guidance states that key establishment for TLS sessions is performed using algorithms and key sizes (ECDSA curves P-256, P-384 and P-521) that depend on the negotiated TLS cipher

suite. The TLS cipher suites must be configured as part of the TOE installation, specified in “To configure the allowed TLS ciphers” in the AGD guidance.

Test Assurance Activities

Assurance Activity AA-BVPP-FCS_CKM.2-ATE-01

The evaluator shall verify the implementation of the key establishment schemes of the supported by the TOE using the applicable tests below.

Key Establishment Schemes

RSAES-PKCS1-v1_5 Key Establishment Schemes

The evaluator shall verify the correctness of the TSF's implementation of RSAES-PKCS1-v1_5 by using a known good implementation for each protocol selected in FTP_ITC_EXT.1 that uses RSAES-PKCS1-v1_5.

SP800-56A ECC Key Establishment Schemes

The evaluator shall verify a TOE's implementation of SP800-56A key agreement schemes using the following Function and Validity tests. These validation tests for each key agreement scheme verify that a TOE has implemented the components of the key agreement scheme according to the specifications in the Recommendation. These components include the calculation of the DLC primitives (the shared secret value Z) and the calculation of the derived keying material (DKM) via the Key Derivation Function (KDF). If key confirmation is supported, the evaluator shall also verify that the components of key confirmation have been implemented correctly, using the test procedures described below. This includes the parsing of the DKM, the generation of MACdata and the calculation of MACtag.

Function Test

The Function test verifies the ability of the TOE to implement the key agreement schemes correctly. To conduct this test, the evaluator shall generate or obtain test vectors from a known good implementation of the TOE supported schemes. For each supported key agreement scheme-key agreement role combination, KDF type, and, if supported, key confirmation role- key confirmation type combination, the tester shall generate 10 sets of test vectors. The data set consists of one set of domain parameter values (FFC) or the NIST approved curve (ECC) per 10 sets of public keys. These keys are static, ephemeral, or both depending on the scheme being tested.

The evaluator shall obtain the DKM, the corresponding TOE’s public keys (static and ephemeral), the MAC tags, and any inputs used in the KDF, such as the Other Information field OI and TOE ID fields.

If the TOE does not use a KDF defined in SP 800-56A, the evaluator shall obtain only the public keys and the hashed value of the shared secret.

The evaluator shall verify the correctness of the TSF’s implementation of a given scheme by using a known good implementation to calculate the shared secret value, derive the keying material DKM, and compare hashes or MAC tags generated from these values.

If key confirmation is supported, the TSF shall perform the above for each implemented approved MAC algorithm.

Validity Test

The Validity test verifies the ability of the TOE to recognize another party’s valid and invalid key agreement results with or without key confirmation. To conduct this test, the evaluator shall obtain a list of the supporting cryptographic functions included in the SP800-56A key agreement implementation to determine which errors the TOE should be able to recognize. The evaluator generates a set of 24 (FFC) or 30 (ECC) test vectors consisting of data sets including domain parameter values or NIST approved curves, the evaluator’s public keys, the TOE’s public/private key pairs, MACTag, and any inputs used in the KDF, such as the other info and TOE ID fields.

The evaluator shall inject an error in some of the test vectors to test that the TOE recognizes invalid key agreement results caused by the following fields being incorrect: the shared secret value Z, the DKM, the other information field OI, the data to be MACed, or the generated MACTag. If the TOE contains the full or partial (only ECC) public key validation, the evaluator will also individually inject errors in both parties’ static public keys, both parties’ ephemeral public keys and the TOE’s static

private key to assure the TOE detects errors in the public key validation function and the partial key validation function (in ECC only). At least two of the test vectors shall remain unmodified and therefore should result in valid key agreement results (they should pass).

The TOE shall use these modified test vectors to emulate the key agreement scheme using the corresponding parameters. The evaluator shall compare the TOE's results with the results using a known good implementation verifying that the TOE detects these errors.

Diffie-Hellman Group 14

The evaluator shall verify the correctness of the TSF's implementation of Diffie-Hellman group 14 by using a known good implementation for each protocol selected in FTP_ITC_EXT.1 that uses Diffie-Hellman Group 14.

FFC Schemes using "safe-prime" groups (identified in Appendix D of SP 800-56A Revision 3)

The evaluator shall verify the correctness of the TSF's implementation of "safe-prime" groups by using a known good implementation for each protocol selected in FTP_ITC_EXT.1 that uses "safe-prime" groups. This test must be performed for each "safe-prime" group that each protocol uses.

Summary

The evaluator verified that Automated Cryptographic Validation Testing System (ACVTS) is used to test all cryptographic algorithms in accordance with the CAVP test procedures. The results have been validated by the CAVP for the ECC based key establishment algorithm, and the certificate information is provided in Table 1 and Table 2.

2.2.2.3 Cryptographic Key Destruction (FCS_CKM_EXT.4)

TSS Assurance Activities

Assurance Activity AA-BVPP-FCS_CKM_EXT.4-ASE-01

The evaluator shall check to ensure the TSS lists each type of key and its origin and location in memory or storage. The evaluator shall verify that the TSS describes when each type of key is cleared.

Summary

The evaluator verified that the "Cryptographic Support" section in the TSS of [ST] contains a table listing each type of key used by the TSF, as well as their origin and location in memory. The table also includes the time of zeroization for each key type.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-BVPP-FCS_CKM_EXT.4-ATE-01

For each key clearing situation the evaluator shall perform one of the following activities:

- The evaluator shall use appropriate combinations of specialized operational or development environments, development tools (debuggers, emulators, simulators, etc.), or instrumented builds (developmental, debug, or release) to demonstrate that keys are cleared correctly, including all intermediate copies of the key that may have been created internally by the TOE during normal cryptographic processing.
- In cases where testing reveals that third-party software modules or programming language run-time environments do not properly overwrite keys, this fact must be documented. Likewise, it must be documented if there is no practical way to determine whether such modules or environments destroy keys properly.
- In cases where it is impossible or impracticable to perform the above tests, the evaluator shall describe how keys are destroyed in such cases, to include:

- Which keys are affected
- The reasons why testing is impossible or impracticable
- Evidence that keys are destroyed appropriately (e.g., citations to component documentation, component developer/vendor attestation, component vendor test results)
- Aggravating and mitigating factors that may affect the timeliness or execution of key destruction (e.g., caching, garbage collection, operating system memory management)

Use of debug or instrumented builds of the TOE and TOE components is permitted in order to demonstrate that the TOE takes appropriate action to destroy keys. These builds should be based on the same source code as are release builds (of course, with instrumentation and debug-specific code added).

Summary

In ESXi, keys are managed by two cryptographic libraries: OpenSSL and BoringCrypto. Both libraries were tested separately. The evaluator used the GDB debugger and separate debug information to instrument the libraries as they executed.

For OpenSSL, the evaluator created a GDB test script that displays the private key material (e.g. AES keys, DRBG state, RSA private keys, ...) before and after zeroization. This GDB test script was then executed in conjunction with the “openssl speed” application.

For BoringCrypto, the evaluator created a GDB test script that displays any memory that is zeroized by the application. This GDB test script was then attached to the running Envoy application (web server). The evaluator connected to the Host Client to stimulate the generation and zeroization of key material.

2.2.2.4 Cryptographic Operation (Hashing) (FCS_COP.1/Hash)

TSS Assurance Activities

Assurance Activity AA-BVPP-FCS_COP.1-HASH-ASE-01

The evaluator shall check that the association of the hash function with other TSF cryptographic functions (for example, the digital signature verification function) is documented in the TSS.

Summary

The evaluator verified that the “Cryptographic Support” section in the TSS of [ST] identifies the usage of the hash functions with the other TSF cryptographic functions (digital signature and keyed-hash message authentication (HMAC) functions).

Guidance Assurance Activities

Assurance Activity AA-BVPP-FCS_COP.1-HASH-AGD-01

The evaluator checks the AGD documents to determine that any configuration that is required to be done to configure the functionality for the required hash sizes is present.

Summary

The evaluator verified that the “Cryptographic Operation” section of the AGD guidance states that FIPS 140 approved algorithms are used for all operations, and that the NIAP evaluated functionalities are enabled by default.

Test Assurance Activities

Assurance Activity AA-BVPP-FCS_COP.1-HASH-ATE-01

SHA-1 and SHA-2 Tests

The TSF hashing functions can be implemented in one of two modes. The first mode is the byte-oriented mode. In this mode the TSF only hashes messages that are an integral number of bytes in length; i.e., the length (in bits) of the message to be hashed is divisible by 8. The second mode is the bit-oriented mode. In this mode the TSF hashes messages of arbitrary length. As there are different tests for each mode, an indication is given in the following sections for the bit-oriented vs. the byte-oriented test MACs.

The evaluator shall perform all of the following tests for each hash algorithm implemented by the TSF and used to satisfy the requirements of this PP.

The following tests require the developer to provide access to a test platform that provides the evaluator with tools that are typically not found on factory products.

Short Messages Test Bit-oriented Mode

The evaluators devise an input set consisting of m+1 messages, where m is the block length of the hash algorithm. The length of the messages range sequentially from 0 to m bits. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF.

Short Messages Test Byte-oriented Mode

The evaluators devise an input set consisting of m/8+1 messages, where m is the block length of the hash algorithm. The length of the messages range sequentially from 0 to m/8 bytes, with each message being an integral number of bytes. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF.

Selected Long Messages Test Bit-oriented Mode

The evaluators devise an input set consisting of m messages, where m is the block length of the hash algorithm. The length of the ith message is 512 + 99*i, where $1 \le i \le m$. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF.

Selected Long Messages Test Byte-oriented Mode

The evaluators devise an input set consisting of m/8 messages, where m is the block length of the hash algorithm. The length of the ith message is 512 + 8*99*i, where $1 \le i \le m/8$. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF.

Pseudorandomly Generated Messages Test

This test is for byte-oriented implementations only. The evaluators randomly generate a seed that is n bits long, where n is the length of the message digest produced by the hash function to be tested. The evaluators then formulate a set of 100 messages and associated digests by following the algorithm provided in Figure 1 of [SHAVS]. The evaluators then ensure that the correct result is produced when the messages are provided to the TSF.

SHA-3 Tests

The tests below are derived from the The Secure Hash Algorithm-3 Validation System (SHA3VS), Updated: April 7, 2016, from the National Institute of Standards and Technology.

For each SHA-3-XXX implementation, XXX represents d, the digest length in bits. The capacity, c, is equal to 2d bits. The rate is equal to 1600-c bits.

The TSF hashing functions can be implemented with one of two orientations. The first is a bit-oriented mode that hashes messages of arbitrary length. The second is a byte-oriented mode that hashes messages that are an integral number of bytes in length (i.e., the length (in bits) of the message to be hashed is divisible by 8). Separate tests for each orientation are given below.

The evaluator shall perform all of the following tests for each hash algorithm and orientation implemented by the TSF and used to satisfy the requirements of this PP. The evaluator shall compare digest values produced by a known-good SHA-3 implementation against those generated by running the same values through the TSF.

Short Messages Test, Bit-oriented Mode

The evaluators devise an input set consisting of rate+1 short messages. The length of the messages ranges sequentially from 0 to rate bits. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF. The message of length 0 is omitted if the TOE does not support zero-length messages.

Short Messages Test, Byte-oriented Mode

The evaluators devise an input set consisting of rate/8+1 short messages. The length of the messages ranges sequentially from 0 to rate/8 bytes, with each message being an integral number of bytes. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF. The message of length 0 is omitted if the TOE does not support zero-length messages.

Selected Long Messages Test, Bit-oriented Mode

The evaluators devise an input set consisting of 100 long messages ranging in size from rate+(rate+1) to rate+(100*(rate+1)), incrementing by rate+1. (For example, SHA-3-256 has a rate of 1088 bits. Therefore, 100 messages will be generated with lengths 2177, 3266, ..., 109988 bits.) The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF.

Selected Long Messages Test, Byte-oriented Mode

The evaluators devise an input set consisting of 100 messages ranging in size from (rate+(rate+8)) to (rate+100*(rate+8)), incrementing by rate+8. (For example, SHA-3-256 has a rate of 1088 bits. Therefore 100 messages will be generated of lengths 2184, 3280, 4376, ..., 110688 bits.) The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF.

Pseudorandomly Generated Messages Monte Carlo) Test, Byte-oriented Mode

The evaluators supply a seed of d bits (where d is the length of the message digest produced by the hash function to be tested. This seed is used by a pseudorandom function to generate 100,000 message digests. One hundred of the digests (every 1000th digest) are recorded as checkpoints. The TOE then uses the same procedure to generate the same 100,000 message digests and 100 checkpoint values. The evaluators then compare the results generated to ensure that the correct result is produced when the messages are generated by the TSF.

Summary

The evaluator verified that Automated Cryptographic Validation Testing System (ACVTS) is used to test all cryptographic algorithms in accordance with the CAVP test procedures. The results have been validated by the CAVP for the hash algorithms, and the certificate information is provided in Table 1, Table 2, and Table 3.

2.2.2.5 Cryptographic Operation (Keyed Hash algorithms) (FCS_COP.1/KeyedHash)

TSS Assurance Activities

Assurance Activity AA-BVPP-FCS_COP.1-KEYEDHASH-ASE-01

The evaluator shall examine the TSS to ensure that it specifies the following values used by the HMAC function: key length, hash function used, block size, and output MAC length used.

Summary

The evaluator verified that the “Cryptographic Support” section in the TSS of [ST] identifies the key lengths, block sizes, and output MAC sizes for the hash functions used by the TOE (HMAC-SHA-256 and HMAC-SHA-384).

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-BVPP-FCS_COP.1-KEYEDHASH-ATE-01

The following tests require the developer to provide access to a test platform that provides the evaluator with tools that are typically not found on factory products.

For each of the supported parameter sets, the evaluator shall compose 15 sets of test data. Each set shall consist of a key and message data. The evaluator shall have the TSF generate HMAC tags for these sets of test data. The resulting MAC tags shall be compared to the result of generating HMAC tags with the same key and IV using a known good implementation.

Summary

The evaluator verified that Automated Cryptographic Validation Testing System (ACVTS) is used to test all cryptographic algorithms in accordance with the CAVP test procedures. The results have been validated by the CAVP for the keyed-hash message authentication algorithms, and the certificate information is provided in Table 1 and Table 2.

2.2.2.6 Cryptographic Operation (Signature Algorithms) (FCS_COP.1/Sig)

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-BVPP-FCS_COP.1-SIG-ATE-01

[TD0874] The following tests require the developer to provide access to a test platform that provides the evaluator with tools that are typically not found on factory products.

ECDSA Algorithm Tests

ECDSA FIPS 186-5 Signature Generation Test

For each supported NIST curve (i.e., P-256, P-384 and P-521) and SHA function pair, the evaluator shall generate 10 1024-bit long messages and obtain for each message a public key and the resulting signature values R and S. To determine correctness, the evaluator shall use the signature verification function of a known good implementation.

ECDSA FIPS 186-5 Signature Verification Test

For each supported NIST curve (i.e., P-256, P-384 and P-521) and SHA function pair, the evaluator shall generate a set of 10 1024-bit message, public key and signature tuples and modify one of the values (message, public key or signature) in five of the 10 tuples. The evaluator shall obtain in response a set of 10 PASS/FAIL values.

RSA Signature Algorithm Tests

Signature Generation Test

The evaluator shall verify the implementation of RSA Signature Generation by the TOE using the Signature Generation Test. To conduct this test, the evaluator shall generate or obtain 10 messages from a trusted reference implementation for each modulus size/SHA combination supported by the TSF. The evaluator shall have the TOE use their private key and modulus value to sign these messages.

The evaluator shall verify the correctness of the TSF's signature using a known good implementation and the associated public keys to verify the signatures.

Signature Verification Test

The evaluator shall perform the Signature Verification test to verify the ability of the TOE to recognize another party's valid and invalid signatures. The evaluator shall inject errors into the test vectors produced during the Signature Verification Test by introducing errors in some of the public keys e, messages, IR format, or signatures. The TOE attempts to verify the signatures and returns success or failure.

The evaluator shall use these test vectors to emulate the signature verification test using the corresponding parameters and verify that the TOE detects these errors.

Summary

The evaluator verified that Automated Cryptographic Validation Testing System (ACVTS) is used to test all cryptographic algorithms in accordance with the CAVP test procedures. The results have been validated by the CAVP for the ECDSA and RSA signature generation and verification algorithms, and the certificate information is provided in Table 1 and Table 2.

2.2.2.7 Cryptographic Operation (AES Data Encryption/Decryption) (FCS_COP.1/UDE)

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-BVPP-FCS_COP.1-UDE-ATE-01

The following tests require the developer to provide access to a test platform that provides the evaluator with tools that are typically not found on factory products.

Tests

AES-CBC Tests

AES-CBC Known Answer Tests

There are four Known Answer Tests (KATs), described below. In all KATs, the plaintext, ciphertext, and IV values shall be 128-bit blocks. The results from each test may either be obtained by the evaluator directly or by supplying the inputs to the implementer and receiving the results in response. To determine correctness, the evaluator shall compare the resulting values to those obtained by submitting the same inputs to a known good implementation.

KAT-1. To test the encrypt functionality of AES-CBC, the evaluator shall supply a set of 10 plaintext values and obtain the ciphertext value that results from AES-CBC encryption of the given plaintext using a key value of all zeros and an IV of all zeros. Five plaintext values shall be encrypted with a 128-bit all-zeros key, and the other five shall be encrypted with a 256-bit all- zeros key.

To test the decrypt functionality of AES-CBC, the evaluator shall perform the same test as for encrypt, using 10 ciphertext values as input and AES-CBC decryption.

KAT-2. To test the encrypt functionality of AES-CBC, the evaluator shall supply a set of 10 key values and obtain the ciphertext value that results from AES-CBC encryption of an all-zeros plaintext using the given key value and an IV of all zeros. Five of the keys shall be 128-bit keys, and the other five shall be 256-bit keys.

To test the decrypt functionality of AES-CBC, the evaluator shall perform the same test as for encrypt, using an all-zero ciphertext value as input and AES-CBC decryption.

KAT-3. To test the encrypt functionality of AES-CBC, the evaluator shall supply the two sets of key values described below and obtain the ciphertext value that results from AES encryption of an all-zeros plaintext using the given key value and an IV of all zeros. The first set of keys shall have 128 128-bit keys, and the second set shall have 256 256-bit keys. Key i in each set shall have the

leftmost i bits be ones and the rightmost N-i bits be zeros, for i in [1,N].

To test the decrypt functionality of AES-CBC, the evaluator shall supply the two sets of key and ciphertext value pairs described below and obtain the plaintext value that results from AES-CBC decryption of the given ciphertext using the given key and an IV of all zeros. The first set of key/ciphertext pairs shall have 128 128-bit key/ciphertext pairs, and the second set of key/ciphertext pairs shall have 256 256-bit key/ciphertext pairs. Key i in each set shall have the leftmost i bits be ones and the rightmost N-i bits be zeros, for i in [1,N]. The ciphertext value in each pair shall be the value that results in an all-zeros plaintext when decrypted with its corresponding key.

KAT-4. To test the encrypt functionality of AES-CBC, the evaluator shall supply the set of 128 plaintext values described below and obtain the two ciphertext values that result from AES-CBC encryption of the given plaintext using a 128-bit key value of all zeros with an IV of all zeros and using a 256-bit key value of all zeros with an IV of all zeros, respectively. Plaintext value i in each set shall have the leftmost i bits be ones and the rightmost 128-i bits be zeros, for i in [1,128].

To test the decrypt functionality of AES-CBC, the evaluator shall perform the same test as for encrypt, using ciphertext values of the same form as the plaintext in the encrypt test as input and AES-CBC decryption.

AES-CBC Multi-Block Message Test

The evaluator shall test the encrypt functionality by encrypting an i-block message where 1 < i <= 10. The evaluator shall choose a key, an IV and plaintext message of length i blocks and encrypt the message, using the mode to be tested, with the chosen key and IV. The ciphertext shall be compared to the result of encrypting the same plaintext message with the same key and IV using a known good implementation.

The evaluator shall also test the decrypt functionality for each mode by decrypting an i-block message where 1 < i <=10. The evaluator shall choose a key, an IV and a ciphertext message of length i blocks and decrypt the message, using the mode to be tested, with the chosen key and IV. The plaintext shall be compared to the result of decrypting the same ciphertext message with the same key and IV using a known good implementation.

AES-CBC Monte Carlo Tests

The evaluator shall test the encrypt functionality using a set of 200 plaintext, IV, and key 3- tuples. 100 of these shall use 128 bit keys, and 100 shall use 256 bit keys. The plaintext and IV values shall be 128-bit blocks. For each 3-tuple, 1000 iterations shall be run as follows:

Input: PT, IV, Key

for i = 1 to 1000:

if i == 1:

CT[1] = AES-CBC-Encrypt(Key, IV, PT)

PT = IV

else:

CT[i] = AES-CBC-Encrypt(Key, PT)

PT = CT[i-1]

The ciphertext computed in the 1000th iteration (i.e., CT[1000]) is the result for that trial. This result shall be compared to the result of running 1000 iterations with the same values using a known good implementation.

The evaluator shall test the decrypt functionality using the same test as for encrypt, exchanging CT and PT and replacing AES-CBC-Encrypt with AES-CBC-Decrypt.

AES-CCM Tests

The evaluator shall test the generation-encryption and decryption-verification functionality of AES-CCM for the following input parameter and tag lengths:

128 bit and 256 bit keys

Two payload lengths. One payload length shall be the shortest supported payload length, greater than or equal to zero bytes. The other payload length shall be the longest supported payload length, less than or equal to 32 bytes (256 bits).

Two or three associated data lengths. One associated data length shall be 0, if supported. One associated data length shall be the shortest supported payload length, greater than or equal to zero bytes. One associated data length shall be the longest supported payload length, less than or equal to 32 bytes (256 bits). If the implementation supports an associated data length of 2^{16} bytes, an associated data length of 2^{16} bytes shall be tested.

Nonce lengths. All supported nonce lengths between 7 and 13 bytes, inclusive, shall be tested.

Tag lengths. All supported tag lengths of 4, 6, 8, 10, 12, 14 and 16 bytes shall be tested.

To test the generation-encryption functionality of AES-CCM, the evaluator shall perform the following four tests:

- **Test 1:** For EACH supported key and associated data length and ANY supported payload, nonce and tag length, the evaluator shall supply one key value, one nonce value and 10 pairs of associated data and payload values and obtain the resulting ciphertext.
- **Test 2:** For EACH supported key and payload length and ANY supported associated data, nonce and tag length, the evaluator shall supply one key value, one nonce value and 10 pairs of associated data and payload values and obtain the resulting ciphertext.
- **Test 3:** For EACH supported key and nonce length and ANY supported associated data, payload and tag length, the evaluator shall supply one key value and 10 associated data, payload and nonce value 3-tuples and obtain the resulting ciphertext.
- **Test 4:** For EACH supported key and tag length and ANY supported associated data, payload and nonce length, the evaluator shall supply one key value, one nonce value and 10 pairs of associated data and payload values and obtain the resulting ciphertext.

To determine correctness in each of the above tests, the evaluator shall compare the ciphertext with the result of generation-encryption of the same inputs with a known good implementation.

To test the decryption-verification functionality of AES-CCM, for EACH combination of supported associated data length, payload length, nonce length and tag length, the evaluator shall supply a key value and 15 nonce, associated data and ciphertext 3-tuples and obtain either a FAIL result or a PASS result with the decrypted payload. The evaluator shall supply 10 tuples that should FAIL and 5 that should PASS per set of 15.

Additionally, the evaluator shall use tests from the IEEE 802.11-02/362r6 document “Proposed Test vectors for IEEE 802.11 TGi”, dated September 10, 2002, Section 2.1 AES-CCMP Encapsulation Example and Section 2.2 Additional AES CCMP Test Vectors to further verify the IEEE 802.11-2007 implementation of AES-CCMP.

AES-GCM Test

The evaluator shall test the authenticated encrypt functionality of AES-GCM for each combination of the following input parameter lengths:

128 bit and 256 bit keys

Two plaintext lengths. One of the plaintext lengths shall be a non-zero integer multiple of 128 bits, if supported. The other plaintext length shall not be an integer multiple of 128 bits, if supported.

Three AAD lengths. One AAD length shall be 0, if supported. One AAD length shall be a non-zero integer multiple of 128 bits, if supported. One AAD length shall not be an integer multiple of 128 bits, if supported.

Two IV lengths. If 96 bit IV is supported, 96 bits shall be one of the two IV lengths tested.

The evaluator shall test the encrypt functionality using a set of 10 key, plaintext, AAD, and IV tuples for each combination of parameter lengths above and obtain the ciphertext value and tag that results from AES-GCM authenticated encrypt. Each supported tag length shall be tested at least once per set of 10. The IV value may be supplied by the evaluator or the implementation being tested, as long as it is known.

The evaluator shall test the decrypt functionality using a set of 10 key, ciphertext, tag, AAD, and IV 5-tuples for each combination of parameter lengths above and obtain a Pass/Fail result on authentication and the decrypted plaintext if Pass. The set shall include five tuples that Pass and five that Fail.

The results from each test may either be obtained by the evaluator directly or by supplying the inputs to the implementer and receiving the results in response. To determine correctness, the evaluator shall compare the resulting values to those obtained by submitting the same inputs to a known good implementation.

AES-XTS Tests

The evaluator shall test the encrypt functionality of XTS-AES for each combination of the following input parameter lengths:

- 256 bit (for AES-128) and 512 bit (for AES-256) keys
- **Three data unit (i.e., plaintext) lengths.** One of the data unit lengths shall be a non-zero integer multiple of 128 bits, if supported. One of the data unit lengths shall be an integer multiple of 128 bits, if supported. The third data unit length shall be either the longest supported data unit length or 2^{16} bits, whichever is smaller.

using a set of 100 (key, plaintext and 128-bit random tweak value) 3-tuples and obtain the ciphertext that results from XTS-AES encrypt.

The evaluator may supply a data unit sequence number instead of the tweak value if the implementation supports it. The data unit sequence number is a base-10 number ranging between 0 and 255 that implementations convert to a tweak value internally.

The evaluator shall test the decrypt functionality of XTS-AES using the same test as for encrypt, replacing plaintext values with ciphertext values and XTS-AES encrypt with XTS-AES decrypt.

AES Key Wrap (AES-KW) and Key Wrap with Padding (AES-KWP) Test

The evaluator shall test the authenticated encryption functionality of AES-KW for EACH combination of the following input parameter lengths:

- 128 and 256 bit key encryption keys (KEKs)
- **Three plaintext lengths.** One of the plaintext lengths shall be two semi-blocks (128 bits). One of the plaintext lengths shall be three semi-blocks (192 bits). The third data unit length shall be the longest supported plaintext length less than or equal to 64 semi-blocks (4096 bits).

using a set of 100 key and plaintext pairs and obtain the ciphertext that results from AES-KW authenticated encryption. To determine correctness, the evaluator shall use the AES-KW authenticated-encryption function of a known good implementation.

The evaluator shall test the authenticated-decryption functionality of AES-KW using the same test as for authenticated-encryption, replacing plaintext values with ciphertext values and AES-KW authenticated-encryption with AES-KW authenticated-decryption.

The evaluator shall test the authenticated-encryption functionality of AES-KWP using the same test as for AES-KW authenticated-encryption with the following change in the three plaintext lengths:

One plaintext length shall be one octet. One plaintext length shall be 20 octets (160 bits).

One plaintext length shall be the longest supported plaintext length less than or equal to 512 octets (4096 bits).

The evaluator shall test the authenticated-decryption functionality of AES-KWP using the same test as for AES-KWP authenticated-encryption, replacing plaintext values with ciphertext values and AES-KWP authenticated-encryption with AES-KWP authenticated-decryption.

AES-CTR Test

- **Test 1:** Known Answer Tests (KATs)

There are four Known Answer Tests (KATs) described below. For all KATs, the plaintext, initialization vector (IV), and ciphertext values shall be 128-bit blocks. The results from each test may either be obtained by the validator directly or by supplying the inputs to the implementer and receiving the

results in response. To determine correctness, the evaluator shall compare the resulting values to those obtained by submitting the same inputs to a known good implementation.

Test 1a: To test the encrypt functionality, the evaluator shall supply a set of 10 plaintext values and obtain the ciphertext value that results from encryption of the given plaintext using a key value of all zeros and an IV of all zeros. Five plaintext values shall be encrypted with a 128-bit all zeros key, and the other five shall be encrypted with a 256-bit all zeros key. To test the decrypt functionality, the evaluator shall perform the same test as for encrypt, using 10 ciphertext values as input.

Test 1b: To test the encrypt functionality, the evaluator shall supply a set of 10 key values and obtain the ciphertext value that results from encryption of an all zeros plaintext using the given key value and an IV of all zeros. Five of the key values shall be 128-bit keys, and the other five shall be 256-bit keys. To test the decrypt functionality, the evaluator shall perform the same test as for encrypt, using an all zero ciphertext value as input.

Test 1c: To test the encrypt functionality, the evaluator shall supply the two sets of key values described below and obtain the ciphertext values that result from AES encryption of an all zeros plaintext using the given key values and an IV of all zeros. The first set of keys shall have 128 128-bit keys, and the second shall have 256 256-bit keys. Key_i in each set shall have the leftmost i bits be ones and the rightmost N-i bits be zeros, for i in [1, N]. To test the decrypt functionality, the evaluator shall supply the two sets of key and ciphertext value pairs described below and obtain the plaintext value that results from decryption of the given ciphertext using the given key values and an IV of all zeros. The first set of key/ciphertext pairs shall have 128 128-bit key/ciphertext pairs, and the second set of key/ciphertext pairs shall have 256 256-bit pairs. Key_i in each set shall have the leftmost I bits be ones and the rightmost N-i bits be zeros for i in [1, N]. The ciphertext value in each pair shall be the value that results in an all zeros plaintext when decrypted with its corresponding key.

Test 1d: To test the encrypt functionality, the evaluator shall supply the set of 128 plaintext values described below and obtain the two ciphertext values that result from encryption of the given plaintext using a 128-bit key value of all zeros and using a 256 bit key value of all zeros, respectively, and an IV of all zeros. Plaintext value i in each set shall have the leftmost bits be ones and the rightmost 128-i bits be zeros, for i in [1, 128]. To test the decrypt functionality, the evaluator shall perform the same test as for encrypt, using ciphertext values of the same form as the plaintext in the encrypt test as input.

- **Test 2:** Multi-Block Message Test

The evaluator shall test the encrypt functionality by encrypting an i-block message where 1 less-than i less-than-or-equal to 10. For each i the evaluator shall choose a key, IV, and plaintext message of length i blocks and encrypt the message, using the mode to be tested, with the chosen key. The ciphertext shall be compared to the result of encrypting the same plaintext message with the same key and IV using a known good implementation. The evaluator shall also test the decrypt functionality by decrypting an i-block message where 1 less-than i less-than-or-equal to 10. For each i the evaluator shall choose a key and a ciphertext message of length i blocks and decrypt the message, using the mode to be tested, with the chosen key. The plaintext shall be compared to the result of decrypting the same ciphertext message with the same key using a known good implementation.

- **Test 3:** Monte-Carlo Test

For AES-CTR mode perform the Monte Carlo Test for ECB Mode on the encryption engine of the counter mode implementation. There is no need to test the decryption engine. The evaluator shall test the encrypt functionality using 200 plaintext/key pairs. 100 of these shall use 128 bit keys, and 100 of these shall use 256 bit keys. The plaintext values shall be 128-bit blocks. For each pair, 1000 iterations shall be run as follows:

For AES-ECB mode
Input: PT, Key
for i = 1 to 1000:
CT[i] = AES-ECB-Encrypt(Key, PT)
PT = CT[i]

The ciphertext computed in the 1000th iteration is the result for that trial. This result shall be compared to the result of running 1000 iterations with the same values using a known good

implementation.

If "invoke platform-provided" is selected, the evaluator confirms that SSH connections are only successful if appropriate algorithms and appropriate key sizes are configured. To do this, the evaluator shall perform the following tests:

- **Test 1:** [Conditional: TOE is an SSH server] The evaluator shall configure an SSH client to connect with an invalid cryptographic algorithm and key size for each listening SSH socket connection on the TOE. The evaluator initiates SSH client connections to each listening SSH socket connection on the TOE and observes that the connection fails in each attempt.
- **Test 2:** [Conditional: TOE is an SSH client] The evaluator shall configure a listening SSH socket on a remote SSH server that accepts only invalid cryptographic algorithms and keys. The evaluator uses the TOE to attempt an SSH connection to this server and observes that the connection fails.

Summary

The evaluator verified that Automated Cryptographic Validation Testing System (ACVTS) is used to test all cryptographic algorithms in accordance with the CAVP test procedures. The results have been validated by the CAVP for the encryption/decryption algorithms, and the certificate information is provided in Table 1 and Table 2.

2.2.2.8 Extended: Entropy for Virtual Machines (FCS_ENT_EXT.1)

TSS Assurance Activities

Assurance Activity AA-BVPP-FCS_ENT_EXT.1-ASE-01

The evaluator shall verify that the TSS describes how the TOE provides entropy to Guest VMs, and how to access the interface to acquire entropy or random numbers. The evaluator shall verify that the TSS describes the mechanisms for ensuring that one VM does not affect the entropy acquired by another.

Summary

The evaluator verified that the “Cryptographic Support” section in the TSS of [ST] states that the hardware RDSEED instruction is passed through to Guest VMs. The isolation of VMs ensures that this passthrough access remains independent between Guest VMs, preventing VMs from affecting each others entropy collection.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-BVPP-FCS_ENT_EXT.1-ATE-01

The evaluator shall perform the following tests:

- Test 1: The evaluator shall invoke entropy from each Guest VM. The evaluator shall verify that each VM acquires values from the interface.
- Test 2: The evaluator shall invoke entropy from multiple VMs as nearly simultaneously as practicable. The evaluator shall verify that the entropy used in one VM is not identical to that invoked from the other VMs.

Summary

The evaluator set up two identical Guest VMs using Ubuntu 24.04.1. The evaluator also wrote a C application which waits for a pre-defined time, then invokes _rdseed32_step() to collect entropy data from RDSEED.

Test 1: the evaluator compiled and executed the test application, and found that entropy data was acquired from RDSEED.

Test 2: the evaluator compiled the test application on both VMs. Then, the evaluator configured the application to collect data at a precise point in time (i.e., simultaneously from both VMs). The evaluator found that entropy data was different across the VMs.

2.2.2.9 HTTPS Protocol (FCS_HTTPS_EXT.1)

TSS Assurance Activities

Assurance Activity AA-BVPP-FCS_HTTPS_EXT.1-ASE-01

The evaluator shall check the TSS to ensure that it is clear on how HTTPS uses TLS to establish an administrative session, focusing on any client authentication required by the TLS protocol vs. security administrator authentication which may be done at a different level of the processing stack.

Summary

The evaluator verified that the “Trusted Path/Channels” section in the TSS of [ST] specifies how TLS/HTTPS is used to establish a trusted path. Additionally, the “Cryptographic Support” section describes the ciphers used and the “Identification and Authentication” section explains how server certificates are validated. There is no TLS certificate-based client authentication.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-BVPP-FCS_HTTPS_EXT.1-ATE-01

Testing for this activity is done as part of the TLS testing; this may result in additional testing if the TLS tests are done at the TLS protocol level.

Summary

Testing for this activity was done as part of the TLS testing.

2.2.2.10 Cryptographic Operation (Random Bit Generation) (FCS_RBG_EXT.1)

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-BVPP-FCS_RBG_EXT.1-ATE-01

The evaluator shall also perform the following tests, depending on the standard to which the RBG conforms.

The evaluator shall perform 15 trials for the RBG implementation. If the RBG is configurable, the evaluator shall perform 15 trials for each configuration. The evaluator shall also confirm that the operational guidance contains appropriate instructions for configuring the RBG functionality.

If the RBG has prediction resistance enabled, each trial consists of (1) instantiate DRBG, (2) generate the first block of random bits (3) generate a second block of random bits (4) uninstantiate. The evaluator verifies that the second block of random bits is the expected value. The evaluator shall generate eight input values for each trial. The first is a count (0 – 14). The next three are entropy input, nonce, and personalization string for the instantiate operation. The next two are additional input and entropy input for the first call to generate. The final two are additional input and entropy input for the second call to generate. These values are randomly generated. “generate one block of random bits” means to generate random bits with number of returned bits equal to the

Output Block Length (as defined in NIST SP 800-90A).

If the RBG does not have prediction resistance, each trial consists of (1) instantiate DRBG, (2) generate the first block of random bits (3) reseed, (4) generate a second block of random bits (5) uninstantiate. The evaluator verifies that the second block of random bits is the expected value. The evaluator shall generate eight input values for each trial. The first is a count (0 – 14). The next three are entropy input, nonce, and personalization string for the instantiate operation. The fifth value is additional input to the first call to generate. The sixth and seventh are additional input and entropy input to the call to reseed. The final value is additional input to the second generate call.

The following paragraphs contain more information on some of the input values to be generated/selected by the evaluator.

- Entropy input: the length of the entropy input value must equal the seed length.
- Nonce: If a nonce is supported (CTR_DRBG with no df does not use a nonce), the nonce bit length is one-half the seed length. Personalization string: The length of the personalization string must be <= seed length. If the implementation only supports one personalization string length, then the same length can be used for both values. If more than one string length is supported, the evaluator shall use personalization strings of two different lengths. If the implementation does not use a personalization string, no value needs to be supplied.
- Additional input: the additional input bit lengths have the same defaults and restrictions as the personalization string lengths.

Summary

The evaluator verified that Automated Cryptographic Validation Testing System (ACVTS) is used to test all cryptographic algorithms in accordance with the CAVP test procedures. The results have been validated by the CAVP for the DRBG algorithms, and the certificate information is provided in Table 1 and Table 2.

2.2.2.11 TLS Protocol (FCS_TLS_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FCS_TLS_EXT.1-ASE-01

No assurance activities defined.

Guidance Assurance Activities

Assurance Activity AA-FCS_TLS_EXT.1-AGD-01

The evaluator shall ensure that the selections indicated in the ST are consistent with selections in the dependent components.

Summary

The evaluator notes that Section 5.2.2.11 of [ST] selects both “TLS as a client” and “TLS as a server”. Consequently, the evaluator verified that FCS_TLSC_EXT.1 (Section 5.2.2.12) and FCS_TLSS_EXT.1 (Section 5.2.2.14) are claimed in [ST] as required.

Test Assurance Activities

Assurance Activity AA-FCS_TLS_EXT.1-ATE-01

No assurance activities defined.

2.2.2.12 TLS Client Protocol (FCS_TLSC_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FCS_TLSC_EXT.1-ASE-01

FCS_TLSC_EXT.1.1: The evaluator shall check the description of the implementation of this protocol in the TSS to ensure that the cipher suites supported are specified. The evaluator shall

check the TSS to ensure that the cipher suites specified include those listed for this component.

FCS_TLSC_EXT.1.2: The evaluator shall ensure that the TSS describes the client's method of establishing all reference identifiers from the application-configured reference identifier, including which types of reference identifiers are supported (e.g. Common Name, DNS Name, URI Name, Service Name, or other application-specific Subject Alternative Names) and whether IP addresses and wildcards are supported. The evaluator shall ensure that this description identifies whether and the manner in which certificate pinning is supported or used by the product.

FCS_TLSC_EXT.1.3: If the selection for authorizing override of invalid certificates is made, then the evaluator shall ensure that the TSS includes a description of how and when user or administrator authorization is obtained. The evaluator shall also ensure that the TSS describes any mechanism for storing such authorizations, such that future presentation of such otherwise-invalid certificates permits establishment of a trusted channel without user or administrator action.

Summary

The evaluator verified that the "Cryptographic Support" section in the TSS of [ST] identifies the cipher suites supported by the TOE when acting as a client. The evaluator verified that those cipher suites exactly matched those listed for this component.

Furthermore, the evaluator verified that the "Cryptographic Support" and "Identification and Authentication" sections in the TSS of [ST] describes the client's method of establishing reference identifiers. The identity of the TLS server is verified from the X.509 certificate presented by the server using the guidelines in RFC 5280 and the rules specified in this SFR. The reference identifier is established from the CN or SAN of the server certificate. The aforementioned sections in the TSS of the [ST] also confirm that the TSF supports IP address and wildcards. Certificate pinning is not supported nor used by the product.

Finally, Section 5.2.2.12 of [ST] does not select the option for authorizing override of invalid certificates.

Guidance Assurance Activities

Assurance Activity AA-FCS_TLSC_EXT.1-AGD-01

FCS_TLSC_EXT.1.1: The evaluator shall also check the operational guidance to ensure that it contains instructions on configuring the product so that TLS conforms to the description in the TSS.

FCS_TLSC_EXT.1.2: The evaluator shall verify that the AGD guidance includes instructions for setting the reference identifier to be used for the purposes of certificate validation in TLS.

FCS_TLSC_EXT.1.3: No assurance activities defined.

Summary

The evaluator reviewed the operational guidance and found that the "Installing the TOE" section contains instructions on how to install the TOE, including the configuration of the TLS client version, cipher suites, and elliptic curves for key agreement. The reference identifier used for certificate validation is not configurable, as the TOE will always use the CN or SAN of the server certificate.

Test Assurance Activities

Assurance Activity AA-FCS_TLSC_EXT.1-ATE-01

FCS_TLSC_EXT.1.1: The evaluator shall also perform the following tests:

- Test 1: The evaluator shall establish a TLS connection using each of the cipher suites specified by the requirement. This connection may be established as part of the establishment of a higher-level protocol, e.g., as part of an EAP session. It is sufficient to observe the successful negotiation of a cipher suite to satisfy the intent of the test; it is not necessary to examine the characteristics of the encrypted traffic in an attempt to discern the cipher suite being used (for example, that the cryptographic algorithm is 128-bit AES and not 256-bit AES).
- Test 2: The goal of the following test is to verify that the TOE accepts only certificates with

appropriate values in the extendedKeyUsage extension, and implicitly that the TOE correctly parses the extendedKeyUsage extension as part of X.509v3 server certificate validation.

The evaluator shall attempt to establish the connection using a server with a server certificate that contains the Server Authentication purpose in the extendedKeyUsage extension and verify that a connection is established. The evaluator shall repeat this test using a different, but otherwise valid and trusted, certificate that lacks the Server Authentication purpose in the extendedKeyUsage extension and ensure that a connection is not established. Ideally, the two certificates should be similar in structure, the types of identifiers used, and the chain of trust.

- Test 3: The evaluator shall send a server certificate in the TLS connection that does not match the server-selected cipher suite (for example, send a ECDSA certificate while using the TLS_RSA_WITH_AES_128_CBC_SHA cipher suite or send a RSA certificate while using one of the ECDSA cipher suites.) The evaluator shall verify that the product disconnects after receiving the server's Certificate handshake message.
- Test 4: The evaluator shall configure the server to select the TLS_NULL_WITH_NULL_NULL cipher suite and verify that the client denies the connection.
- Test 5: The evaluator shall perform the following modifications to the traffic:
  - Test 5.1: Change the TLS version selected by the server in the Server Hello to an undefined TLS version (for example 1.5 represented by the two bytes 03 06) and verify that the client rejects the connection.
  - Test 5.2: Change the TLS version selected by the server in the Server Hello to the most recent unsupported TLS version (for example 1.1 represented by the two bytes 03 02) and verify that the client rejects the connection.
  - Test 5.3: [conditional] If DHE or ECDHE cipher suites are supported, modify at least one byte in the server's nonce in the Server Hello handshake message, and verify that the client does not complete the handshake and no application data flows.
  - Test 5.4: Modify the server's selected cipher suite in the Server Hello handshake message to be a cipher suite not presented in the Client Hello handshake message. The evaluator shall verify that the client does not complete the handshake and no application data flows.
  - Test 5.5: [conditional] If DHE or ECDHE cipher suites are supported, modify the signature block in the server's Key Exchange handshake message, and verify that the client does not complete the handshake and no application data flows. This test does not apply to cipher suites using RSA key exchange. If a TOE only supports RSA key exchange in conjunction with TLS, then this test shall be omitted.
  - Test 5.6: Modify a byte in the Server Finished handshake message, and verify that the client does not complete the handshake and no application data flows.

    Test 5.7: Send a message consisting of random bytes from the server after the server has issued the Change Cipher Spec message and verify that the client does not complete the handshake and no application data flows. The message must still have a valid 5-byte record header in order to ensure the message will be parsed as TLS.

FCS_TLSC_EXT.1.2: The evaluator shall configure the reference identifier according to the AGD guidance and perform the following tests during a TLS connection. If the TOE supports certificate pinning, all pinned certificates must be removed before performing Tests 1 through 6. A pinned certificate must be added prior to performing Test 7.

- Test 1: The evaluator shall present a server certificate that contains a CN that does not match the reference identifier and does not contain the SAN extension. The evaluator shall verify that the connection fails.

Note that some systems might require the presence of the SAN extension. In this case the connection would still fail but for the reason of the missing SAN extension instead of the mismatch of CN and reference identifier. Both reasons are acceptable to pass Test 1.

- Test 2: The evaluator shall present a server certificate that contains a CN that matches the reference identifier, contains the SAN extension, but does not contain an identifier in the SAN that matches the reference identifier. The evaluator shall verify that the connection fails. The

evaluator shall repeat this test for each supported SAN type.

- Test 3: [conditional] If the TOE does not mandate the presence of the SAN extension, the evaluator shall present a server certificate that contains a CN that matches the reference identifier and does not contain the SAN extension. The evaluator shall verify that the connection succeeds. If the TOE does mandate the presence of the SAN extension, this Test shall be omitted.
- Test 4: The evaluator shall present a server certificate that contains a CN that does not match the reference identifier but does contain an identifier in the SAN that matches. The evaluator shall verify that the connection succeeds.
- Test 5: The evaluator shall perform the following wildcard tests with each supported type of reference identifier. The support for wildcards is intended to be optional. If wildcards are supported, the first, second, and third tests below shall be executed. If wildcards are not supported, then the fourth test below shall be executed.
  - Test 5.1: [conditional]: If wildcards are supported, the evaluator shall present a server certificate containing a wildcard that is not in the left-most label of the presented identifier (e.g. foo.*.example.com) and verify that the connection fails.
  - Test 5.2: [conditional]: If wildcards are supported, the evaluator shall present a server certificate containing a wildcard in the left-most label but not preceding the public suffix (e.g. *.example.com). The evaluator shall configure the reference identifier with a single left-most label (e.g. foo.example.com) and verify that the connection succeeds. The evaluator shall configure the reference identifier without a left-most label as in the certificate (e.g. example.com) and verify that the connection fails. The evaluator shall configure the reference identifier with two left-most labels (e.g. bar.foo.example.come) and verify that the connection fails.
  - Test 5.3: [conditional]: If wildcards are supported, the evaluator shall present a server certificate containing a wildcard in the left-most label immediately preceding the public suffix (e.g. *.com). The evaluator shall configure the reference identifier with a single left-most label (e.g. foo.com) and verify that the connection fails. The evaluator shall configure the reference identifier with two left-most labels (e.g. bar.foo.com) and verify that the connection fails.
  - Test 5.4: [conditional]: If wildcards are not supported, the evaluator shall present a server certificate containing a wildcard in the left-most label (e.g. *.example.com). The evaluator shall configure the reference identifier with a single left-most label (e.g. foo.example.com) and verify that the connection fails.
- Test 6: [conditional] If URI or Service name reference identifiers are supported, the evaluator shall configure the DNS name and the service identifier. The evaluator shall present a server certificate containing the correct DNS name and service identifier in the URIName or SRVName fields of the SAN and verify that the connection succeeds. The evaluator shall repeat this test with the wrong service identifier (but correct DNS name) and verify that the connection fails.
- Test 7: [conditional] If pinned certificates are supported the evaluator shall present a certificate that does not match the pinned certificate and verify that the connection fails.

FCS_TLSC_EXT.1.3: [TD0513] The evaluator shall demonstrate that using an invalid certificate (unless excepted) results in the function failing as follows, unless excepted:

- **Test 1a:** The evaluator shall demonstrate that a server using a certifcate with a valid certification path successfully connects.
- **Test 1b:** The evaluator shall modify the certificate chain used by the server in test 1a to be invalid and demonstrate that a server using a certificate without a valid certification path to a trust store element of the TOE results in an authentication failure.
- **Test 1c [conditional]:** If the TOE trust store can be managed, the evaluator shall modify the trust store element used in Test 1a to be untrusted and demonstrate that a connection attempt from the same server used in Test 1a results in an authentication failure.
- **Test 2:** The evaluator shall demonstrate that a server using a certificate which has been revoked results in an authentication failure.

- **Test 3:** The evaluator shall demonstrate that a server using a certificate which has passed its expiration date results in an authentication failure.
- **Test 4:** The evaluator shall demonstrate that a server using a certificate which does not have a valid identifier results in an authentication failure.

Summary

The evaluator started a remote TLS server using the OpenSSL library. All packets were logged using tshark (part of the Wireshark tools).

Then, the evaluator used a test script to automatically update the hostname for the remote audit server to point to the TLS server. Additionally, the CA certificate used to issue the certificate of the TLS server was imported into the ESXi certificate store.

Finally, the `esxcli system syslog reload` command was used to initiate a TLS connection to the remote audit server (TLS server). Depending on the specific test, the connection succeeded or failed:

FCS_TLSC_EXT.1.1:

Test 1: The connection was successfully established using each of the supported cipher suites listed in the [ST]. The evaluator confirmed from the packet dump that the selected cipher suite was used.

Test 2: The evaluator created a server certificate with server authentication selected in the Extended Key Usage field and verified the connection was successfully established. The evaluator then created a second, otherwise identical, server certificate with server authentication *not* selected in the Extended Key Usage field and verified the connection was rejected as expected.

Test 3: The evaluator created an ECDSA-based server certificate but patched the OpenSSL TLS server to select an RSA-based cipher suite. The evaluator verified the connection was rejected as expected.

Test 4: The evaluator patched the OpenSSL TLS server to always select the TLS_NULL_WITH_NULL_NULL cipher suite, regardless of the support indicated by the client. The evaluator verified the connection was rejected as expected.

Test 5.1: The evaluator patched the OpenSSL TLS server to select the “03 05” TLS version to the client, corresponding to a non-existent TLS 1.4 version, regardless of the support indicated by the client. The evaluator verified the connection was rejected as expected.

Test 5.2: The evaluator patched the OpenSSL TLS server to initiate a TLS 1.1 connection, regardless of the support indicated by the client. The evaluator verified the connection was rejected as expected.

Test 5.3: The evaluator patched the OpenSSL TLS server to modify the nonce of the server. The evaluator verified the connection was rejected as expected.

Test 5.4: The evaluator patched the OpenSSL TLS server to modify the selected server cipher suite. The evaluator verified the connection was rejected as expected.

Test 5.5: The evaluator patched the OpenSSL TLS server to modify the signature in the Server Key Exchange message. The evaluator verified the connection was rejected as expected.

Test 5.6: The evaluator patched the OpenSSL TLS server to modify the message digest in the Server Finished message. The evaluator verified the connection was rejected as expected.

Test 5.7: The evaluator patched the OpenSSL TLS server to send another Server Key Exchange message after the Change Cipher Spec message. The evaluator verified the connection was rejected as expected.

FCS_TLSC_EXT.1.2:

Test 1: The evaluator created a server certificate with an incorrect Common Name and no Subject Alternative Name (SAN) extension and verified the connection was rejected as expected.

Test 2: The evaluator created a server certificate with a correct Common Name but with an incorrect DNS name entry in the SAN extension. The evaluator verified the connection was rejected as expected. The evaluator repeated this test for an incorrect IP address in the SAN extension.

Test 3: The [ST] indicates that the TOE does not mandate the presence of the SAN extension, and falls back to the Common Name if the SAN extension is not present. The evaluator created a server certificate with a correct Common Name and SAN extension and verified the connection was successfully established.

Test 4: The evaluator created a server certificate with an incorrect Common Name but with a correct DNS name entry in the SAN extension and verified the connection was successfully established.

Test 5.1: The evaluator created a server certificate with a “server.*.domain.com” entry in the Common Name and SAN extension. Then, the evaluator configured the TLS server to listen on “server.subdomain.domain.com” and initiated a TLS connection from the ESXi host. The evaluator verified the connection was rejected as expected.

Test 5.2: The evaluator created a server certificate with a “*.domain.com” entry in the Common Name and SAN extension. Then, the evaluator configured the TLS server to listen on “server.domain.com”, “domain.com”, and “server.subdomain.domain.com”, and initiated TLS connections from the ESXi host. The evaluator verified that the first connection corresponding to “server.domain.com” was successfully established, and the latter two were rejected as expected.

Test 5.3: The evaluator created a server certificate with a “*.com” entry in the Common Name and SAN extension. Then, the evaluator configured the TLS server to listen on “domain.com” and “server.domain.com”, and initiated TLS connections from the ESXi host. The evaluator verified the connections were rejected as expected.

Test 5.4: Not applicable as the TOE supports wildcards.

Test 6: Not applicable as the TOE does not support URI or Service name reference identifiers.

Test 7: Not applicable as the TOE does not support pinned certificates.

FCS_TLSC_EXT.1.3:

Test 1a: The evaluator created a valid server certificate and imported the issuer CA certificate into the ESXi certificate store, thereby creating a valid certification path. The evaluator verified the connection was successfully established.

Test 1b: The evaluator created a valid server certificate, identical to the certificate created in Test 1a, but using a different issuer CA certificate, thereby creating an invalid certification path. The evaluator verified the connection was rejected as expected.

Test 1c: The evaluator created a valid server certificate, identical to the certificate created in Test 1a, but without importing the issuer CA certificate into the ESXi certificate store, thereby creating an invalid certification path. The evaluator verified the connection was rejected as expected.

Test 2: The evaluator created a valid server certificate and imported the issuer CA certificate into the ESXi certificate store. Then, the evaluator created and hosted a certificate revocation list, with the server certificate marked as “revoked”. The evaluator verified the connection was rejected as expected.

Test 3: The evaluator created an expired server certificate and imported the issuer CA certificate into the ESXi certificate store. The evaluator verified the connection was rejected as expected.

Test 4: The evaluator created a server certificate with a correct Common Name but with an incorrect DNS name entry in the SAN extension and imported the issuer CA certificate into the ESXi certificate store. The evaluator verified the connection was rejected as expected.

2.2.2.13 TLS Client Support for Supported Groups Extension (FCS_TLSC_EXT.5)

TSS Assurance Activities

Assurance Activity AA-FCS_TLSC_EXT.5-ASE-01

The evaluator shall verify that TSS describes the Supported Groups Extension.

Summary

The evaluator verified that the “Cryptographic Support” section in the TSS of [ST] describes the Supported Groups Extension used for key establishment and its supported values.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-FCS_TLSC_EXT.5-ATE-01

The evaluator shall also perform the following test:

- Test 1: The evaluator shall configure a server to perform key exchange using each of the TOE’s supported curves and/or groups. The evaluator shall verify that the TOE successfully connects to the server.

Summary

The evaluator started a remote TLS server using the OpenSSL library. All packets were logged using tshark (part of the Wireshark tools). The server was configured to always select a specific curve.

Then, the evaluator used a test script to automatically update the hostname for the remote audit server to point to the TLS server. Additionally, the CA certificate used to issue the certificate of the TLS server was imported into the ESXi certificate store.

Finally, the `esxcli system syslog reload` command was used to initiate a TLS connection to the remote audit server (TLS server). The evaluator verified the connection was successfully established.

The steps above were repeated for each of the supported curves.

2.2.2.14 TLS Server Protocol (FCS_TLSS_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FCS_TLSS_EXT.1-ASE-01

FCS_TLSS_EXT.1.1: The evaluator shall check the description of the implementation of this protocol in the TSS to ensure that the cipher suites supported are specified. The evaluator shall

check the TSS to ensure that the cipher suites specified include those listed for this component.

FCS_TLSS_EXT.1.2: The evaluator shall verify that the TSS contains a description of the denial of old SSL and TLS versions consistent relative to selections in FCS_TLSS_EXT.1.2.

FCS_TLSS_EXT.1.3: The evaluator shall verify that the TSS describes the key agreement parameters of the server's Key Exchange message.

Summary

The evaluator verified that the “Cryptographic Support” section in the TSS of [ST] identifies the cipher suites supported by the TOE when acting as a server. The evaluator verified that those cipher suites exactly matched those listed for this component.

Furthermore, the evaluator verified that the aforementioned section states that all SSL and TLS versions older than TLS 1.2 are rejected by the TOE. The evaluator verified that those versions exactly matched those selected for this component.

Finally, the evaluator verified that the “Cryptographic Support” section describes the supported curves for ECC based key establishment.

Guidance Assurance Activities

Assurance Activity AA-FCS_TLSS_EXT.1-AGD-01

FCS_TLSS_EXT.1.1: The evaluator shall also check the operational guidance to ensure that it contains instructions on configuring the TOE so that TLS conforms to the description in the TSS.

FCS_TLSS_EXT.1.2: The evaluator shall verify that the AGD guidance includes any configuration necessary to meet this requirement.

FCS_TLSS_EXT.1.3: The evaluator shall verify that any configuration guidance necessary to meet the requirement must be contained in the AGD guidance.

Summary

The evaluator reviewed the operational guidance and found that the “Installing the TOE” section contains instructions on how to install the TOE, including the configuration of the TLS server version, cipher suites, and elliptic curves for key agreement.

Test Assurance Activities

Assurance Activity AA-FCS_TLSS_EXT.1-ATE-01

FCS_TLSS_EXT.1.1: The evaluator shall also perform the following tests:

- Test 1: The evaluator shall establish a TLS connection using each of the cipher suites specified by the requirement. This connection may be established as part of the establishment of a higher-level protocol, e.g., as part of an EAP session. It is sufficient to observe the successful negotiation of a cipher suite to satisfy the intent of the test; it is not necessary to examine the characteristics of the encrypted traffic in an attempt to discern the cipher suite being used (for example, that the cryptographic algorithm is 128-bit AES and not 256-bit AES).
- Test 2: The evaluator shall send a Client Hello to the server with a list of cipher suites that does not contain any of the cipher suites in the server’s ST and verify that the server denies the connection. Additionally, the evaluator shall send a Client Hello to the server containing only the TLS_NULL_WITH_NULL_NULL cipher suite and verify that the server denies the connection.
- Test 3: If RSA key exchange is used in one of the selected ciphersuites, the evaluator shall use a client to send a properly constructed Key Exchange message with a modified EncryptedPreMasterSecret field during the TLS handshake. The evaluator shall verify that the handshake is not completed successfully and no application data flows.
- Test 4: The evaluator shall perform the following modifications to the traffic:
  - Test 4.1: [TD0469] This test is removed.
  - Test 4.2: Modify a byte in the data of the client's Finished handshake message, and verify

that the server rejects the connection and does not send any application data.

- Test 4.3: [TD0779] Demonstrate that the TOE will not resume a session for which the client failed to complete the handshake (independent of TOE support for session resumption):
- Test 4.3i [conditional]: If the TOE does not support session resumption based on session IDs according to RFC4346 (TLS1.1) or RFC5246 (TLS1.2) or session tickets according to RFC5077, the evaluator shall perform the following test:
  - *a)* The evaluator shall send a Client Hello with a zero-length session identifier and with a SessionTicket extension containing a zero-length ticket.
  - *b)* The evaluator shall verify the server does not send a NewSessionTicket handshake message (at any point in the handshake).
  - *c)* The evaluator shall verify the Server Hello message contains a zero-length session identifier or passes the following steps:

Note: The following steps are only performed if the ServerHello message contains a non-zero length SessionID.

  - *d)* The evaluator shall complete the TLS handshake and capture the SessionID from the ServerHello.
  - *e)* The evaluator shall send a ClientHello containing the SessionID captured in step d). This can be done by keeping the TLS session in step d) open or start a new TLS session using the SessionID captured in step d).
  - *f)* The evaluator shall verify the TOE (1) implicitly rejects the SessionID by sending a ServerHello containing a different SessionID and by performing a full handshake (as shown in Figure 1 of RFC 4346 or RFC 5246), or (2) terminates the connection in some way that prevents the flow of application data.
- Test 4.3ii [conditional]: If the TOE supports session resumption using session IDs according to RFC4346 (TLS1.1) or RFC5246 (TLS1.2), the evaluator shall carry out the following steps (note that for each of these tests, it is not necessary to perform the test case for each supported version of TLS):
  - *a)* The evaluator shall conduct a successful handshake and capture the TOE-generated session ID in the Server Hello message. The evaluator shall then initiate a new TLS connection and send the previously captured session ID to show that the TOE resumed the previous session by responding with ServerHello containing the same SessionID immediately followed by ChangeCipherSpec and Finished messages (as shown in Figure 2 of RFC 4346 or RFC 5246).
  - *b)* The evaluator shall initiate a handshake and capture the TOE-generated session ID in the Server Hello message. The evaluator shall then, within the same handshake, generate or force an unencrypted fatal Alert message immediately before the client would otherwise send its ChangeCipherSpec message thereby disrupting the handshake. The evaluator shall then initiate a new Client Hello using the previously captured session ID, and verify that the server (1) implicitly rejects the session ID by sending a ServerHello containing a different SessionID and performing a full handshake (as shown in figure 1 of RFC 4346 or RFC 5246), or (2) terminates the connection in some way that prevents the flow of application data.
- Test 4.3iii [conditional]: If the TOE supports session tickets according to RFC5077, the evaluator shall carry out the following steps (note that for each of these tests, it is not necessary to perform the test case for each supported version of TLS):
  - *a)* The evaluator shall permit a successful TLS handshake to occur in which a session ticket is exchanged with the non-TOE client. The evaluator shall then attempt to correctly reuse the previous session by sending the session ticket in the ClientHello. The evaluator shall confirm that the TOE successfully resumes the session in accordance with section 3.1 of RFC 5077.
  - *b)* The evaluator shall permit a successful TLS handshake to occur in which a session ticket is exchanged with the non-TOE client. The evaluator will then modify the session ticket and send it as part of a new Client Hello message. The evaluator shall confirm that the

TOE either (1) implicitly rejects the session ticket by performing a full handshake (as shown in figure 3 or 4 of RFC 5077), or (2) terminates the connection in some way that prevents the flow of application data.

c) The evaluator shall send the TSF a Client Hello with a SessionTicket extension, and observe the TSF responds with a Server Hello with an empty SessionTicket extension. The evaluator shall then send the TSF a invalid Finished message, and observe that the TSF terminates the session without sending a valid newTicket message.

Note: if the TSF sends a newTicket message prior to terminating the session, the evaluator shall confirm the ticket is invalid by attempting to use the ticket to renew the session and observe that the TSF either (1) implicitly rejects the session ticket by performing a full handshake (as shown in figure 3 or 4 of RFC 5077), or (2) terminates the connection in some way that prevents the flow of application data.

- Test 4.4: Send a message consisting of random bytes from the client after the client has issued the ChangeCipherSpec message and verify that the server denies the connection.

FCS_TLSS_EXT.1.2:

- Test 1: The evaluator shall send a Client Hello requesting a connection with version SSL 2.0 and verify that the server denies the connection. The evaluator shall repeat this test with SSL 3.0 and TLS 1.0, and TLS 1.1 if it is selected.

FCS_TLSS_EXT.1.3: The evaluator shall conduct the following tests. The testing can be carried out manually with a packet analyzer or with an automated framework that similarly captures such empirical evidence. Note that this testing can be accomplished in conjunction with other testing activities. For each of the following tests, determining that the size matches the expected size is sufficient.

- Test 1: [conditional] If RSA-based key establishment is selected, the evaluator shall configure the TOE with a certificate containing a supported RSA size and attempt a connection. The evaluator shall verify that the size used matches that which is configured and that the connection is successfully established. The evaluator shall repeat this test for each supported size of RSA-based key establishment.
- Test 2: [conditional] If finite-field (i.e. non-EC) Diffie-Hellman ciphers are selected, the evaluator shall attempt a connection using a Diffie-Hellman key exchange with a supported parameter size or supported group. The evaluator shall verify that the key agreement parameters in the Key Exchange message are the ones configured. The evaluator shall repeat this test for each supported parameter size or group.
- Test 3: [conditional] If ECDHE ciphers are selected, the evaluator shall attempt a connection using an ECDHE ciphersuite with a supported curve. The evaluator shall verify that the key agreement parameters in the Key Exchange message are the ones configured. The evaluator shall repeat this test for each supported elliptic curve.

Summary

The evaluator used the Host Client provided by ESXi to test the TOE's built-in HTTPS server. Firstly, the evaluator configured the TOE to use a server private key and certificate generated by the evaluator.

Then, the evaluator connected to the Host Client using the OpenSSL library. All packets were logged using tshark (part of the Wireshark tools). Depending on the specific test, the connection succeeded or failed:

FCS_TLSS_EXT.1.1:

Test 1: The connection was successfully established using each of the supported cipher suites listed in the [ST]. The evaluator confirmed from the packet dump that the selected cipher suite was used.

Test 2: The connection was rejected as expected for each of the unsupported cipher suites (e.g., those using RSA-based key establishment). In addition, the evaluator patched the

OpenSSL client to always select the TLS_NULL_WITH_NULL_NULL cipher suite and verified the connection was rejected as expected.

Test 3: Not applicable as RSA-based key exchange is not used.

Test 4.1: [TD0469] This test is removed.

Test 4.2: The evaluator patched the OpenSSL client to modify the message digest in the Client Finished message. The evaluator verified the connection was rejected as expected.

Test 4.3i: Not applicable as the TOE supports session resumption.

Test 4.3ii: Not applicable as the TOE does not support session resumption using session IDs.

Test 4.3iii: The evaluator first connected and resumed the session to the TOE server with a valid session ticket. The evaluator verified the connection and session resumption were successful. The evaluator also verified that the TOE server implicitly rejects a modified session ticket by performing a full handshake. Then, the evaluator patched the OpenSSL client to send the Client Finished message before the Change Cipher Spec message, and verified the connection was rejected as expected. Finally, upon resumption of the interrupted session, the TOE server implicitly rejected the attempt by performing a full handshake.

Test 4.4: The evaluator patched the OpenSSL client to send a Next Protocol message after the Client Finished message. The evaluator verified the connection was rejected as expected.

FCS_TLSS_EXT.1.2:

Test 1: the evaluator verified that all connections for SSL 2.0, SSL 3.0, TLS 1.0, and TLS 1.1 were rejected as expected.

FCS_TLSS_EXT.1.3:

Test 1: Not applicable as RSA-based key exchange is not used.

Test 2: The connection was successfully established using each of the supported parameter sizes listed in the [ST]. The evaluator confirmed from the packet dump that the selected parameter size was used.

Test 3: Not applicable as FFC Diffie-Hellman key exchange is not used.

2.2.3 User data protection (FDP)

2.2.3.1 Hardware-Based Isolation Mechanisms (FDP_HBI_EXT.1)

TSS Assurance Activities

Assurance Activity AA-BVPP-FDP_HBI_EXT.1-ASE-01

> The evaluator shall ensure that the TSS provides evidence that hardware-based isolation mechanisms are used to constrain VMs when VMs have direct access to physical devices, including an explanation of the conditions under which the TSF invokes these protections.

Summary

The evaluator verified that the “User Data Protection” section in the TSS of [ST] lists the hardware-based isolation mechanisms (VT-x with EPT and VT-d) used to constrain VMs. The TSS further states that these protections are always active.

Guidance Assurance Activities

Assurance Activity AA-BVPP-FDP_HBI_EXT.1-AGD-01

> The evaluator shall verify that the operational guidance contains instructions on how to ensure that

the platform-provided, hardware-based mechanisms are enabled.

Summary

The evaluator verified that the “Hardware-based VM Isolation” section of the operational guidance states that the hardware-based VM isolation mechanisms are always enabled.

Test Assurance Activities

No assurance activities defined.

2.2.3.2 Physical Platform Resource Controls (FDP_PPR_EXT.1)

TSS Assurance Activities

Assurance Activity AA-BVPP-FDP_PPR_EXT.1-ASE-01

The evaluator shall examine the TSS to determine that it describes the mechanism by which the VMM controls a Guest VM's access to physical platform resources. This description shall cover all of the physical platforms allowed in the evaluated configuration by the ST. It should explain how the VMM distinguishes among Guest VMs, and how each physical platform resource that is controllable (that is, listed in the assignment statement in the first element) is identified to an Administrator.

The evaluator shall ensure that the TSS describes how the Guest VM is associated with each physical resource, and how other Guest VMs cannot access a physical resource without being granted explicit access. For TOEs that implement a robust interface (other than just "allow access" or "deny access"), the evaluator shall ensure that the TSS describes the possible operations or modes of access between a Guest VM's and physical platform resources.

If physical resources are listed in the second element, the evaluator shall examine the TSS and operational guidance to determine that there appears to be no way to configure those resources for access by a Guest VM. The evaluator shall document in the evaluation report their analysis of why the controls offered to configure access to physical resources can't be used to specify access to the resources identified in the second element (for example, if the interface offers a drop-down list of resources to assign, and the denied resources are not included on that list, that would be sufficient justification in the evaluation report).

Summary

The evaluator verified that the “User Data Protection” section in the TSS of [ST] lists the hardware-based isolation mechanisms (VT-x with EPT and VT-d) used to control Guest VM access to physical platform resources. This covers all physical platforms allowed in the evaluated configuration. Furthermore, this section lists two device classes (USB devices and network adapters) which can be configured to be accessed by Guest VMs. The configuration is described in terms of a security policy on a global and per-VM configuration.

Furthermore, the evaluator verified that the aforementioned section describes the modes of access between Guest VMs and physical platform resources: the global and per-VM configuration options resolve to a simple allow/deny access when the VM attempts to access the resource.

Finally, the second element lists two physical resources: PCI passthrough and SCSI passthrough. The “User Data Protection” section states that, in the evaluated configuration, Guest VM access to PCI passthrough devices or raw device mappings to storage LUNs is blocked. The evaluator confirmed that it is impossible to assign PCI passthrough devices (the “PCI device” option is grayed out in the VM settings) or raw disks (the “New raw disk” option is grayed out in the VM settings).

Guidance Assurance Activities

Assurance Activity AA-BVPP-FDP_PPR_EXT.1-AGD-01

The evaluator shall examine the operational guidance to determine that it describes how an administrator is able to configure access to physical platform resources for Guest VMs for each

platform allowed in the evaluated configuration according to the ST. The evaluator shall also determine that the operational guidance identifies those resources listed in the second and third elements of the component and notes that access to these resources is explicitly denied/allowed, respectively.

Summary

The evaluator confirmed that the “Protection of User (VM) Data (FDP)” section of the operational guidance, specifically “USB” and “Physical Network”, describes how an administrator can configure Guest VMs to have access to USB devices and physical network interfaces, respectively. Moreover, this section in “Physical Platform Resources” also states that raw disks and other devices (such as PCI passthrough devices, vGPU devices, and SCSI passthrough devices) are not to be used. There are no physical platform resources always explicitly allowed to all Guest VMs.

Test Assurance Activities

Assurance Activity AA-BVPP-FDP_PPR_EXT.1-ATE-01

Using the operational guidance, the evaluator shall perform the following tests for each physical platform identified in the ST:

- Test 1: For each physical platform resource identified in the first element, the evaluator shall configure a Guest VM to have access to that resource and show that the Guest VM is able to successfully access that resource.
- Test 2: For each physical platform resource identified in the first element, the evaluator shall configure the system such that a Guest VM does not have access to that resource and show that the Guest VM is unable to successfully access that resource.
- Test 3: [conditional]: For TOEs that have a robust control interface, the evaluator shall exercise each element of the interface as described in the TSS and the operational guidance to ensure that the behavior described in the operational guidance is exhibited.
- Test 4: [conditional]: If the TOE explicitly denies access to certain physical resources, the evaluator shall attempt to access each listed (in FDP_PPR_EXT.1.2) physical resource from a Guest VM and observe that access is denied.
- Test 5: [conditional]: If the TOE explicitly allows access to certain physical resources, the evaluator shall attempt to access each listed (in FDP_PPR_EXT.1.3) physical resource from a Guest VM and observe that the access is allowed. If the operational guidance specifies that access is allowed simultaneously by more than one Guest VM, the evaluator shall attempt to access each resource listed from more than one Guest VM and show that access is allowed.

Summary

The evaluator set up a Guest VM using Ubuntu 24.04.1 and connected a USB drive to the physical system on which ESXi executes.

Test 1: the evaluator used the Host Client to attach a virtual NIC (corresponding to a physical network adapter) and then the USB drive to the Guest VM. The evaluator verified that the Guest VM could access the network and read/write from the USB drive.

Test 2: the evaluator used the Host Client to remove the USB drive and then the virtual NIC from the Guest VM. The evaluator verified the Guest VM could no longer read/write from the USB drive nor access the network.

Test 3: not applicable as the TOE does not implement a robust control interface.

Test 4: the evaluator used the lspci and lssci commands to verify that the Guest VM does not have access to any physical PCI devices or physical disks from the ESXi host.

Test 5: not applicable as the TOE does not explicitly allow access to any physical resources.

2.2.3.3 Residual Information in Memory (FDP_RIP_EXT.1)

TSS Assurance Activities

Assurance Activity AA-BVPP-FDP_RIP_EXT.1-ASE-01

The evaluator shall ensure that the TSS documents the process used for clearing physical memory prior to allocation to a Guest VM, providing details on when and how this is performed. Additionally, the evaluator shall ensure that the TSS documents the conditions under which physical memory is not cleared prior to allocation to a Guest VM, and describes when and how the memory is cleared.

Summary

The evaluator verified that the “User Data Protection” section in the TSS of [ST] describes how volatile memory is cleared before being allocated to Guest VMs. Memory pages are overwritten by zeroes, either prior to being allocated to a VM or prior to de-allocation or migration from a VM. By default, the Mem.MemEagerZero advanced option is set to zero, which means zeroizing happens only when pages are allocated to Guest VMs; pages are not zeroed when de-allocated from Guest VMs by default. In any case, the volatile memory is always set to 0 prior to allocation to a Guest VM.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

No assurance activities defined.

2.2.3.4 Residual Information on Disk (FDP_RIP_EXT.2)

TSS Assurance Activities

Assurance Activity AA-BVPP-FDP_RIP_EXT.2-ASE-01

The evaluator shall ensure that the TSS documents how the TSF ensures that disk storage is zeroed upon allocation to Guest VMs. Also, the TSS must document any conditions under which disk storage is not cleared prior to allocation to a Guest VM. Any file system format and metadata information needed by the evaluator to perform the below test shall be made available to the evaluator, but need not be published in the TSS.

Summary

The evaluator verified that the “User Data Protection” section in the TSS of [ST] describes how disk storage is cleared. For performance reasons, the TOE defines different provisioning policies. In the case of Thick Provision Lazy Zeroed, the disk storage is not zeroized prior to provisioning, but only prior to access by the Guest VM. In the case of Thick Provision Eager Zeroed, the disk storage is zeroized prior to allocation. Finally, Thin Provisioned allocates and zeroes storage blocks upon access. From the VM perspective, there is no difference, as zeroization will always have happened prior to accessing the storage blocks.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-BVPP-FDP_RIP_EXT.2-ATE-01

The evaluator shall perform the following test:

- Test 1: On the host, the evaluator creates a file that is more than half the size of a connected physical storage device (or multiple files whose individual sizes add up to more than half the size of the storage media). This file (or files) shall be filled entirely with a non-zero value. Then, the file (or files) shall be released (freed for use but not cleared). Next, the evaluator (as a VS Administrator) creates a virtual disk at least that large on the same

	physical storage device and connects it to a powered-off VM. Then, from outside the Guest VM, scan through and check that all the non-metadata (as documented in the TSS) in the file corresponding to that virtual disk is set to zero.

Summary

The evaluator created a new VMFS datastore approximately 4 GiB in size, then created a random test file that was 2 GiB in size. The evaluator confirmed at least 50% of the datastore was marked as used. Finally, the evaluator removed the test file and created a new Guest VM, which was allocated a 2 GiB disk image. The evaluator verified that the disk image was set to zeroes.

2.2.3.5 VM Separation (FDP_VMS_EXT.1)

TSS Assurance Activities

Assurance Activity AA-BVPP-FDP_VMS_EXT.1-ASE-01

The evaluator shall examine the TSS to verify that it documents all inter-VM communications mechanisms (as defined above), and explains how the TSF prevents the transfer of data between VMs outside of the mechanisms listed in FDP_VMS_EXT.1.1.

Summary

The evaluator verified that the “User Data Protection” section in the TSS of [ST] documents the only supported inter-VM communication mechanism: virtual networking. It also explicitly states that there are no other ways to access data or transfer data between VMs, other than the virtual networking mechanism.

Guidance Assurance Activities

Assurance Activity AA-BVPP-FDP_VMS_EXT.1-AGD-01

The evaluator shall examine the operational guidance to ensure that it documents how to configure all inter-VM communications mechanisms, including how they are invoked and how they are disabled.

Summary

The evaluator reviewed the operational guidance and found that the “Virtual Networking” section describes how to configure inter-VM communications (virtual machine networking) and how to disable this mechanism (by disabling access to a virtual switch).

Test Assurance Activities

Assurance Activity AA-BVPP-FDP_VMS_EXT.1-ATE-01

The evaluator shall perform the following tests for each documented inter-VM communications channel:

- Test 1:

a) Create two VMs without specifying any communications mechanism or overriding the default configuration.

b) Test that the two VMs cannot communicate through the mechanisms selected in FDP_VMS_EXT.1.1.

c) Create two new VMs, overriding the default configuration to allow communications through a channel selected in FDP_VMS_EXT.1.1.

d) Test that communications can be passed between the VMs through the channel.

e) Create two new VMs, the first with the inter-VM communications channel currently being tested enabled, and the second with the inter-VM communications channel currently being tested disabled.

f) Test that communications cannot be passed between the VMs through the channel.

g) As an Administrator, enable inter-VM communications between the VMs on the second VM.

h) Test that communications can be passed through the inter-VM channel.

i) As an Administrator again, disable inter-VM communications between the two VMs.

j) Test that communications can no longer be passed through the channel.

FDP_VMS_EXT.1.2 is met if communication is unsuccessful in step (b). FDP_VMS_EXT.1.3 is met if communication is successful in step (d) and unsuccessful in step (f).

Summary

The evaluator set up two identical Guest VMs using Ubuntu 24.04.1, assigned but not connected to the VM Network. The evaluator used ping to attempt to communicate between the two VMs, which failed. Then, the evaluator set up two new, identical Guest VMs using Ubuntu 24.04.1. Both were assigned and connected to the VM Network. The evaluator verified successful communication using ping. This process was repeated, connecting and disconnecting the VMs from the VM Network as required by the assurance activity. The evaluator found that communications were successful and unsuccessful as expected.

2.2.3.6 Virtual Networking Components (FDP_VNC_EXT.1)

TSS Assurance Activities

Assurance Activity AA-BVPP-FDP_VNC_EXT.1-ASE-01

The evaluator shall examine the TSS (or a proprietary annex) to verify that it describes the mechanism by which virtual network traffic is ensured to be visible only to Guest VMs configured to be on that virtual network.

Summary

The evaluator verified that the “User Data Protection” section in the TSS of [ST] states that traffic traversing a virtual network is visible only to Guest VMs that are members of the virtual network. This is enforced by the TOE encapsulating the packets using VLANs. Guest VMs are only connected to a virtual network when explicitly configured by the Administrator.

Guidance Assurance Activities

Assurance Activity AA-BVPP-FDP_VNC_EXT.1-AGD-01

The evaluator must ensure that the Operational Guidance describes how to create virtualized networks and connect VMs to each other and to physical networks.

Summary

The evaluator reviewed the operational guidance and found that the “Virtual Networking” section describes how to configure virtual machine networking and how to connect VMs to each other (using virtual switches) and to physical networks (using virtual NICs).

Test Assurance Activities

Assurance Activity AA-BVPP-FDP_VNC_EXT.1-ATE-01

- Test 1: The evaluator shall assume the role of the Administrator and attempt to configure a VM to connect to a network component. The evaluator shall verify that the attempt is successful. The evaluator shall then assume the role of an unprivileged user and attempt the same connection. If the attempt fails, or there is no way for an unprivileged user to configure VM network connections, the requirement is met.
- Test 2: The evaluator shall assume the role of the Administrator and attempt to configure a VM to connect to a physical network. The evaluator shall verify that the attempt is successful. The evaluator shall then assume the role of an unprivileged user and make the same attempt. If the attempt fails, or there is no way for an unprivileged user to configure

VM network connections, the requirement is met.

Summary

Test 1: the evaluator first logged in to the Host Client using the administrator (root) account, verified it was possible to change the settings for the VM, and added it to the virtual VM Network. Then, the evaluator logged in to the Host Client using a non-administrator account and verified it was not possible to change any settings for the VM, including network settings.

Test 2: the evaluator first logged in to the Host Client using the administrator (root) account, verified it was possible to create a new virtual switch, and added it to a physical uplink. Then, the evaluator logged in to the Host Client using a non-administrator account and verified it was not possible to create a new virtual switch, thus preventing connections to physical networks.

2.2.4 Identification and authentication (FIA)

2.2.4.1 Authentication Failure Handling (FIA_AFL_EXT.1)

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-BVPP-FIA_AFL_EXT.1-ATE-01

The evaluator shall perform the following tests for each credential selected in FIA_AFL_EXT.1.1:

The evaluator will set an Administrator-configurable threshold n for failed attempts, or note the ST-specified assignment.

- Test 1: The evaluator will attempt to authenticate remotely with the credential n-1 times. The evaluator will then attempt to authenticate using a good credential and verify that authentication is successful.
- Test 2: The evaluator will make n attempts to authenticate using a bad credential. The evaluator will then attempt to authenticate using a good credential and verify that the attempt is unsuccessful. Note that the authentication attempts and lockouts must also be logged as specified in FAU_GEN.1.

After reaching the limit for unsuccessful authentication attempts the evaluator will proceed as follows:

- Test 1: If the Administrator action selection in FIA_AFL_EXT.1.2 is selected, then the evaluator will confirm by testing that following the operational guidance and performing each action specified in the ST to re-enable the remote Administrator's access results in successful access (when using valid credentials for that Administrator).
- Test 2: If the time period selection in FIA_AFL_EXT.1.2 is selected, the evaluator will wait for just less than the time period configured and show that an authentication attempt using valid credentials does not result in successful access. The evaluator will then wait until just after the time period configured and show that an authentication attempt using valid credentials results in successful access.

Summary

The evaluator set up a test account for this assurance activity, then configured the ESXi to lock the user account for 90 seconds after 5 failed login attempts.

FIA_AFL_EXT.1.1:

Test 1: using the VIM API, the evaluator made four attempts to log in to the test account using an incorrect password. Then, the evaluator attempted to log in to the test account with the correct password, and verified it succeeded.

Test 2: using the VIM API, the evaluator made five attempts to log in to the test account using an incorrect password. Then, the evaluator attempted to log in to the test account with the correct password, and verified it failed (as the account was locked). The evaluator also verified the authentication attempts and account locking were logged.

FIA_AFL_EXT.1.2:

Test 1: not applicable, as the Administrator action is not selected in FIA_AFL_EXT.1.2.

Test 2: using the VIM API, the evaluator made five attempts to log in to the test account using an incorrect password. Then, the evaluator waited for 80 seconds and attempted to log in to the test account with the correct password, and verified it failed (as the account was still locked). Finally, the evaluator waited for another 95 seconds and attempted to log in to the test account with the correct password again. This succeeded.

2.2.4.2 Password Management (FIA_PMG_EXT.1)

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

Assurance Activity AA-BVPP-FIA_PMG_EXT.1-AGD-01

The evaluator shall examine the operational guidance to determine that it provides guidance to security administrators in the composition of strong passwords, and that it provides instructions on setting the minimum password length.

Summary

The evaluator examined the operational guidance and found that the “Password Management” section provides guidance for administrators to set strong passwords, including the use of the Security.PasswordQualityControl setting. This setting should require a mix of characters from multiple character classes, and a minimum password length that satisfies the deployment’s password requirements.

Test Assurance Activities

Assurance Activity AA-BVPP-FIA_PMG_EXT.1-ATE-01

The evaluator shall also perform the following test.

- Test 1: The evaluator shall compose passwords that either meet the requirements, or fail to meet the requirements, in some way. For each password, the evaluator shall verify that the TOE supports the password. While the evaluator is not required (nor is it feasible) to test all possible combinations of passwords, the evaluator shall ensure that all characters, rule characteristics, and a minimum length listed in the requirement are supported, and justify the subset of those characters chosen for testing.

Summary

The evaluator composed one compliant password and seven non-compliant passwords to test different use cases against the TOE. The non-compliant passwords covered the following password policy exceptions: same password; no special characters; all numbers; no numbers; no lowercase letters; too few characters; no lowercase characters after changing policy. The evaluator found that all attempts to configure a non-compliant password failed.

2.2.4.3 Multiple Authentication Mechanisms (FIA_UAU.5)

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-BVPP-FIA_UAU.5-ATE-01

If ‘username and password authentication‘ is selected, the evaluator will configure the VS with a known username and password and conduct the following tests:

- Test 1: The evaluator will attempt to authenticate to the VS using the known username and password. The evaluator will ensure that the authentication attempt is successful.
- Test 2: The evaluator will attempt to authenticate to the VS using the known username but an incorrect password. The evaluator will ensure that the authentication attempt is unsuccessful.

If ‘username and PIN that releases an asymmetric key‘ is selected, the evaluator will examine the TSS for guidance on supported protected storage and will then configure the TOE or OE to establish a PIN which enables release of the asymmetric key from the protected storage (such as a TPM, a hardware token, or isolated execution environment) with which the VS can interface. The evaluator will then conduct the following tests:

- Test 1: The evaluator will attempt to authenticate to the VS using the known user name and PIN. The evaluator will ensure that the authentication attempt is successful.
- Test 2: The evaluator will attempt to authenticate to the VS using the known user name but an incorrect PIN. The evaluator will ensure that the authentication attempt is unsuccessful.

If ‘X.509 certificate authentication‘ is selected, the evaluator will generate an X.509v3 certificate for an Administrator user with the Client Authentication Enhanced Key Usage field set. The evaluator will provision the VS for authentication with the X.509v3 certificate. The evaluator will ensure that the certificates are validated by the VS as per FIA_X509_EXT.1.1 and then conduct the following tests:

- Test 1: The evaluator will attempt to authenticate to the VS using the X.509v3 certificate. The evaluator will ensure that the authentication attempt is successful.
- Test 2: The evaluator will generate a second certificate identical to the first except for the public key and any values derived from the public key. The evaluator will attempt to authenticate to the VS with this certificate. The evaluator will ensure that the authentication attempt is unsuccessful.

If ‘SSH public-key credential authentication‘ is selected, the evaluator shall generate a public-private host key pair on the TOE using RSA or ECDSA, and a second public-private key pair on a remote client. The evaluator shall provision the VS with the client public key for authentication over SSH, and conduct the following tests:

- Test 1: The evaluator will attempt to authenticate to the VS using a message signed by the client private key that corresponds to provisioned client public key. The evaluator will ensure that the authentication attempt is successful.
- Test 2: The evaluator will generate a second client key pair and will attempt to authenticate to the VS with the private key over SSH without first provisioning the VS to support the new key pair. The evaluator will ensure that the authentication attempt is unsuccessful.

Summary

The evaluator set up a test account for this assurance activity. Only “username and password authentication” is selected, so the applicable tests are executed:

Test 1: the evaluator used the VIM API to log in to the test account using the correct password. The login attempt succeeded.

Test 2: the evaluator used the VIM API to log in to the test account using an incorrect password. The login attempt failed.

2.2.4.4 Administrator Identification and Authentication (FIA_UIA_EXT.1)

TSS Assurance Activities

Assurance Activity AA-BVPP-FIA_UIA_EXT.1-ASE-01

The evaluator shall examine the TSS to determine that it describes the logon process for each logon method (local, remote (HTTPS, SSH, etc.)) supported for the product. This description shall contain information pertaining to the credentials allowed/used, any protocol transactions that take place, and what constitutes a “successful logon.”

Summary

The evaluator verified that the “Trusted Path/Channels” section in the TSS of [ST] lists the two logon methods available to the Administrator: the VIM API or ESXCLI. Both of these interfaces use TLS/HTTPS. Furthermore, the evaluator verified that the “Identification and Authentication” section in the TSS of [ST] describes that username and password-based authentication is implemented, and a successful login is determined by matching the provided username and password with the ones managed by the TOE. Subsequent TLS connections use session tokens (HTTP cookies) to maintain the session.

Guidance Assurance Activities

Assurance Activity AA-BVPP-FIA_UIA_EXT.1-AGD-01

[TD0814] The evaluator shall examine the operational guidance to determine that any necessary preparatory steps (e.g., establishing credential material such as pre-shared keys, tunnels, certificates) to logging in are described. For each supported login method, the evaluator shall ensure the operational guidance provides clear instructions for successfully logging on. If configuration is necessary to ensure the services provided before login are limited, the evaluator shall determine that the operational guidance provides sufficient instruction on limiting the allowed services.

Summary

The evaluator examined the operational guidance and verified that the “Establishing Remote Administrative Sessions” section provides clear instructions for successfully logging in via each of the TOE’s supported login mechanisms (VIM API and ESXCLI), including the necessary preparatory steps prior to logging in.

The TOE does not provide any management service prior to login.

Test Assurance Activities

No assurance activities defined.

2.2.4.5 X.509 Certificate Validation (FIA_X509_EXT.1)

TSS Assurance Activities

Assurance Activity AA-BVPP-FIA_X509_EXT.1-ASE-01

The evaluator shall ensure the TSS describes where the check of validity of the certificates takes place. The evaluator ensures the TSS also provides a description of the certificate path validation algorithm.

The evaluator shall examine the TSS to confirm that it describes the behavior of the TOE when a connection cannot be established during the validity check of a certificate used in establishing a trusted channel. If the requirement that the administrator is able to specify the default action, then

the evaluator shall ensure that the operational guidance contains instructions on how this configuration action is performed.

Summary

The evaluator verified that the “Identification and Authentication” section in the TSS of [ST] describes the X.509 certificate validation checks. Validation occurs as part of the TLS client when verifying a TLS server certificate. The TSS also describes the certificate path validation algorithm: a chain of trust is established from the repository of trusted certificate authorities to the certificate to be validated.

Further, the TSS states that certificate revocation checking is implemented using certificate revocation lists (CRLs). If a connection cannot be established, i.e., the CRL cannot be downloaded, the TLS connection is always terminated. There are no configurable actions.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-BVPP-FIA_X509_EXT.1-ATE-01

[TD0905] The tests described must be performed in conjunction with the other Certificate Services evaluation activities, including the uses listed in FIA_X509_EXT.2.1. The tests for the extendedKeyUsage rules are performed in conjunction with the uses that require those rules.

- Test 1: The evaluator shall demonstrate that validating a certificate without a valid certification path results in the function failing, for each of the following reasons, in turn:
  - by establishing a certificate path in which one of the issuing certificates is not a CA certificate,
  - by omitting the basicConstraints field in one of the issuing certificates,
  - by setting the basicConstraints field in an issuing certificate to have CA=False,
  - by omitting the CA signing bit of the key usage field in an issuing certificate, and
  - by setting the path length field of a valid CA field to a value strictly less than the certificate path.

The evaluator shall then establish a valid certificate path consisting of valid CA certificates, and demonstrate that the function succeeds. The evaluator shall then remove trust in one of the CA certificates, and show that the function fails.

- Test 2: The evaluator shall demonstrate that validating an expired certificate results in the function failing.
- Test 3: (conditional, performed except for use cases identified in exceptions that cannot be configured to allow revocation) The evaluator shall test that the TOE can properly handle revoked certificates – conditional on whether CRL, OCSP, OCSP stapling, or OCSP multi-stapling is selected; if multiple methods are selected, then a test is performed for each method. The evaluator has to only test one up in the trust chain (future revisions may require to ensure the validation is done up the entire chain). The evaluator shall ensure that a valid certificate is used, and that the validation function succeeds. The evaluator shall then attempt the test with a certificate that will be revoked (for each method chosen in the selection) and verify that the validation function fails. If the exceptions are configurable, the evaluator shall attempt to configure the exceptions to allow revocation checking for each function indicated in FIA_X509_EXT.2.
- Test 4: If any OCSP option is selected, the evaluator shall present a delegated OCSP certificate that does not have the OCSP signing purpose and verify that validation of the OCSP response fails. If CRL is selected, the evaluator shall configure the CA to sign a CRL with a certificate that does not have the cRLsign key usage bit set and verify that validation of the CRL fails.
- Test 5: (Conditional on support for EC certificates as indicated in FCS_COP.1/SIG). The

evaluator shall establish a valid, trusted certificate chain consisting of an EC leaf certificate, an EC Intermediate CA certificate not designated as a trust anchor, and an EC certificate designated as a trusted anchor, where the elliptic curve parameters are specified as a named curve. The evaluator shall confirm that the TOE validates the certificate chain.

- Test 6: (Conditional on support for EC certificates as indicated in FCS_COP.1/SIG). The evaluator shall replace the intermediate certificate in the certificate chain for Test 5 with a modified certificate, where the modified intermediate CA has a public key information field where the EC parameters uses an explicit format version of the Elliptic Curve parameters in the public key information field of the intermediate CA certificate from Test 5, and the modified Intermediate CA certificate is signed by the trusted EC root CA, but having no other changes. The evaluator shall confirm the TOE treats the certificate as invalid.

Summary

The evaluator started a remote TLS server using the OpenSSL library. All packets were logged using tshark (part of the Wireshark tools).

Then, the evaluator used a test script to automatically update the hostname for the remote audit server to point to the TLS server. Additionally, the root CA of a certificate chain consisting of four certificates (one root CA, two intermediate CAs, and one server node certificate) was imported into the ESXi certificate store.

Finally, the `esxcli system syslog reload` command was used to initiate a TLS connection to the remote audit server (TLS server). Depending on the specific test, the connection succeeded or failed:

Test 1a: The evaluator created an (otherwise valid) intermediate CA certificate with a basicConstraints extension containing the “CA=false” entry. The evaluator verified the connection was rejected as expected, indicating that the certificate chain validation failed.

Test 1b: The evaluator created an (otherwise valid) intermediate CA certificate without a basicConstraints extension. The evaluator verified the connection was rejected as expected, indicating that the certificate chain validation failed.

Test 1c: The evaluator created an (otherwise valid) intermediate CA certificate with a basicConstraints extension containing the “CA=false” entry. The evaluator verified the connection was rejected as expected, indicating that the certificate chain validation failed.

Test 1d: The evaluator created an (otherwise valid) intermediate CA certificate without the certificate signing purpose set in the keyUsage extension. The evaluator verified the connection was rejected as expected, indicating that the certificate chain validation failed.

Test 1e: The evaluator created an (otherwise valid) intermediate CA certificate with a path length set to 0. The evaluator verified the connection was rejected as expected, indicating that the certificate chain validation failed.

Test 1f: The evaluator created a valid certificate chain. The evaluator verified the connection was successfully established, indicating that the certificate chain validation succeeded. The evaluator then removed an intermediate certificate from the chain and verified the connection was rejected as expected, indicating that the certificate chain validation failed.

Test 2: The evaluator created an (otherwise valid) expired intermediate CA certificate. The evaluator verified the connection was rejected as expected, indicating that the certificate chain validation failed.

Test 3: The evaluator created a valid certificate chain. Then, the evaluator created and hosted a certificate revocation list, with the server node certificate marked as “valid”. The evaluator verified the connection was accepted as expected. Finally, the tester configured the server node certificate to be marked as “revoked” and verified the connection was rejected as expected.

Test 4: The evaluator created a valid certificate chain. Then, the evaluator created and hosted a certificate revocation list, with the server node certificate marked as “valid”, but did not add the CRL signing purpose to the certificate used to sign the CRL. The evaluator verified the connection was rejected as expected.

Test 5: The evaluator created a valid certificate chain, using named elliptic curve parameters for each certificate. The evaluator verified the connection was successfully established, indicating that the certificate chain validation succeeded. These steps were repeated for each of the supported curves.

Test 6: The evaluator created a valid certificate chain, using explicit elliptic curve parameters for an intermediate CA certificate. The evaluator verified the connection was rejected as expected, indicating that the certificate chain validation failed. These steps were repeated for each of the supported curves.

2.2.4.6 X.509 Certificate Authentication (FIA_X509_EXT.2)

TSS Assurance Activities

Assurance Activity AA-BVPP-FIA_X509_EXT.2-ASE-01

The evaluator shall check the TSS to ensure that it describes how the TOE chooses which certificates to use, and any necessary instructions in the administrative guidance for configuring the operating environment so that the TOE can use the certificates.

The evaluator shall examine the TSS to confirm that it describes the behavior of the TOE when a connection cannot be established during the validity check of a certificate used in establishing a trusted channel. If the requirement states that the administrator specifies the default action, then the evaluator shall ensure that the operational guidance contains instructions on how this configuration action is performed.

Summary

The evaluator verified that the “Identification and Authentication” section in the TSS of [ST] describes how the TOE chooses which certificates to use: the TOE maintains a repository of trusted certificate authorities. The “Configure Auditing” and “X.509 Certificate Validation and Authentication” sections of the administrative guidance describe how this repository can be updated.

Further, the TSS states that certificate revocation checking is implemented using certificate revocation lists (CRLs). If a connection cannot be established, i.e., the CRL cannot be downloaded, the TLS connection is always terminated. There are no configurable actions.

Guidance Assurance Activities

Assurance Activity AA-BVPP-FIA_X509_EXT.2-AGD-01

[TD0814] The evaluator shall ensure that the guidance documentation describes the configuration required in the operating environment so the TOE can use the certificates. The guidance documentation shall also include any required configuration on the TOE to use the certificates. The guidance documentation shall also describe the steps for the Security Administrator to follow if the connection cannot be established during the validity check of a certificate used in establishing a trusted channel.

Summary

The evaluator confirmed that the “Configure Auditing” and “X.509 Certificate Validation and Authentication” sections of the guidance documentation describe how to manipulate the certificate store containing the trusted certificate authorities. The ESXCLI command (certificatestore) or VIM API HostCertificateManager can list the existing certificate authorities, as well as add, list, or replace certificate authorities.

If a connection cannot be established during the revocation check of a certificate, the TLS connection is always terminated. There are no configurable actions and no steps for the administrator to follow.

Test Assurance Activities

Assurance Activity AA-BVPP-FIA_X509_EXT.2-ATE-01

The evaluator shall perform Test 1 for each function listed in FIA_X509_EXT.2.1 that requires the use of certificates:

- Test 1: The evaluator shall demonstrate that using a certificate without a valid certification path results in the function failing. Using the administrative guidance, the evaluator shall then load a certificate or certificates needed to validate the certificate to be used in the function, and demonstrate that the function succeeds. The evaluator then shall delete one of the certificates, and show that the function fails.
- Test 2: The evaluator shall demonstrate that using a valid certificate requires that certificate validation checking be performed in at least some part by communicating with a non-TOE IT entity. The evaluator shall then manipulate the environment so that the TOE is unable to verify the validity of the certificate, and observe that the action selected in FIA_X509_EXT.2.2 is performed. If the selected action is administrator-configurable, then the evaluator shall follow the operational guidance to determine that all supported administrator-configurable options behave in their documented manner.

Summary

The evaluator started a remote TLS server using the OpenSSL library. All packets were logged using tshark (part of the Wireshark tools).

Then, the evaluator used a test script to automatically update the hostname for the remote audit server to point to the TLS server. The CA certificate was not imported into the ESXi certificate store.

Finally, the `esxcli system syslog reload` command was used to initiate a TLS connection to the remote audit server (TLS server). Depending on the specific test, the connection succeeded or failed:

Test 1: The evaluator observed that the server certificate failed to validate. The steps above were repeated twice more: once after importing the CA certificate used to issue the certificate of the TLS server (resulting in a successful TLS connection), and once again after removing the CA certificate (resulting in an unsuccessful TLS connection).

Test 2: The evaluator created a valid certificate chain consisting of four certificates (one root CA, two intermediate CAs, and one server node certificate). The root certificate of the chain was imported into the ESXi certificate store. Then, the evaluator created and hosted a certificate revocation list, with the server node certificate marked as “valid”. The evaluator verified the connection was accepted as expected. Finally, the tester stopped the CRL hosting server and verified the connection was rejected as expected (as the TOE could not retrieve the CRL).

2.2.5 Security management (FMT)

2.2.5.1 Management of Security Functions Behavior (FMT_MOF_EXT.1)

TSS Assurance Activities

Assurance Activity AA-SVPPM-FMT_MOF_EXT.1-ASE-01

The evaluator shall examine the TSS and Operational Guidance to ensure that it describes which security management functions require Administrator privilege and the actions associated with each

management function. The evaluator shall verify that for each management function and role specified in the FMT_MOF_EXT.1.1 Server Virtualization Management Functions Table (Table 3), the defined role is able to perform all mandatory functions as well as all optional or selection-based functions claimed in the ST.

Summary

The evaluator verified that the “Security Management” section in the TSS of [ST] lists all management functions that can be performed by the sole Administrator role. During testing (as described below), the evaluator verified that the Administrator can perform all functions listed in the aforementioned section.

Guidance Assurance Activities

Assurance Activity AA-SVPPM-FMT_MOF_EXT.1-AGD-01

The evaluator shall examine the Operational Guidance to ensure that it describes how the Administrator or User are able to perform each management function that the ST claims the TOE supports.

The evaluator shall verify for each claimed management function that the Operational Guidance is sufficiently detailed to allow the function to be performed.

Summary

[ST] claims the TOE supports the management functions in the following list. The evaluator examined the operational guidance and determined it describes how the administrator performs each of the claimed management functions and provides sufficient detail to enable the administrator to perform each function. The list identifies, for each claimed function, the specific locations in the guidance documentation that describes how the administrator performs the function:

- Ability to update the Virtualization System: Section “Trusted Updates”
- Ability to configure Administrator password policy as defined in FIA_PMG_EXT.1: Section “Password Management”
- Ability to create, configure and delete VMs: Table “Security-Relevant Management Functions”, third row
- Ability to set default initial VM configurations: Section “Management APIs (Consolidated)”
- Ability to configure virtual networks including VM: Section “Virtual Networking”
- Ability to configure and manage the audit system and audit data: Section “Audit Configuration (FAU)”
- Ability to configure VM access to physical devices: Section “Physical Platform Resources”, “USB”, and “Physical Network”
- Ability to configure inter-VM data sharing: Section “Virtual Networking”
- Ability to configure removable media policy: Section “Removable Devices and Media”
- Ability to configure the cryptographic functionality: Section “Cryptographic Configuration (FCS)”
- Ability to change default authorization factors: Section “Authentication Configuration (FIA)”
- Ability to configure remote connection inactivity timeout: Section “Session Timeouts”

- Ability to configure lockout policy for unsuccessful authentication attempts through limiting number of attempts during a time period: Section “Authentication Failure Handling”
- Ability to configure name/address of audit/logging server to which to send audit/logging records: Section “Configuring Remote Audit Server”
- Ability to configure name/address of network time server: Table “Security-Relevant Management Functions”, 19th row
- Ability to configure banner: Section “Administrative Access Banner”
- Ability to connect/disconnect removable devices to/from a VM: Section “Removable Devices and Media” respectively.
- Ability to start a VM: Table “Security-Relevant Management Functions”, 22nd row
- Ability to stop/halt a VM: Table “Security-Relevant Management Functions”, 23rd row
- Ability to checkpoint a VM: Table “Security-Relevant Management Functions”, 24th row
- Ability to suspend a VM: Table “Security-Relevant Management Functions”, 25th row
- Ability to resume a VM: Section “Management APIs (Consolidated)” notes resuming a VM is performed by starting a VM that is in a suspended state.

Test Assurance Activities

Assurance Activity AA-SVPPM-FMT_MOF_EXT.1-ATE-01

The evaluator shall test each management function for each role listed in the FMT_MOF_EXT.1.1 Server Virtualization Management Functions Table (Table 3) in the ST to demonstrate that the function can be performed by the roles that are authorized to do so and the result of the function is demonstrated. The evaluator shall also verify for each claimed management function that if the TOE claims not to provide a particular role with access to the function, then it is not possible to access the TOE as that role and perform that function.

Summary

The evaluator set up an unprivileged (guest) user account for this assurance activity. Then, the evaluator performed each management function using the administrator (root) account, and verified the functionality completed successfully. Finally, the evaluator attempted to perform the management functions using the guest user account and verified no management functions were accessible.

2.2.5.2 Separation of Management and Operational Networks (FMT_SMO_EXT.1)

TSS Assurance Activities

Assurance Activity AA-BVPP-FMT_SMO_EXT.1-ASE-01

The evaluator shall examine the TSS to verify that it describes how management and operational traffic is separated.

Summary

The evaluator verified that the “Security Management” section in the TSS of [ST] states that management and operational traffic is always separated logically by using separate logical networks. In addition, the Administrator can also choose to physically separate the networks using separate NICs.

Guidance Assurance Activities

Assurance Activity AA-BVPP-FMT_SMO_EXT.1-AGD-01

The evaluator shall examine the operational guidance to verify that it details how to configure the VS to keep Management and Operational traffic separate.

Summary

The evaluator examined the operational guidance and found that the “Isolating VM Networks from the Management Network” section details how the administrator can configure the TOE to place the management network and guest networking on physically isolated networks.

Test Assurance Activities

Assurance Activity AA-BVPP-FMT_SMO_EXT.1-ATE-01

The evaluator shall configure the TOE as documented in the guidance. If separation is logical, then the evaluator shall capture packets on the management network. If plaintext Guest network traffic is detected, the requirement is not met.

If separation uses trusted channels, then the evaluator shall capture packets on the network over which traffic is tunneled. If plaintext Guest network traffic is detected, the requirement is not met.

If data encryption is used, then the evaluator shall capture packets on the network over which the data is sent while a VM or other large data structure is being transmitted. If plaintext VM contents are detected, the requirement is not met.

Summary

The evaluator configured a packet capture on the TOE’s management network. A VM on a guest network was then used to ping another VM on the same network. The second VM was then used to transfer some plaintext test data to the first VM. A connection was also opened to the TOE’s management interface. No traffic from the VMs on the guest network was seen on the management interface, all traffic to the TOE’s management interface was protected by TLS, and no plaintext HTTP traffic was detected.

The ST does not select “data encryption” as a method for separating management and operational networks.

2.2.6 Protection of the TSF (FPT)

2.2.6.1 Non-Existence of Disconnected Virtual Devices (FPT_DVD_EXT.1)

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-BVPP-FPT_DVD_EXT.1-ATE-01

The evaluator shall connect a device to a VM, then from within the guest scan the VM's devices to ensure that the connected device is present--using a device driver or other available means to scan the VM's I/O ports or PCI Bus interfaces. (The device's interface should be documented in the TSS under FPT_VDP_EXT.1.) The evaluator shall remove the device from the VM and run the scan again. This requirement is met if the device's interfaces are no longer present.

Summary

The evaluator set up a Guest VM using Ubuntu 24.04.1 and connected a USB drive to the physical system on which ESXi executes.

The evaluator used the Host Client to successfully connect the USB drive to the VM. The evaluator verified in Ubuntu that the device was visible using lsblk and df. The evaluator then disconnected the USB drive from the VM and verified in Ubuntu that the device was no longer visible using lsblk and df.

2.2.6.2 Execution Environment Mitigations (FPT_EEM_EXT.1)

TSS Assurance Activities

Assurance Activity AA-BVPP-FPT_EEM_EXT.1-ASE-01

> The evaluator shall examine the TSS to ensure that it states, for each platform listed in the ST, the execution environment-based vulnerability mitigation mechanisms used by the TOE on that platform. The evaluator shall ensure that the lists correspond to what is specified in FPT_EEM_EXT.1.1.

Summary

The evaluator verified that the “Protection of the TSF” section in the TSS of [ST] states that address space randomization, memory execution protection, and stack buffer overflow protection are used to prevent unintended machine code execution. The evaluator confirmed this corresponds to what is selected in FPT_EEM_EXT.1. The TOE executes only on the Dell PowerEdge R660 platform.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

No assurance activities defined.

2.2.6.3 Hardware Assists (FPT_HAS_EXT.1)

TSS Assurance Activities

Assurance Activity AA-BVPP-FPT_HAS_EXT.1-ASE-01

> The evaluator shall examine the TSS to ensure that it states, for each platform listed in the ST, the hardware assists and memory-handling extensions used by the TOE on that platform. The evaluator shall ensure that these lists correspond to what is specified in the applicable FPT_HAS_EXT component.

Summary

The evaluator verified that the “Protection of the TSF” section in the TSS of [ST] states that VT-x is used to reduce the use of binary translation and EPT is used to eliminate the need for shadow page tables. The evaluator confirmed this corresponds to what is selected in FPT_HAS_EXT.1. The TOE executes only on the Dell PowerEdge R660 platform.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

No assurance activities defined.

2.2.6.4 Hypercall Controls (FPT_HCL_EXT.1)

TSS Assurance Activities

Assurance Activity AA-BVPP-FPT_HCL_EXT.1-ASE-01

The evaluator shall examine the TSS (or proprietary TSS Annex) to ensure that all hypercall functions are documented at the level necessary for the evaluator to run the below test. Documentation for each hypercall interface must include: how to invoke the interface, parameters and legal values, and any conditions under which the interface can be invoked (e.g., from guest user mode, guest privileged mode, during guest boot only).

Summary

The vendor provided the hypercall documentation in the form of a proprietary TSS annex. The evaluator inspected this document and confirmed that all hypercall functions are documented in sufficient detail, including how to invoke it, parameters and legal values, and conditions under which the interfaces can be invoked.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-BVPP-FPT_HCL_EXT.1-ATE-01

The evaluator shall perform the following test:

For each hypercall interface documented in the TSS or proprietary TSS Annex, the evaluator shall attempt to invoke the function from within the VM using an invalid parameter (if any). If the VMM or VS crashes or generates an exception, or if no error is returned to the guest, then the test fails. If an error is returned to the guest, then the test succeeds.

Summary

The evaluator set up a Guest VM using Ubuntu 24.04.1, then used a proprietary test application and script to invoke each hypercall that accepts input parameters. For each hypercall, an invalid input parameter is provided, which resulted in an error or failure indicator.

2.2.6.5 Removable Devices and Media (FPT_RDM_EXT.1)

TSS Assurance Activities

Assurance Activity AA-BVPP-FPT_RDM_EXT.1-ASE-01

The evaluator shall examine the TSS to ensure it describes the association between the media or devices supported by the TOE and the actions that can occur when switching information domains.

Summary

The evaluator verified that the “Protection of the TSF” section in the TSS of [ST] states that access to removable (USB, CD/DVD) media and devices is configured explicitly by the Administrator for each VM. This policy applies to both virtual and physical media. ISO images are mounted read-only.

The evaluator also verified that the TSS describes that access to the removable media is prevented in case of switching information domains.

Guidance Assurance Activities

Assurance Activity AA-BVPP-FPT_RDM_EXT.1-AGD-01

The evaluator shall examine the operational guidance to ensure it documents how an administrator or user configures the behavior of each media or device.

Summary

The evaluator verified that “Removable Devices and Media” section of the operational guidance documents how an administrator or user configures the behavior of each media or device.

Test Assurance Activities

Assurance Activity AA-BVPP-FPT_RDM_EXT.1-ATE-01

The evaluator shall perform the following test for each listed media or device:

- Test 1: The evaluator shall configure two VMs that are members of different information domains, with the media or device connected to one of the VMs. The evaluator shall disconnect the media or device from the VM and connect it to the other VM. The evaluator shall verify that the action performed is consistent with the action assigned in the TSS.

Summary

The evaluator set up two identical Guest VMs using Ubuntu 24.04.1 and connected a USB drive to the physical system on which ESXi executes.

For the physical USB media type, the evaluator used the Host Client to successfully connect the USB drive to the first VM. The evaluator then disconnected the USB drive from the first VM and successfully connected it to the second VM.

For the virtual CD/DVD image media type, the evaluator used the Host Client to successfully connect a datastore ISO image to the first VM. The evaluator then disconnected the ISO image from the first VM and successfully connected it to the second VM.

In both cases, the evaluator confirmed the action performed is consistent with the action assigned in the TSS.

2.2.6.6 Trusted Updates to the Virtualization System (FPT_TUD_EXT.1)

TSS Assurance Activities

Assurance Activity AA-BVPP-FPT_TUD_EXT.1-ASE-01

The evaluator shall verify that the TSS describes all TSF software update mechanisms for updating the system software. Updates to the TOE either have a hash associated with them, or are signed by an authorized source. The evaluator shall verify that the description includes either a digital signature or published hash verification of the software before installation and that installation fails if the verification fails. The evaluator shall verify that the TSS describes the method by which the digital signature or published hash is verified to include how the candidate updates are obtained, the processing associated with verifying the update, and the actions that take place for both successful and unsuccessful verification. If digital signatures are used, the evaluator shall also ensure the definition of an authorized source is contained in the TSS.

If the ST author indicates that a certificate-based mechanism is used for software update digital signature verification, the evaluator shall verify that the TSS contains a description of how the certificates are contained on the device. The evaluator also ensures that the TSS (or administrator guidance) describes how the certificates are installed/updated/selected, if necessary.

Summary

The evaluator verified that the “Protection of the TSF” section in the TSS of [ST] describes the sole mechanism to update the system software. Each update is signed by VMware, the only authorized source of updates. If the digital signature validation check fails, the installation fails. Otherwise the installation proceeds, applying the update to the TOE. Section 6.7 in the TSS of [ST] also lists the location of the public keys on the TOE. These public keys are part of the TOE software and not installed/updated/selected unless as part of a validated software update.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-BVPP-FPT_TUD_EXT.1-ATE-01

The evaluator shall perform the following tests:

- Test 1: The evaluator performs the version verification activity to determine the current version of the product. The evaluator obtains a legitimate update using procedures described in the operational guidance and verifies that it is successfully installed on the TOE. After the update, the evaluator performs the version verification activity again to verify the version correctly corresponds to that of the update.
- Test 2: The evaluator performs the version verification activity to determine the current version of the product. The evaluator obtains or produces illegitimate updates as defined below, and attempts to install them on the TOE. The evaluator verifies that the TOE rejects all of the illegitimate updates. The evaluator performs this test using all of the following forms of illegitimate updates:

1) A modified version (e.g., using a hex editor) of a legitimately signed or hashed update
2) An image that has not been signed/hashed
3) An image signed with an invalid hash or invalid signature (e.g., by using a different key as expected for creating the signature or by manual modification of a legitimate hash/signature)

Summary

The vendor provided the evaluator with an ESXi update ZIP file of a depot. For convenience, the evaluator performed Test 2 before Test 1:

Test 2: the evaluator modified the update by changing the contents of a package in the depot, by removing the signature for a package in the depot, and by modifying the public keys for a package in the depot. All modifications were performed individually, and the evaluator verified the installation failed in all cases.

Test 1: the evaluator confirmed the version of ESXi, then installed the unmodified update. The evaluator verified the version changed successfully.

2.2.6.7 Virtual Device Parameters (FPT_VDP_EXT.1)

TSS Assurance Activities

Assurance Activity AA-BVPP-FPT_VDP_EXT.1-ASE-01

The evaluator shall examine the TSS to ensure it lists all virtual devices accessible by the guest OS. The TSS, or a separate proprietary document, must also document all virtual device interfaces at the level of I/O ports or PCI Bus interfaces - including port numbers (absolute or relative to a base), port name, address range, and a description of legal input values.

The TSS must also describe the expected behavior of the interface when presented with illegal input values. This behavior must be deterministic and indicative of parameter checking by the TSF.

The evaluator must ensure that there are no obvious or publicly known virtual I/O ports missing from the TSS.

There is no expectation that evaluators will examine source code to verify the “all” part of the evaluation activity.

Summary

The evaluator verified that the “Protection of the TSF” sectopm in the TSS of [ST] lists all virtual devices accessible to Guest VMs. The vendor provided further virtual device documentation in the form of a proprietary TSS annex. The evaluator inspected this

document and confirmed that all virtual device interfaces are documented in sufficient detail. All virtual devices, with the exception of floppy drives, serial ports, and parallel ports use the PCI bus interface. A Super I/O virtual device is used to implement the remaining ports. For each of the virtual interfaces/devices, the proprietary annex specifies the address descriptions and sizes, input / output types, references to the specification, and any additionally notes if necessary. There are no obvious or publicly known virtual I/O ports missing from the proprietary annex.

“Protection of the TSF” also states that illegal input values to the virtual devices are rejected.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-BVPP-FPT_VDP_EXT.1-ATE-01

For each virtual device interface, the evaluator shall attempt to access the interface using at least one parameter value that is out of range or illegal. The test is passed if the interface behaves in the manner documented in the TSS. Interfaces that do not have input parameters need not be tested. This test can be performed in conjunction with the tests for FPT_DVD_EXT.1.

Summary

The evaluator used a proprietary Guest VM including test applications to configure PCI devices. For each of the PCI devices accessible to the VM, the evaluator attempted to modify the parameters to an invalid value. The values remain unchanged, and no catastrophic error occurred.

2.2.6.8 VMM Isolation from VMs (FPT_VIV_EXT.1)

TSS Assurance Activities

Assurance Activity AA-BVPP-FPT_VIV_EXT.1-ASE-01

The evaluator shall verify that the TSS (or a proprietary annex to the TSS) describes how the TSF ensures that guest software cannot degrade or disrupt the functioning of other VMs, the VMM or the platform. And how the TSF prevents guests from invoking higher-privilege platform code, such as the examples in the note.

Summary

The evaluator verified that the “Protection of the TSF” section in the TSS of [ST] describes why VMs are not able to degrade or disrupt the functioning of other VMs, the VMM, or the platform. The following security mechanisms are provided:

- Intel VT-x and VT-d hardware virtualization to ensure VM/VM and VM/TOE isolation.
- Virtualization of System Management Mode and System Management Interrupts. There is no correlation between the virtualized and physical SMIs.
- No platform firmware, I/O ports, or MMIO registers are mapped to Guest VMs.
- Attempting to update microcode from a Guest VM will be logged and dropped.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

No assurance activities defined.

2.2.7 TOE access (FTA)

2.2.7.1 TOE Access Banner (FTA_TAB.1)

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-BVPP-FTA_TAB.1-ATE-01

The evaluator shall configure the TOE to display the advisory warning message “TEST TEST Warning Message TEST TEST”. The evaluator shall then log out and confirm that the advisory message is displayed before login can occur.

Summary

ESXi can be configured to display two types of advisory messages: one prior to login and one after login. The evaluator configured the message to “TEST TEST Warning Message TEST TEST”, then confirmed the message was displayed prior and after logging in to the Host Client.

2.2.8 Trusted path/channels (FTP)

2.2.8.1 Trusted Channel Communications (FTP_ITC_EXT.1)

TSS Assurance Activities

Assurance Activity AA-BVPP-FTP_ITC_EXT.1-ASE-01

The evaluator will review the TSS to determine that it lists all trusted channels the TOE uses for remote communications, including both the external entities and remote users used for the channel as well as the protocol that is used for each.

Summary

The evaluator verified that the “Trusted Path/Channels” section in the TSS of [ST] describes TLS and HTTPS as the trusted channel used for remote communications. TLS/HTTPS is used for remote administration using the VIM API or ESXCLI. TLS is used to protect communications with the remote audit server.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-BVPP-FTP_ITC_EXT.1-ATE-01

The evaluator will configure the TOE to communicate with each external IT entity and type of remote user identified in the TSS. The evaluator will monitor network traffic while the VS performs communication with each of these destinations. The evaluator will ensure that for each session a trusted channel was established in conformance with the protocols identified in the selection.

Summary

Three purposes are identified in the selection: audit servers, remote administrators, and separation of management and operational networks. These communication channels have been tested as part of FAU_STG_EXT.1, FTP_TRP.1, and FMT_SMO_EXT.1, respectively. In all cases, a TLS channel was established in conformance with TLS 1.2.

2.2.8.2 Trusted Path (FTP_TRP.1)

TSS Assurance Activities

Assurance Activity AA-BVPP-FTP_TRP.1-ASE-01

The evaluator shall examine the TSS to determine that the methods of remote TOE administration are indicated, along with how those communications are protected. The evaluator shall also confirm that all protocols listed in the TSS in support of TOE administration are consistent with those specified in the requirement, and are included in the requirements in the ST.

Summary

The evaluator verified that the “Security Management” and “Trusted Path/Channels” sections in the TSS of [ST] list the methods of remote TOE administration: VIM API and ESXCLI. These methods are protected using TLS/HTTPS, which is consistent with the trusted path listed in FTP_ITC_EXT.1. The [ST] includes the requirements for HTTPS and TLS, and claims conformance to the Functional Package for Transport Layer Security.

Guidance Assurance Activities

Assurance Activity AA-BVPP-FTP_TRP.1-AGD-01

The evaluator shall confirm that the operational guidance contains instructions for establishing the remote administrative sessions for each supported method.

Summary

The evaluator reviewed the operational guidance and confirmed that the “Establishing Remote Administrative Sessions” section contains instructions for securely establishing remote administrative sessions: using the VIM API or ESXCLI.

Test Assurance Activities

Assurance Activity AA-BVPP-FTP_TRP.1-ATE-01

The evaluator shall also perform the following tests:

- Test 1: The evaluators shall ensure that communications using each specified (in the operational guidance) remote administration method is tested during the course of the evaluation, setting up the connections as described in the operational guidance and ensuring that communication is successful.
- Test 2: For each method of remote administration supported, the evaluator shall follow the operational guidance to ensure that there is no available interface that can be used by a remote user to establish remote administrative sessions without invoking the trusted path.
- Test 3: The evaluator shall ensure, for each method of remote administration, the channel data is not sent in plaintext.
- Test 4: The evaluator shall ensure, for each method of remote administration, modification of the channel data is detected by the TOE.

Additional evaluation activities are associated with the specific protocols.

Summary

The tests were performed using the VIM API and ESXCLI, as identified in the operational guidance. All packets were logged using tshark (part of the Wireshark tools).

Test 1: the evaluator confirmed that a connection to the ESXi host using the HTTPS protocol on port 443 was successful, and remote administration was possible.

Test 2: the evaluator found that a connection to the ESXi host using the HTTP protocol on port 443 was unsuccessful.

Test 3 and 4: the evaluator inspected the packet logs and verified only TLS data was sent. As TLS protects the confidentiality and integrity of the data, it is impossible to inspect the plaintext channel data or modify the channel data.

2.2.8.3 User Interface: I/O Focus (FTP_UIF_EXT.1)

TSS Assurance Activities

Assurance Activity AA-BVPP-FTP_UIF_EXT.1-ASE-01

The evaluator shall ensure that the TSS lists the supported user input devices.

Summary

The evaluator verified that the “Trusted Path/Channels” section in the TSS of [ST] lists the remote administration interfaces. The ESXCLI interface is not used to manipulate VMs, so is irrelevant for user input. The VIM API uniquely identifies VMs using a system-assigned identifier (the Managed Object Identifier), which can be used to programmatically interact with VMs.

Guidance Assurance Activities

Assurance Activity AA-BVPP-FTP_UIF_EXT.1-AGD-01

The evaluator shall ensure that the operational guidance specifies how the current input focus is indicated to the user.

Summary

The evaluator confirmed that the “User Interface Indicators” section of the operational guidance states that the VIM API identifies virtual machines with a Managed Object Identifier (MOID), which is the unique identifier used within the API.

Test Assurance Activities

Assurance Activity AA-BVPP-FTP_UIF_EXT.1-ATE-01

For each supported input device, the evaluator shall demonstrate that the input from each device listed in the TSS is directed to the VM that is indicated to have the input focus.

Summary

The evaluator created three Guest VMs using the VIM API and confirmed each VM was assigned a different MOID. The evaluator used the MOID to interface with the VMs (power on / power off / destroy).

2.2.8.4 User Interface: Identification of VM (FTP_UIF_EXT.2)

TSS Assurance Activities

Assurance Activity AA-BVPP-FTP_UIF_EXT.2-ASE-01

The evaluator shall ensure that the TSS describes the mechanism for identifying VMs to the user, how identities are assigned to VMs, and how conflicts are prevented.

Summary

The evaluator verified that the “Trusted Path/Channels” section in the TSS of [ST] lists the remote administration interfaces. The ESXCLI interface is not used to manipulate VMs, so is irrelevant for user input. The VIM API uniquely identifies VMs using a system-assigned identifier (the Managed Object Identifier), which can be used to programmatically interact with VMs. As this identifier is uniquely assigned, conflicts are prevented.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-BVPP-FTP_UIF_EXT.2-ATE-01

The evaluator shall perform the following test:

The evaluator shall attempt to create and start at least three Guest VMs on a single display device where the evaluator attempts to assign two of the VMs the same identifier. If the user interface displays different identifiers for each VM, then the requirement is met. Likewise, the requirement is met if the system refuses to create or start a VM when there is already a VM with the same identifier.

Summary

The evaluator created three Guest VMs using the VIM API, two of which had the same display name. Each VM was assigned a different MOID. The evaluator used the MOID to successfully interface with the VMs (power on / power off / destroy).

2.3 Security Assurance Requirements

2.3.1 Development (ADV)

2.3.1.1 Functional specification (ADV_FSP.1)

There are no specific evaluation activities associated with these SARs. The functional specification documentation is provided to support the evaluation activities described in Section 5.2, and other activities described for AGD, ATE, and AVA SARs. The requirements on the content of the functional specification information is implicitly assessed by virtue of the other evaluation activities being performed; if the evaluator is unable to perform an activity because there is insufficient interface information, then an adequate functional specification has not been provided.

2.3.2 Guidance documents (AGD)

2.3.2.1 Operational user guidance (AGD_OPE.1)

Assurance Activity AA-BVPP-AGD_OPE.1-AGD-01

Some of the contents of the operational guidance will be verified by the evaluation activities in Section 5.2 and evaluation of the TOE according to the CEM. The following additional information is also required.

The operational guidance shall contain instructions for configuring the password characteristics, number of allowed authentication attempt failures, the lockout period times for inactivity, and the notice and consent warning that is to be provided when authenticating.

The operational guidance shall contain step-by-step instructions suitable for use by an end-user of the VS to configure a new, out-of-the-box system into the configuration evaluated under this Protection Profile.

The documentation shall describe the process for verifying updates to the TOE, either by checking the hash or by verifying a digital signature. The evaluator shall verify that this process includes the following steps:

- Instructions for querying the current version of the TOE software.
- For hashes, a description of where the hash for a given update can be obtained. For digital signatures, instructions for obtaining the certificate that will be used by the FCS_COP.1/SIG mechanism to ensure that a signed update has been received from the certificate owner. This may be supplied with the product initially, or may be obtained by some other means.
- Instructions for obtaining the update itself. This should include instructions for making the update accessible to the TOE (e.g., placement in a specific directory).
- Instructions for initiating the update process, as well as discerning whether the process was successful or unsuccessful. This includes generation of the hash/digital signature.

Summary

The evaluator reviewed the operational guidance and found that the required instructions are present:

- To configure the password characteristics: the “Password Management” section.
- To configure the number of allowed authentication attempt failures and the lockout period times for inactivity: the “Authentication Failure Handling” section.
- To configure the notice and consent warning that is to be provided when authenticating: the “Administrative Access Banner” section.
- Step-by-step instructions to enter the evaluated configuration: “Installation Guidance and Preparative Procedures” section.

- Process for verifying updates to ESXi, including querying the current version (“Verify Software” section), obtaining the update itself (“Obtain Software” section), and initiating the update process (“Install or Update Software” and “Trusted Updates” sections). There are no instructions required to obtain the certificates, as they are supplied with the TOE (“Trusted Updates” section).

2.3.2.2 Preparative procedures (AGD_PRE.1)

Assurance Activity AA-BVPP-AGD_PRE.1-AGD-01

As indicated in the introduction above, there are significant expectations with respect to the documentation—especially when configuring the operational environment to support TOE functional requirements. The evaluator shall check to ensure that the guidance provided for the TOE adequately addresses all platforms (that is, combination of hardware and operating system) claimed for the TOE in the ST.

The operational guidance shall contain step-by-step instructions suitable for use by an end-user of the VS to configure a new, out-of-the-box system into the configuration evaluated under this Protection Profile.

Summary

The evaluator verified that the “Evaluated Configuration And TOE Overview” section in the operational guidance explains the platforms claimed for ESXi, and these platforms match those listed in [ST]. The instructions in the “Installing the TOE” are adequate to address the platforms, to install ESXi according to the evaluated configuration.

2.3.3 Life-cycle support (ALC)

2.3.3.1 Labelling of the TOE (ALC_CMC.1)

Assurance Activity AA-BVPP-ALC_CMC.1-ALC-01

The evaluator shall check the ST to ensure that it contains an identifier (such as a product name/version number) that specifically identifies the version that meets the requirements of the ST.

The evaluator shall check the AGD guidance and TOE samples received for testing to ensure that the version number is consistent with that in the ST.

If the vendor maintains a website advertising the TOE, the evaluator shall examine the information on the website to ensure that the information in the ST is sufficient to distinguish the product.

Summary

The evaluator verified that Section 1.1 of [ST] identifies the version that meets the requirements of the ST. The evaluator cross-checked this version with the [CCGUIDE] and the provided software package, and verified it was consistent. Finally, the evaluator found https://docs.vmware.com/en/VMware-vSphere/8.0/rn/vsphere-esxi-803-release-notes/index.html, which confirms that the [ST] is sufficient to distinguish the product.

2.3.3.2 TOE CM coverage (ALC_CMS.1)

Assurance Activity AA-BVPP-ALC_CMS.1-ALC-01

The evaluator shall ensure that the developer has identified (in public-facing development guidance for their platform) one or more development environments appropriate for use in developing applications for the developer’s platform. For each of these development environments, the developer shall provide information on how to configure the environment to ensure that buffer overflow protection mechanisms in the environment are invoked (e.g., compiler and linker flags). The evaluator shall ensure that this documentation also includes an indication of whether such protections are on by default, or have to be specifically enabled. The evaluator shall ensure that the TSF is uniquely identified (with respect to other products from the TSF vendor), and that documentation provided by the developer in association with the requirements in the ST is

associated with the TSF using this unique identification.

Summary

There is no documentation to develop application specifically for the TOE, as the TOE is simply a hypervisor.

The evaluator verified that Section 1.1 of [ST] uniquely identifies the version that meets the requirements of the ST (i.e., the TSF). The evaluator cross-checked this version with the [CCGUIDE] and verified it was consistent.

2.3.3.3 Timely security updates (ALC_TSU_EXT.1)

Assurance Activity AA-BVPP-ALC_TSU_EXT.1-ALC-01

This component requires the TOE developer, in conjunction with any other necessary parties, to provide information as to how the VS is updated to address security issues in a timely manner. The documentation describes the process of providing updates to the public from the time a security flaw is reported/discovered, to the time an update is released. This description includes the parties involved (e.g., the developer, hardware vendors) and the steps that are performed (e.g., developer testing), including worst case time periods, before an update is made available to the public.

The description shall include the process for creating and deploying security updates for the TOE software/firmware.

The description shall express the time window as the length of time, in days, between public disclosure of a vulnerability and the public availability of security updates to the TOE.

The description shall include the mechanisms publicly available for reporting security issues pertaining to the TOE.

The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

Summary

The evaluator verified that the “Timely Security Updates” section in the TSS of [ST] describes how Broadcom timely addresses any security issues in the TOE. This includes a description of the TOE packages, a link to the Broadcom Security Response Policy, the parties involved and test performed, and time periods for updates. This section also includes a description how to install VIBs (also in the operational guidance). Finally, the evaluator confirmed that this Section provides an e-mail address to report security issues pertaining to the TOE.

2.3.4 Tests (ATE)

2.3.4.1 Independent testing - conformance (ATE_IND.1)

Assurance Activity AA-BVPP-ATE_IND.1-ATE-01

The evaluator shall prepare a test plan and report documenting the testing aspects of the system. While it is not necessary to have one test case per test listed in an evaluation activity, the evaluators must document in the test plan that each applicable testing requirement in the ST is covered.

The Test Plan identifies the platforms to be tested, and for those platforms not included in the test plan but included in the ST, the test plan provides a justification for not testing the platforms. This justification must address the differences between the tested platforms and the untested platforms, and make an argument that the differences do not affect the testing to be performed. It is not sufficient to merely assert that the differences have no affect; rationale must be provided. If all platforms claimed in the ST are tested, then no rationale is necessary.

The test plan describes the composition of each platform to be tested, and any setup that is necessary beyond what is contained in the AGD documentation. It should be noted that the

evaluators are expected to follow the AGD documentation for installation and setup of each platform either as part of a test or as a standard pre-test condition. This may include special test drivers or tools. For each driver or tool, an argument (not just an assertion) is provided that the driver or tool will not adversely affect the performance of the functionality by the TOE and its platform. This also includes the configuration of cryptographic engines to be used. The cryptographic algorithms implemented by these engines are those specified by this PP and used by the cryptographic protocols being evaluated (IPsec, TLS/HTTPS, SSH).

The test plan identifies high-level test objectives as well as the test procedures to be followed to achieve those objectives. These procedures include expected results. The test report (which could just be an annotated version of the test plan) details the activities that took place when the test procedures were executed, and includes the actual results of the tests. This shall be a cumulative account, so if there was a test run that resulted in a failure; a fix installed; and then a successful re-run of the test, the report would show a “fail” and “pass” result (and the supporting details), and not just the “pass” result.

Summary

The evaluators prepared a Detailed Test Report (DTR) to specify all the tests covering the assurance activities in this evaluation. The DTR, which contains test requirements i.e., test assurance activities), test procedures, expected test results, actual test results, along with the verdict, is provided to the validation body but is not published.

The test environment was set up according to a setup strategy that followed the evaluated configuration requirements specified in the guidance [CCGUIDE] and Security Target [ST], supplemented by configurations required to perform testing. The testing was performed at the atsec CCTL located in Austin, TX, US, in December 2024, April 2025, and May 2025. All platforms claimed in [ST] were tested.

The following tools were used for testing:

- Dell PowerEdge R660 (running the TOE, VMware ESXi 8.0 Update 3e), with the ESXi Host Client and GDB installed
- Lenovo ThinkPad T440p, with:
  - OpenSSL 3.2.0 or newer
  - Wireshark version 4.2.9 (including tshark)
  - Python 3.12.8
  - ESXCLI 8.0.0-22179150
  - Rsyslog 8.2312.0
- Proprietary test scripts developed by the CCTL

The independent testing of the TOE was originally performed on VMware ESXi 8.0 Update 3 (build 24022510). During the course of the evaluation, the vendor released a new version, VMware ESXi 8.0 Update 3e (build 24674464), which became the new TOE. The evaluators consulted with the vendor to analyze the differences between these two builds and created a list of SFRs that were potentially impacted. The testing evaluation activities for these SFRs were re-performed on the new build:

- FAU_STG_EXT.1.1
- FAU_STG_EXT.1.2
- FCS_CKM_EXT.4
- FDP_PPR_EXT.1
- FDP_VNC_EXT.1
- FPT_DVD_EXT.1

- FPT_HCL_EXT.1
- FPT_TUD_EXT.1 (Test 1)
- FPT_VDP_EXT.1
- FIA_X509_EXT.1
- FIA_X509_EXT.2
- FMT_MOF_EXT.1
- FCS_TLSC_EXT.1
- FCS_TLSC_EXT.5
- FCS_TLSS_EXT.1

2.3.5 Vulnerability assessment (AVA)

2.3.5.1 Vulnerability survey (AVA_VAN.1)

Assurance Activity AA-BVPP-AVA_VAN.1-AVA-01

As with ATE_IND the evaluator shall generate a report to document their findings with respect to this requirement. This report could physically be part of the overall test report mentioned in ATE_IND, or a separate document. The evaluator performs a search of public information to determine the vulnerabilities that have been found in virtualization in general, as well as those that pertain to the particular TOE. The evaluator documents the sources consulted and the vulnerabilities found in the report. For each vulnerability found, the evaluator either provides a rationale with respect to its non-applicability or the evaluator formulates a test (using the guidelines provided in ATE_IND) to confirm the vulnerability, if suitable. Suitability is determined by assessing the attack vector needed to take advantage of the vulnerability. For example, if the vulnerability can be detected by pressing a key combination on boot-up, a test would be suitable at the assurance level of this PP. If exploiting the vulnerability requires expert skills and an electron microscope, for instance, then a test would not be suitable and an appropriate justification would be formulated.

Summary

The evaluator prepared an Appendix to the Evaluation Technical Report (ETR) to specify the vulnerability search and analysis performed as part of this assurance activity. The evaluator searched for publicly known vulnerabilities applicable to the TOE from sources published by the vendor, MITRE, NIST, and CISA. In addition to the lists of fixes published by the vendor, the evaluator performed manual searches as the evaluation of the TOE progressed.

The search terms included the TOE name and version, evaluated configuration platform (including processor), software (including cryptographic libraries) and hardware that compose the TOE, and other technical terms related to the security of the TOE as required.

All vulnerabilities affecting the TOE were documented and their applicability to the TOE security functionality determined. The evaluator found no vulnerabilities that impact the TSF and are not mitigated.

Finally, the evaluator performed a generic port scan on the TOE to search for any undocumented open network ports. The evaluator found that no ports are unexpectedly open. In other words: all ports are associated with the TOE and publicly documented as such.

A. Appendixes

A.1. References

CC	**Common Criteria for Information Technology Security Evaluation** Version CC:2022 R1 Date November 2022 Location https://www.commoncriteriaportal.org/files/ccfiles/CC2022PART1R1.pdf Location https://www.commoncriteriaportal.org/files/ccfiles/CC2022PART2R1.pdf Location https://www.commoncriteriaportal.org/files/ccfiles/CC2022PART3R1.pdf Location https://www.commoncriteriaportal.org/files/ccfiles/CC2022PART4R1.pdf Location https://www.commoncriteriaportal.org/files/ccfiles/CC2022PART5R1.pdf
CCEVS-LG	**CCEVS LabGrams** Author(s) NIAP Date May 2025 Location https://www.niap-ccevs.org/labgrams
CCEVS-PL	**CCEVS Scheme Policy Letters** Author(s) NIAP Date May 2025 Location https://www.niap-ccevs.org/policies
CCEVS-PUB	**CCEVS Scheme Publications** Author(s) NIAP Date May 2025 Location https://www.niap-ccevs.org/publications
CCEVS-TD	**Technical Decisions** Author(s) NIAP Date May 2025 Location https://www.niap-ccevs.org/technical-decisions/
CCDB-2017-05-17	**CC and CEM addenda - Exact Conformance, Selection-Based SFRs, Optional SFRs** Version 0.5 Date 2017-05-17 Location https://www.commoncriteriaportal.org/files/ccfiles/CCDB-2017-05-17-CCaddenda-Exact_Conformance.pdf
CCGUIDE	**VMware ESXi 8.0 Update 3e NIAP Common Criteria Guidance Supplement** Date 2025-05-14 File name st_vid11533-agd.pdf
CEM	**Common Methodology for Information Technology Security Evaluation** Version CC:2022 R1 Date November 2022 Location https://www.commoncriteriaportal.org/files/ccfiles/CEM2022R1.pdf
MOD_SV_V1.1	**PP-Module for Server Virtualization Systems** Author(s) NIAP Version 1.1 Date 2021-06-14
PKG_TLS_V1.1	**Functional Package for Transport Layer Security (TLS)** Author(s) NIAP Version 1.1 Date 2019-03-01
PP_BASE_VIRTUALIZATION_V1.1	**Protection Profile for Virtualization** Author(s) NIAP Version 1.1 Date 2021-06-14
ST	**VMware ESXi 8.0 Update 3e Security Target** Date 2025-05-19 File name st_vid11533-st.pdf

A.2. Glossary

Augmentation

The addition of one or more requirement(s) to a package.

Authentication data

Information used to verify the claimed identity of a user.

Authorised user

A user who may, in accordance with the SFRs, perform an operation.

Class

A grouping of CC families that share a common focus.

Component

The smallest selectable set of elements on which requirements may be based.

Connectivity

The property of the TOE which allows interaction with IT entities external to the TOE. This includes exchange of data by wire or by wireless means, over any distance in any environment or configuration.

Dependency

A relationship between components such that if a requirement based on the depending component is included in a PP, ST or package, a requirement based on the component that is depended upon must normally also be included in the PP, ST or package.

Deterministic RNG (DRNG)

An RNG that produces random numbers by applying a deterministic algorithm to a randomly selected seed and, possibly, on additional external inputs.

Element

An indivisible statement of security need.

Entropy

The entropy of a random variable X is a mathematical measure of the amount of information gained by an observation of X.

Evaluation

Assessment of a PP, an ST or a TOE, against defined criteria.

Evaluation Assurance Level (EAL)

An assurance package, consisting of assurance requirements drawn from CC Part 3, representing a point on the CC predefined assurance scale.

Evaluation authority

A body that implements the CC for a specific community by means of an evaluation scheme and thereby sets the standards and monitors the quality of evaluations conducted by bodies within that community.

Evaluation scheme

The administrative and regulatory framework under which the CC is applied by an evaluation authority within a specific community.

Exact conformance

a subset of Strict Conformance as defined by the CC, is defined as the ST containing all of the requirements in the Security Requirements section of the PP, and potentially requirements from Appendices of the PP. While iteration is allowed, no additional requirements (from the CC parts 2 or 3) are allowed to be included in the ST. Further, no requirements in the Security Requirements section of the PP are allowed to be omitted.

Extension

The addition to an ST or PP of functional requirements not contained in Part 2 and/or assurance requirements not contained in Part 3 of the CC.

External entity

Any entity (human or IT) outside the TOE that interacts (or may interact) with the TOE.

Family

A grouping of components that share a similar goal but may differ in emphasis or rigour.

Formal

Expressed in a restricted syntax language with defined semantics based on well-established mathematical concepts.

Guidance documentation

Documentation that describes the delivery, preparation, operation, management and/or use of the TOE.

Identity

A representation (e.g. a string) uniquely identifying an authorised user, which can either be the full or abbreviated name of that user or a pseudonym.

Informal

Expressed in natural language.

Object

A passive entity in the TOE, that contains or receives information, and upon which subjects perform operations.

Operation (on a component of the CC)

Modifying or repeating that component. Allowed operations on components are assignment, iteration, refinement and selection.

Operation (on an object)

A specific type of action performed by a subject on an object.

Operational environment

The environment in which the TOE is operated.

Organisational Security Policy (OSP)

A set of security rules, procedures, or guidelines imposed (or presumed to be imposed) now and/or in the future by an actual or hypothetical organisation in the operational environment.

Package

A named set of either functional or assurance requirements (e.g. EAL 3).

PP evaluation

Assessment of a PP against defined criteria.

Protection Profile (PP)

An implementation-independent statement of security needs for a TOE type.

Random number generator (RNG)

A group of components or an algorithm that outputs sequences of discrete values (usually represented as bit strings).

Refinement

The addition of details to a component.

Role

A predefined set of rules establishing the allowed interactions between a user and the TOE.

Secret

Information that must be known only to authorised users and/or the TSF in order to enforce a specific SFP.

Secure state

A state in which the TSF data are consistent and the TSF continues correct enforcement of the SFRs.

Security attribute

A property of subjects, users (including external IT products), objects, information, sessions and/or resources that is used in defining the SFRs and whose values are used in enforcing the SFRs.

Security Function Policy (SFP)

A set of rules describing specific security behaviour enforced by the TSF and expressible as a set of SFRs.

Security objective

A statement of intent to counter identified threats and/or satisfy identified organisation security policies and/or assumptions.

Security Target (ST)

An implementation-dependent statement of security needs for a specific identified TOE.

Seed

Value used to initialize the internal state of an RNG.

Selection

The specification of one or more items from a list in a component.

Semiformal

Expressed in a restricted syntax language with defined semantics.

ST evaluation

Assessment of an ST against defined criteria.

Subject

An active entity in the TOE that performs operations on objects.

Target of Evaluation (TOE)

A set of software, firmware and/or hardware possibly accompanied by guidance.

TOE evaluation

Assessment of a TOE against defined criteria.

TOE resource

Anything useable or consumable in the TOE.

TOE Security Functionality (TSF)

A set consisting of all hardware, software, and firmware of the TOE that must be relied upon for the correct enforcement of the SFRs.

Transfers outside of the TOE

TSF mediated communication of data to entities not under control of the TSF.

True RNG (TRNG)

A device or mechanism for which the output values depend on some unpredictable source (noise source, entropy source) that produces entropy.

Trusted channel

A means by which a TSF and a remote trusted IT product can communicate with necessary confidence.

Trusted path

A means by which a user and a TSF can communicate with necessary confidence.

TSF data

Data created by and for the TOE, that might affect the operation of the TOE.

TSF Interface (TSFI)

A means by which external entities (or subjects in the TOE but outside of the TSF) supply data to the TSF, receive data from the TSF and invoke services from the TSF.

User

See external entity

User data

Data created by and for the user, that does not affect the operation of the TSF.