SUSE Linux Enterprise Server 15 SP4 Assurance Activity Report

Version:	5
Date:	2023-12-05
Status:	RELEASE
Classification:	Public
Filename:	BSI-DSZ-CC-1213_AAR_231205_v5
Product:	SUSE Linux Enterprise Server 15 SP4
Sponsor:	SUSE, LLC
Evaluation Facility:	atsec information security GmbH
Certification ID:	BSI-DSZ-CC-1213
Certification Body:	BSI
Author(s):	Sebastian Mayer
Quality Assurance:	Clemens Wittinger

atsec information security GmbH
Ismaninger Straße 19
81675 München

Phone: +49 (0)89 / 442 498 30
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	Application Notes
1.0	2023-09-29	Sebastian Mayer	First version	
2.0	2023-10-26	Sebastian Mayer	Update with current references	
3.0	2023-11-03	Sebastian Mayer	Update with current references	
4.0	2023-11-21	Sebastian Mayer	Update with current references	
5.0	2023-12-05	Sebastian Mayer	Update with current references	

Table of Contents

List of Tables

1 Evaluation Basis and Documents

This evaluation is based on the "Common Criteria for Information Technology Security Evaluation" version 3.1 revision 5 [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 General Purpose Operating Systems Version 4.2.1" [OSPPv4.2.1];
- "Functional Package for Secure Shell (SSH)" [SSHPKGv1.0]; and
- "Funktionalitätsklassen und Evaluationsmethodologie für deterministische Zufallszahlengeneratoren (Functionality Classes and Evaluation Methodology for Deterministic RNGs)" [AIS20]

, as specified in the Security Target [ST].

The following scheme documents and interpretations have been considered:

- [AIS14]: "Anforderungen an Aufbau und Inhalt von Einzelprüfberichten für Evaluationen nach CC", version 7 as of 2010-08-03.
- [AIS19]: "Anforderungen an Aufbau und Inhalt der Zusammenfassung des ETR (Evaluation Technical Report) für Evaluationen nach CC (Common Criteria)", version 9 as of 2014-11-03.
- [AIS23]: "Zusammentragen von Nachweisen der Entwickler (Collection of Developer Evidence)", version 4 as of 2017-03-15.
- [AIS32]: "CC-Interpretationen im deutschen Zertifizierungsschema", version 7 as of 2011-06-08.
- [BSIAAA]: "Anforderungen an Antragsteller zur Anerkennung als Prüfstelle im Bereich Common Criteria, CC-Prüfstellen", version 1.5 as of 2023-06-01.
- [CCDB-2017-05-17]: "CC and CEM addenda - Exact Conformance, Selection-Based SFRs, Optional SFRs", version 0.5 as of 2017-05-17.
- [CC-EP]: "Programm [CC-Prüfstellen]: CC-Evaluierungsprozess", version 1.3 as of 2023-07-01.
- [CCEVS-TD0365]: "FCS_CKM_EXT.4 selections", version as of 2018-10-12.
- [CCEVS-TD0386]: "Platform-Provided Verification of Update", version as of 2019-02-07.
- [CCEVS-TD0441]: "Updated TLS Ciphersuites for OS PP", version as of 2019-08-21.
- [CCEVS-TD0463]: "Clarification for FPT_TUD_EXT", version as of 2019-11-12.
- [CCEVS-TD0493]: "X.509v3 certificates when using digital signatures for Boot Integrity", version as of 2020-03-04.
- [CCEVS-TD0496]: "GPOS PP adds allow-with statement for VPN Client V2.1", version as of 2020-01-29.
- [CCEVS-TD0501]: "Cryptographic selections and updates for OS PP", version as of 2020-09-03.
- [CCEVS-TD0525]: "Updates to Certificate Revocation (FIA_X509_EXT.1)", version as of 2020-07-01.
- [CCEVS-TD0578]: "SHA-1 is no longer mandatory", version as of 2021-02-12.
- [CCEVS-TD0630]: "FCS_COP.1 requirements for Secure Shell", version as of 2022-06-17.

- [CCEVS-TD0649]: "Conformance claims for OS PP v4.2.1", version as of 2022-06-17.
- [CCEVS-TD0666]: "Ambiguous intent of FCS_SSHS_EXT.1 tests", version as of 2022-09-13.
- [CCEVS-TD0682]: "Addressing Ambiguity in FCS_SSHS_EXT.1 Tests", version as of 2022-12-13.
- [CCEVS-TD0694]: "FCS_SSH_EXT.1.3 Inconsistency", version as of 2022-12-14.
- [CCEVS-TD0695]: "Choice of 128 or 256 bit size in AES-CTR in SSH Functional Package.", version as of 2022-12-14.
- [CCEVS-TD0715]: "Updates to FIA_X509_EXT.1 for Exception Processing and Test Conditions ", version as of 2023-04-04.
- [CCEVS-TD0777]: "Clarification to Selections for Auditable Events for FCS_SSH_EXT.1", version as of 2023-08-23.
- [GER]: "Guidelines for Evaluation Reports according to Common Criteria version 3.1", version 2.0 as of 2010-07-01.
- [JIL01]: "Joint Interpretation Library: Collection of Developer Evidence", version 1.5 as of January 2012.
- [VB-Prod]: "Verfahrensbeschreibung zur Zertifizierung von Produkten (VB-Produkte)", version 4.0 as of 2023-06-01.

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 Security Functional Requirements

2.1.1 Security audit (FAU)

2.1.1.1 Audit Data Generation (FAU_GEN.1)

FAU_GEN.1.1

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

Assurance Activity AA-FAU_GEN.1.1-AGD-01

> *The evaluator will check the administrative guide and ensure that it lists all of the auditable events. The evaluator will check to make sure that every audit event type selected in the ST is included.*

Summary

The evaluator examined 4.3 "Configuring the Audit Subsystem" of [ECG] which provides related guidance for auditing. This section refers to the man pages `auditd(8)`, `auditd.conf(5)`, and `auditctl(8)`. The evaluator examined these man pages. `auditd(8)` describes the audit log management daemon, `auditd.conf(5)` describes configuration file for the audit daemon, and `auditctl(8)` is the utility used for configuring the audit rules. Additional detail is provided by the online documentation https://documentation.suse.com/sles/15-SP4/html/SLES-all/cha-audit-scenarios.html . The evaluator noted that, although [ECG] does not provide an explicit listing of the auditable events selected in [ST] or explicit description of each audit record, these man pages (as well as the additional man pages they referred to, e.g., `aureport`) together provide a pretty comprehensive description of all auditable events implemented by the Linux TOE. The online documentation should be useful to make the content more accessible.

The evaluator also took into consideration that the TOE type is a general purpose Linux operating system which comes with a rather mature and expansive user documentation system. The evaluator examined the relevant man pages and the online guidance cited above and determined that they provide sufficient coverage of all the auditable events selected in [ST]. Thus, the evaluator accepts the man pages as appropriate operational guidance related to auditing.

Test Assurance Activities

Assurance Activity AA-FAU_GEN.1.1-ATE-01

The evaluator will test the OS's ability to correctly generate audit records by having the TOE generate audit records for the events listed in the ST. This should include all instance types of an event specified. When verifying the test results, the evaluator will 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.

Summary

The evaluator verified audit record generation for all events defined in FAU_GEN.1 of the ST. The evaluator examined the audit format and verified that the audit logs included all the necessary information.

FAU_GEN.1.2

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

Assurance Activity AA-FAU_GEN.1.2-AGD-01

The evaluator will check the administrative guide and ensure that it provides a format for audit records. Each audit record format type must be covered, along with a brief description of each field. The evaluator will ensure that the fields contains the information required.

Summary

As stated previously, section 4.3 "Configurating the Audit Subsystem" of [ECG] provides related guidance for auditing.

This section refers to the man pages `auditd(8)`, `auditd.conf(5)`, and `auditctl(8)`. The evaluator examined these man pages. `auditd(8)` describes the audit log management daemon, `auditd.conf(5)` describes configuration file for the audit daemon, and `auditctl(8)` is the utility used for configuring the audit rules. The evaluator noted that, although [ECG] does not provide an explicit listing of the auditable events selected in [ST] or explicit description of each audit record, these man pages (as well as the additional man pages they referred to, e.g., `aureport`) together provide a pretty comprehensive description of all auditable events implemented by the Linux TOE.

The evaluator also performed a thorough examination of the relevant man pages and determined that they provide sufficient coverage of all the auditable events selected in [ST].

Test Assurance Activities

Assurance Activity AA-FAU_GEN.1.2-ATE-01

The evaluator shall test the OS's ability to correctly generate audit records by having the TOE generate audit records for the events listed in the ST. The evaluator will ensure the audit records generated during testing match the format specified in the administrative guide, and that the fields in each audit record provide the required information.

Summary

The evaluator verified audit record generation for all events defined in FAU_GEN.1 of the ST. The evaluator examined the audit format and verified that the audit logs included all the necessary information.

2.1.2 Cryptographic support (FCS)

2.1.2.1 Cryptographic Key Generation (FCS_CKM.1)

TSS Assurance Activities

Assurance Activity AA-FCS_CKM.1-ASE-01

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

Summary

The schemes specified in [ST] are RSA, ECC and FFC. The schemes including the specified key sizes and the reference to the usage in TSS are listed in the following:

- The RSA scheme uses cryptographic key sizes of 2048-bit or greater as also defined in section 7.2.2.1 FCS_CKM.1 Cryptographic Key Generation of the [ST].
- The ECC scheme uses the "NIST curves" P-256, P-384 and P-521 as also defined in section 7.2.2.1 FCS_CKM.1 Cryptographic Key Generation of the [ST].
- The FFC scheme uses "Safe-Primes as defined by the NIST Special Publication 800-56A Revision 3" as also defined in section 7.2.2.1 FCS_CKM.1 Cryptographic Key Generation of the [ST].

Guidance Assurance Activities

Assurance Activity AA-FCS_CKM.1-AGD-01

The evaluator will verify that the AGD guidance instructs the administrator how to configure the OS to use the selected key generation scheme(s) and key size(s) for all uses defined in this PP.

Summary

The developer provided a large set of documentation comprising general administrator and end-user guidance.

It includes the following:

- Manual pages (man pages) [MANPAGES]: Each describes one command or one interface function. Collectively, they contain descriptions of all security-relevant TOE interfaces.
- Evaluated configuration guide (ECG) [ECG]: Covers the installation procedure and specific security topics relevant in the context of this evaluation. It takes precedence over all other user documentation in case there is a conflict of information.

According to the ST, the TOE supports (only) SSHv2 for interactive usage by remote entities.

The evaluator examined section 3.11.3 "SSH key-based authentication" of [ECG], which states that to generate keys that can be used for key-based authentication, the tool `ssh-keygen` is provided. Section 3.21.6 "Cryptographic key handling" states explicitly the cryptographic mechanisms (algorithms and key sizes) that must be used for SSH.

The details provided are "Encryption algorithms" (including the key size in each case), "Public key algorithms", "MAC algorithms" and "Key exchange".

The evaluator examined the man pages of `ssh-keygen(8)` and verified that it allows for specification of the keys and the key sizes listed in [ECG].

Test Assurance Activities

Assurance Activity AA-FCS_CKM.1-ATE-01

Evaluation Activity Note: The following tests may require the vendor to furnish a developer environment and developer tools that are typically not available to end-users of the OS.

Key Generation for FIPS PUB 186-4 RSA Schemes

The evaluator will verify the implementation of RSA Key Generation by the OS 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:

1. *Random Primes:*
   - *Provable primes*
   - *Probable primes*
2. *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 must seed the TSF key generation routine with sufficient data to deterministically generate the RSA key pair. This includes the random seed(s), 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 will verify the correctness of the TSF's implementation by comparing values generated by the TSF with those generated from a known good implementation.

If possible, the Random Probable primes method should also be verified against a known good implementation as described above. Otherwise, the evaluator will have the TSF generate 10 keys pairs for each supported key length nlen and verify:

- $n = p \cdot q$,
- *p and q are probably prime according to Miller-Rabin tests,*
- $GCD(p-1,e) = 1$,
- $GCD(q-1,e) = 1$,
- $2^{16} \leq e \leq 2^{256}$ *and e is an odd integer,*
- $|p-q| > 2^{nlen/2 - 100}$,
- $p \geq 2^{nlen/2 -1/2}$,
- $q \geq 2^{nlen/2 -1/2}$,
- $2^{(nlen/2)} < d < LCM(p-1,q-1)$,
- $e \cdot d = 1 \bmod LCM(p-1,q-1)$.

Summary

The test setup involved the lab's ACVP tool which used the test vectors obtained from the NIAP server to exercise the TSFI for the cryptographic function responses. Because of schema requirements, an ACVP tool function was used were the TOE test vector responses were compared against the responses of a local reference implementation under control of the lab. When these responses match the test was considered successful.

This test is covered by CAVP-like tests that are performed as part of this evaluation, notably the OpenSSL RSA keyGen entries in the validation output of the ACVP tool. All tests passed.

Assurance Activity AA-FCS_CKM.1-ATE-02

Key Generation for Elliptic Curve Cryptography (ECC)

FIPS 186-4 ECC Key Generation Test

For each supported NIST curve, i.e., P-256, P-384 and P-521, the evaluator will 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 will submit the generated key pairs to the public key verification (PKV) function of a known good implementation.

FIPS 186-4 Public Key Verification (PKV) Test

For each supported NIST curve, i.e., P-256, P-384 and P-521, the evaluator will 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 will obtain in response a set of 10 PASS/FAIL values.

Summary

Please see setup for cryptographic tests for a summary of the setup and result verification.

This test is covered by CAVP-like tests that are performed as part of this evaluation, notably the OpenSSL ECDSA keygen entries in the validation output of the ACVP tool. All tests passed.

Assurance Activity AA-FCS_CKM.1-ATE-03

Key Generation for Finite-Field Cryptography (FFC)

The evaluator will 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 2 ways (or methods) to generate the cryptographic prime q and the field prime p:

- *Cryptographic and Field Primes:*
  - *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:

- *Cryptographic Group Generator:*
  - *Generator g constructed through a verifiable process*
  - *Generator g constructed through an unverifiable process*

The Key generation specifies 2 ways to generate the private key x:

- *Private Key:*
  - *len(q) bit output of RBG where $1 \le x \le q-1$*
  - *len(q) + 64 bit output of RBG, followed by a mod q-1 operation where $1 \le x \le q-1$*

The security strength of the RBG must 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/or the group generator g for a verifiable process, the evaluator must seed the TSF parameter generation routine with sufficient data to deterministically generate the parameter set. For each key length supported, the evaluator will have the TSF generate 25 parameter sets and key pairs. The evaluator will verify the correctness of the TSF's implementation by comparing values generated by the TSF with those generated from a known good implementation. Verification must also confirm:

- *g != 0,1*
- *q divides p-1*
- *$g^q \bmod p = 1$*
- *$g^x \bmod p = y$*

for each FFC parameter set and key pair.

Summary

This test is covered by CAVP-like tests that are performed as part of this evaluation, notably the DSA keygen entries in the validation output of the ACVP tool. All tests passed.

Assurance Activity AA-FCS_CKM.1-ATE-04

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

[TD0501] Testing for FFC Schemes using Diffie-Hellman group 14 and/or "safe-prime" groups is done as part of testing in FCS_CKM.2.1

Summary

Please see setup for cryptographic tests for a summary of the setup and result verification.

This test is covered by CAVP-like tests that are performed as part of this evaluation, notably the OpenSSL safePrimes entries in the validation output of the ACVP tool. All tests passed.

2.1.2.2 Cryptographic Key Establishment (FCS_CKM.2)

TSS Assurance Activities

Assurance Activity AA-FCS_CKM.2-ASE-01

The evaluator will 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 will examine the TSS to verify that it identifies the usage for each scheme.

Summary

The key establishment schemes are RSAES-PKCS1-v1_5 RSA-based key establishment, Elliptic curve-based key establishment and Finite field-based key establishment. All three schemes are stated in section 7.2.2.2 FCS_CKM.2 Cryptographic Key Establishment of the TSS. Further the key establishment schemes match the key generation schemes in FCS_CKM.1.1.

Assurance Activity AA-FCS_CKM.2-ASE-02

SP800-56B Key Establishment Schemes

The evaluator will verify that the TSS describes whether the OS acts as a sender, a recipient, or both for RSA-based key establishment schemes.

The evaluator will ensure that the TSS describes how the OS handles decryption errors. In accordance with NIST Special Publication 800-56B, the OS must not reveal the particular error that occurred, either through the contents of any outputted or logged error message or through timing variations.

Summary

The evaluator determined that for the RSA-based key establishment schemes the OS is described as recipient in the TSS. How the OS handles decryption errors is described in section 7.1.2.3 of the TSS.

Guidance Assurance Activities

Assurance Activity AA-FCS_CKM.2-AGD-01

The evaluator will verify that the AGD guidance instructs the administrator how to configure the OS to use the selected key establishment scheme(s).

Summary

[ECG] section 3.21.6 "Cryptographic key handling" provides related guidance to configure OpenSSH and TLS. The detailed configuration of the OpenSSH daemon is in the `sshd_config` man page of [MANPAGES].

Per the sections, configuration for the SSH server and SSH client requires specifying the configuration files `/etc/ssh/sshd_config`, with the following configuration options including options for key established schemes:

- Encryption algorithms: AES128-CBC, AES256-CBC, AES128-GCM@openssh.com and AES256-GCM@openssh.com
- Public key algorithms: RSA-SHA2-256, RSA-SHA2-512, ECDSA-SHA2-NISTP384, ECDSA-SHA2-NISTP521
- MAC algorithms: HMAC-SHA2-256, HMAC-SHA2-512, AES128-GCM@openssh.com, AES256-GCM@openssh.com
- Key Exchange algorithms: Diffie-Hellman-group14-SHA256, Diffie-Hellman-group16-SHA512, Diffie-Hellman-group18-SHA512, ECDH-SHA2-NISTP256, ECDH-SHA2-NISTP384 or ECDH-SHA2-NISTP521

For TLS the following cipher suites are mandated:

- TLS_DHE_RSA_WITH_AES_128_CBC_SHA256
- TLS_DHE_RSA_WITH_AES_128_GCM_SHA256
- TLS_DHE_RSA_WITH_AES_256_GCM_SHA384
- TLS_DHE_RSA_WITH_AES_256_CBC_SHA256
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256
- TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384

in addition, the following supported groups extensions in the client hello are mandated:

- secp256r1
- secp384r1
- secp521r1

The evaluator checked, that the information provided in [ECG] is internally consistent and consistent with the ST, section 6.1.2.13 "SSH Protocol (FCS_SSH_EXT.1)".

Section 3.21.7 "Cryptographic key generation and establishment" provide general guidance on key generation mandating the following for key generation:

- RSA with key sizes of 2048-bit, 3072-bit, and 4096 bit
- Elliptic Curve Cryptography (ECC) using the NIST curves NIST-P 256, NIST P-384, or NIST P-521
- Finite-Field Cryptography (FFC) using approved Safe-Prime Groups as specified in NIST Special Publication 800-56A Revision 3

For key establishment the following is mandated:

- RSA with RSAES-PKCS1-v1_5 as specified RFC 8017
- Elliptic Curve Cryptography (ECC) with ECC CDH
- Finite-Field Cryptography (FFC) with FFC DH

Test Assurance Activities

Assurance Activity AA-FCS_CKM.2-ATE-01

Evaluation Activity 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 Establishment Schemes

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

SP800-56A Key Establishment Schemes

The evaluator will verify the OS'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 the OS has implemented the components of the key agreement scheme according to the specifications in the Recommendation. These components include the calculation of the discrete logarithm cryptography (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 will 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 MAC data and the calculation of MAC tag.

Function Test

The Function test verifies the ability of the OS to implement the key agreement schemes correctly. To conduct this test the evaluator will generate or obtain test vectors from a known good implementation of the OS's 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 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 will obtain the DKM, the corresponding OS's public keys (static and/or ephemeral), the MAC tag(s), and any inputs used in the KDF, such as the Other Information field OI and OS id fields.

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

The evaluator will 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 OS shall perform the above for each implemented approved MAC algorithm.

Validity Test

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

The evaluator will inject an error in some of the test vectors to test that the OS 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 MAC'd, or the generated MAC tag. If the OS 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 OS's static private key to assure the OS detects errors in the public key validation function and/or 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 OS shall use these modified test vectors to emulate the key agreement scheme using the corresponding parameters. The evaluator will compare the OS's results with the results using a known good implementation verifying that the OS detects these errors.

Summary

Please see setup for cryptographic tests for a summary of the setup and result verification.

This test is covered by CAVP-like tests that are performed as part of this evaluation, notably the OpenSSL KDF TLS, KAS-ECC-SSC, and KAS-FFC-SSC entries in the validation output of the ACVP tool. All tests passed.

Assurance Activity AA-FCS_CKM.2-ATE-02

> ***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.*

Summary

Please see setup for cryptographic tests for a summary of the setup and result verification.

This test is covered by CAVP-like tests that are performed as part of this evaluation, notably the OpenSSL RSA sigVer validation output of the ACVP tool. All tests passed.

Assurance Activity AA-FCS_CKM.2-ATE-03

> ***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.*

Summary

Please see setup for cryptographic tests for a summary of the setup and result verification.

This test is covered by CAVP-like tests that are performed as part of this evaluation, notably the OpenSSL KAS-FFC-SSC entries validation output of the ACVP tool. All tests passed.

Assurance Activity AA-FCS_CKM.2-ATE-04

> ***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

Please see setup for cryptographic tests for a summary of the setup and result verification.

This test is covered by CAVP-like tests that are performed as part of this evaluation, notably the KAS-FFC-SSC entries for OpenSSL validation output of the ACVP tool. All tests passed.

2.1.2.3 Cryptographic Key Destruction (FCS_CKM_EXT.4)

TSS Assurance Activities

Assurance Activity AA-FCS_CKM_EXT.4-ASE-01

> *The evaluator examines the TSS to ensure it describes how the keys are managed in volatile memory. This description includes details of how each identified key is introduced into volatile memory (e.g. by derivation from user input, or by unwrapping a wrapped key stored in non-volatile memory) and how they are overwritten.*

The evaluator will check to ensure the TSS lists each type of key that is stored in in non-volatile memory, and identifies how the TOE interacts with the underlying platform to manage keys (e.g., store, retrieve, destroy). The description includes details on the method of how the TOE interacts with the platform, including an identification and description of the interfaces it uses to manage keys (e.g., file system APIs, platform key store APIs).

If the ST makes use of the open assignment and fills in the type of pattern that is used, the evaluator examines the TSS to ensure it describes how that pattern is obtained and used. The evaluator will verify that the pattern does not contain any CSPs.

The evaluator will check that the TSS identifies any configurations or circumstances that may not strictly conform to the key destruction requirement.

*[TD0365] If the selection **"destruction of all key encrypting keys protecting target key according to FCS_CKM_EXT.4.1, where none of the KEKs protecting the target key are derived"** is included the evaluator shall examine the TOE's keychain in the TSS and identify each instance when a key is destroyed by this method. In each instance the evaluator shall verify all keys capable of decrypting the target key are destroyed in accordance with a specified key destruction method in FCS_CKM_EXT.4.1 The evaluator shall verify that all of the keys capable of decrypting the target key are not able to be derived to reestablish the keychain after their destruction.*

Summary

The evaluator determined that the management of keys in volatile memory, which keys are handled in volatile memory and how they are cleared / destroyed, is described in section 7.2.2.3 "Cryptographic Key Destruction".

The interaction of the TOE with the underlying platform to manage keys which are stored in non-volatile memory is explained in section 7.2.2.3 by referring to the employed cryptographic scheme (LUKS). This section also contains a description how KEKs and corresponding volume keys used by LUKS are being derived, handled and cleared in accordance wih the specified destruction method. The section describes that with the destruction of the master volume key the content of the disk is cryptographically erased. The evaluator could thus verify that all keys capable of decrypting the target key are destroyed in accordance with a specified key destruction method. As the user passphrase data used to derive the KEK and thus the keys capable of decrypting the target (master volume) key are destroyed / cleared with the removal of power, they can no longer be used for their re-establishment after their destruction.

Section 7.2.2.3 describes for non-volatile memory on an SSD, that "the TOE provides the tool `fstrim`. After a deletion of a file with sensitive data, this tool uses the SSD TRIM command to inform the SSD to discard unused blocks bypassing wear leveling". For non-volatile memory in a HDD section 7.2.2.3 states "the tool `shred` is available that overwrites files multiple times with random data".

The evaluator concluded that all requirements for the destruction of keys in volatile and non-volatile memory are described in the TSS.

Guidance Assurance Activities

Assurance Activity AA-FCS_CKM_EXT.4-AGD-01

There are a variety of concerns that may prevent or delay key destruction in some cases. The evaluator will check that the guidance documentation identifies configurations or circumstances that may not strictly conform to the key destruction requirement, and that this description is consistent with the relevant parts of the TSS and any other relevant Required Supplementary Information. The evaluator will check that the guidance documentation provides guidance on situations where key destruction may be delayed at the physical layer and how such situations can be avoided or mitigated if possible.

Some examples of what is expected to be in the documentation are provided here.

When the TOE does not have full access to the physical memory, it is possible that the storage may be implementing wear-leveling and garbage collection. This may create additional copies of the key that are logically inaccessible but persist physically. In this case, to mitigate this the drive should support the TRIM command and implements garbage collection to destroy these persistent copies when not actively engaged in other tasks.

Drive vendors implement garbage collection in a variety of different ways, as such there is a variable amount of time until data is truly removed from these solutions. There is a risk that data may persist for a longer amount of time if it is contained in a block with other data not ready for erasure. To reduce this risk, the operating system and file system of the OE should support TRIM, instructing the non-volatile memory to erase copies via garbage collection upon their deletion. If a RAID array is being used, only set-ups that support TRIM are utilized. If the drive is connected via PCI-Express, the operating system supports TRIM over that channel.

The drive should be healthy and contains minimal corrupted data and should be end-of-lifed before a significant amount of damage to drive health occurs, this minimizes the risk that small amounts of potentially recoverable data may remain in damaged areas of the drive.

Summary

[ECG] section 3.8 "Secure erasure" provides related guidance to ensure key destruction.

In order to securely erase any key material held in files, section 3.8 states that it is mandatory to delete those files and that, once the files are deleted, `/usr/sbin/fstrim` has to be invoked in case of SSDs, and `/usr/sbin/shred` in case of HDDs.

In section 7.4 "SSH key-based authentication", [ECG] also anticipates that unprivileged users cannot run the `fstrim` command, stating

> *Please also note that you MUST contact your system administrator to securely erase your SSH key.*

Beyond that, [ECG] does not provide further instructions (e.g. to retire worn drives early). It does however contain a caveat regarding the key deletion strategy be removing power to memory in section 3.21.6:

> *Please note, cryptographic keys kept in volatile memory are completely cleared only after a full powercycle.*

.

The TOE supports encryption of data at rest using AES. Chapter 2 of [ECG] advises to enable it. This helps to mitigate leakage of cryptographic material that may be temporarily stored on a disk if the encryption encompasses any temporary or swap partition that may be configured optionally.

Test Assurance Activities

Assurance Activity AA-FCS_CKM_EXT.4-ATE-01

- ***Test 1:*** *Applied to each key held as in volatile memory and subject to destruction by overwrite by the TOE (whether or not the value is subsequently encrypted for storage in volatile or non-volatile memory). In the case where the only selection made for the destruction method key was removal of power, then this test is unnecessary. The evaluator will:*
  1. *Record the value of the key in the TOE subject to clearing.*
  2. *Cause the TOE to perform a normal cryptographic processing with the key from Step #1.*
  3. *Cause the TOE to clear the key.*
  4. *Cause the TOE to stop the execution but not exit.*
  5. *Cause the TOE to dump the entire memory of the TOE into a binary file.*
  6. *Search the content of the binary file created in Step #5 for instances of the known key value from Step #1.*

  Steps 1-6 ensure that the complete key does not exist anywhere in volatile memory. If a copy is found, then the test fails.
- ***Test 2:*** *Applied to each key help in non-volatile memory and subject to destruction by the TOE. The evaluator will use special tools (as needed), provided by the TOE developer if necessary, to ensure the tests function as intended.*

1. *Identify the purpose of the key and what access should fail when it is deleted. (e.g. the data encryption key being deleted would cause data decryption to fail.)*
2. *Cause the TOE to clear the key.*
3. *Have the TOE attempt the functionality that the cleared key would be necessary for.*

The test succeeds if step 3 fails.

*[TD0365] Tests 3 and 4 do not apply for the selection "**instructing the underlying platform to destroy the representation of the key**", as the TOE has no visibility into the inner workings and completely relies on the underlying platform.*

- ***Test 3:** The following tests apply only to selection a), since the TOE in this instance has more visibility into what is happening within the underlying platform (e.g., a logical view of the media). In selection b), the TOE has no visibility into the inner workings and completely relies on the underlying platform, so there is no reason to test the TOE beyond test 2.*
  For selection a), the following tests are used to determine the TOE is able to request the platform to overwrite the key with a TOE supplied pattern.
  Applied to each key held in non-volatile memory and subject to destruction by overwrite by the TOE. The evaluator will use a tool that provides a logical view of the media (e.g., MBR file system):
  1. *Record the value of the key in the TOE subject to clearing.*
  2. *Cause the TOE to perform a normal cryptographic processing with the key from Step #1.*
  3. *Cause the TOE to clear the key.*
  4. *Search the logical view that the key was stored in for instances of the known key value from Step #1. If a copy is found, then the test fails.*
- ***Test 4:** Applied to each key held as non-volatile memory and subject to destruction by overwrite by the TOE. The evaluator will use a tool that provides a logical view of the media:*
  1. *Record the logical storage location of the key in the TOE subject to clearing.*
  2. *Cause the TOE to perform a normal cryptographic processing with the key from Step #1.*
  3. *Cause the TOE to clear the key.*
  4. *Read the logical storage location in Step #1 of non-volatile memory to ensure the appropriate pattern is utilized.*

  The test succeeds if correct pattern is used to overwrite the key in the memory location. If the pattern is not found the test fails.

Summary

Test 1 for volatile memory does not apply as the method of deletion is an organizational measure (power-cycle). Test 2 for and non-volatile data on HDDs the unrecoverable data in case of key deletion has been tested.

2.1.2.4 Cryptographic Operation - Encryption/Decryption (FCS_COP.1(1))

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

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

The evaluator will verify that the AGD documents contain instructions required to configure the OS to use the required modes and key sizes. The evaluator will execute all instructions as specified to configure the OS to the appropriate state.

Summary

[ECG] section 3.21.6 "Cryptographic key handling" provides related guidance to configure OpenSSH. The detailed configuration of the OpenSSH daemon is in the sshd_config man page of [MANPAGES].

Per section 3.21.4ff of [ECG], configuration for the SSH server requires specifying the configuration file `/etc/ssh/sshd_config`, with the following configuration options including options data encryption and decryption for:

- Encryption algorithms: AES128-CTR, AES256-CTR, AES128-CBC, AES256-CBC, AES128-GCM@openssh.com and AES256-GCM@openssh.com
- Public key algorithms: RSA-SHA2-256, RSA-SHA2-512, ECDSA-SHA2-NISTP384, ECDSA-SHA2-NISTP521
- MAC algorithms: HMAC-SHA2-256, HMAC-SHA2-512, AEAD_AES_128_GCM, AEAD_AES_256_GCM
- Key exchange: Diffie-Hellman-group14-SHA1, ECDH-SHA2-NISTP521 ECDH-SHA2-NISTP256, ECDH-SHA2-NISTP384, or Diffie-Hellman-group15-SHA512, Diffie-Hellman-group16-SHA512, Diffie-Hellman-group18-SHA512, ECDH-SHA2-NISTP256, Diffie-Hellman-group14-SHA256, Diffie-Hellman-group17-SHA512, ECDH-SHA2-NISTP384, ECDH-SHA2-NISTP521

The sshd configuration has been performed as part of the automated installation routine.

Test Assurance Activities

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

The evaluator will execute all instructions as specified to configure the OS to the appropriate state. The evaluator will perform all of the following tests for each algorithm implemented by the OS and used to satisfy the requirements of this PP:

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 will 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 will 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 will 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 will 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 will 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 will 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 will 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 will 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 will 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 will test the encrypt functionality by encrypting an i-block message where 1 < i ≤ 10. The evaluator will 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 will also test the decrypt functionality for each mode by decrypting an i-block message where 1 < i ≤10. The evaluator will 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 will 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 will test the decrypt functionality using the same test as for encrypt, exchanging CT and PT and replacing AES-CBC-Encrypt with AES-CBC-Decrypt.

Summary

This test is covered by CAVP-like tests that are performed as part of this evaluation, notably the AES-CBC entries for OpenSSL validation output of the ACVP tool. All tests passed.

Assurance Activity AA-FCS_COP.1-1-ATE-02

AES-GCM Monte Carlo Tests

The evaluator will 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 will 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 will 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 will compare the resulting values to those obtained by submitting the same inputs to a known good implementation.

AES-GCM Test

The evaluator will 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 will 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 will 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 will compare the resulting values to those obtained by submitting the same inputs to a known good implementation.

Summary

This test is covered by CAVP-like tests that are performed as part of this evaluation, notably the AES-GCM entries for OpenSSL validation output of the ACVP tool. All tests passed.

Assurance Activity AA-FCS_COP.1-1-ATE-03

AES-CCM Tests

The evaluator will 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 216 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 will 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 will 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 will 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 will 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 will 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 will 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 will supply 10 tuples that should FAIL and 5 that should PASS per set of 15.

Additionally, the evaluator will 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 AESCCMP Encapsulation Example and Section 2.2 Additional AES CCMP Test Vectors to further verify the IEEE 802.11-2007 implementation of AES-CCMP.

Summary

This test is covered by CAVP-like tests that are performed as part of this evaluation, notably the AES-CCM entries for OpenSSL validation output of the ACVP tool. All tests passed.

Assurance Activity AA-FCS_COP.1-1-ATE-04

XTS-AES Test

The evaluator will 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 nonzero 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 216 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 will 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 XTSAES decrypt.

Summary

This test is covered by CAVP-like tests that are performed as part of this evaluation, notably the AES-XTS entries for the kernel validation output of the ACVP tool. All tests passed.

Assurance Activity AA-FCS_COP.1-1-ATE-05

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

The evaluator will 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 will use the AES-KW authenticated-encryption function of a known good implementation.

The evaluator will 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 will 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 will 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.

Summary

No AES key wrap has been implemented by the TOE.

2.1.2.5 Cryptographic Operation - Hashing (FCS_COP.1(2))

TSS Assurance Activities

Assurance Activity AA-FCS_COP.1-2-ASE-01

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

Summary

The evaluator determined that the association of hash functions with other cryptographic functions is documented in section 7.2.2.5 FCS_COP.1(2) Cryptographic Operation - Hashing as following:

The hashing algorithms are used for signature services and HMAC services.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-FCS_COP.1-2-ATE-01

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 testmacs. The evaluator will 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 application that provides the evaluator with tools that are typically not found in the production application.

- ***Test 1:** Short Messages Test (Bit oriented Mode) - The evaluator will generate 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 evaluator will 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.*
- ***Test 2:** Short Messages Test (Byte oriented Mode) - The evaluator will generate 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 evaluator will 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.*

- ***Test 3:*** *Selected Long Messages Test (Bit oriented Mode) - The evaluator will generate 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 ≤ i ≤ m. The message text shall be pseudorandomly generated. The evaluator will 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.*
- ***Test 4:*** *Selected Long Messages Test (Byte oriented Mode) - The evaluator will generate 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 ≤ i ≤ m/8. The message text shall be pseudorandomly generated. The evaluator will 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.*
- ***Test 5:*** *Pseudorandomly Generated Messages Test - This test is for byte-oriented implementations only. The evaluator will 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 evaluator will then formulate a set of 100 messages and associated digests by following the algorithm provided in Figure 1 of [SHAVS]. The evaluator will then ensure that the correct result is produced when the messages are provided to the TSF.*

Summary

This test is covered by CAVP-like tests that are performed as part of this evaluation, notably the SHA2-256, SHA2-512 for OpenSSL and kernel validation output of the ACVP tool. All tests passed.

2.1.2.6 Cryptographic Operation - Signing (FCS_COP.1(3))

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-FCS_COP.1-3-ATE-01

The evaluator will perform the following activities based on the selections in the ST.

The following tests require the developer to provide access to a test application that provides the evaluator with tools that are typically not found in the production application.

ECDSA Algorithm Tests

- ***Test 1:*** *ECDSA FIPS 186-4 Signature Generation Test. For each supported NIST curve (i.e., P-256, P-384 and P-521) and SHA function pair, the evaluator will 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 will use the signature verification function of a known good implementation.*
- ***Test 2:*** *ECDSA FIPS 186-4 Signature Verification Test. For each supported NIST curve (i.e., P-256, P-384 and P-521) and SHA function pair, the evaluator will 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 will verify that 5 responses indicate success and 5 responses indicate failure.*

Summary

This test is covered by CAVP-like tests that are performed as part of this evaluation, notably the ECDSA (sigGen, sigVer validation output of the ACVP tool entries for OpenSSL). All tests passed.

Assurance Activity AA-FCS_COP.1-3-ATE-02

RSA Signature Algorithm Tests

- ***Test 1:*** *Signature Generation Test. The evaluator will verify the implementation of RSA Signature Generation by the OS using the Signature Generation Test. To conduct this test the evaluator must generate or obtain 10 messages from a trusted reference implementation for each modulus size/SHA combination supported by the TSF. The evaluator will have the OS use its private key and modulus value to sign these messages. The evaluator will verify the correctness of the TSF's signature using a known good implementation and the associated public keys to verify the signatures.*
- ***Test 2:*** *Signature Verification Test. The evaluator will perform the Signature Verification test to verify the ability of the OS to recognize another party's valid and invalid signatures. The evaluator will 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, and/or signatures. The evaluator will verify that the OS returns failure when validating each signature.*

Summary

This test is covered by CAVP-like tests that are performed as part of this evaluation, notably the RSA (sigGen, sigVer validation output of the ACVP tool entries for OpenSSL). All tests passed.

2.1.2.7 Cryptographic Operation - Keyed-hash Message Authentication (FCS_COP.1(4))

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-FCS_COP.1-4-ATE-01

The evaluator will perform the following activities based on the selections in the ST.

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

Summary

This test is covered by CAVP-like tests that are performed as part of this evaluation, notably the HMAC-SHA2-256, HMC-SHA2-512 entries for OpenSSL validation output of the ACVP tool. All tests passed.

2.1.2.8 Random Bit Generation (FCS_RBG_EXT.1)

FCS_RBG_EXT.1.1

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-FCS_RBG_EXT.1.1-ATE-01

The evaluator will perform the following tests:

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

If the RNG 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 will 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 RNG 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 will 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 list contains 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 Derivation Function 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 less than or equal to 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 support, the evaluator will 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

This test is covered by CAVP-like tests that are performed as part of this evaluation, notably the ctrDRBG for OpenSSL, and hmacDRBG for the kernel validation output of the ACVP tool. All tests passed.

FCS_RBG_EXT.1.2

TSS Assurance Activities

Assurance Activity AA-FCS_RBG_EXT.1.2-ASE-01

Documentation shall be produced - and the evaluator will perform the activities - in accordance with Appendix E in [OSPPv4.2.1] *and the Clarification to the Entropy Documentation and Assessment Annex in* [CCEVS-EDAC].

In the future, specific statistical testing (in line with NIST SP800-90B) will be required to verify the entropy estimates.

Summary

The evaluator analysed the requirements stated in Appendix E of [OSPPv4.2.1] and determined that the minimum entropy is stated as 256 bit and the noise source as software-based in FCS_RBG_EXT.1.2 of the [ST].

The Entropy-Assessment-Report ([EAR]) verifies the consistency of the noise source and minimum entropy as stated in the [ST].

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-FCS_RBG_EXT.1.2-ATE-01

> *Documentation shall be produced - and the evaluator will perform the activities - in accordance with Appendix E in* [OSPPv4.2.1] *and the Clarification to the Entropy Documentation and Assessment Annex in* [CCEVS-EDAC].
>
> *In the future, specific statistical testing (in line with NIST SP800-90B) will be required to verify the entropy estimates.*

Summary

The entropy comes from the Linux RNG for both the DRBG of kernel and OpenSSL. The Linux entropy is being tested as part of a separate project dedicated to the Linux RNG which is documented in [BSIRNG]. There, it is stated that the min-entropy estimation is done on the high-resolution time stamp and involves the following calculations as per SP800-90B

- Most Common Value Estimate
- Collision Estimate
- Markov Estimate
- Compression Estimate
- t-Typle Estimate
- Longest Repeated Substring (LRS) Estimate
- Multi Most Common in Window Prediction Estimate
- Lag Prediction Estimate
- MultiMMC Prediction Estimate
- LZ78Y Prediction Estimate

The min-entropy estimates obtained using the SP800-90B Entropy Assessment tool for IRQ noise sources. For the interrupt noice source, the min-entropy per interrupt event was estimated. Each interrupt event normally contains 64 bits of data, but only the 8 least-significant bits were considered in this study. Therefore, taking the min-entropy values and dividing by 8-bits give entropy rates. The determination of the entropy per interrupt is important because a certain number of interrupts events are used as input to an internal RNG pool, which also affects Linux entropy estimator. As a result, the evaluators determined that the actually gathered entropy is higher than estimated by the TOE. Details on the calculations and the actual values are part of [BSIRNG].

2.1.2.9 SSH Protocol (FCS_SSH_EXT.1)

FCS_SSH_EXT.1.1

TSS Assurance Activities

Assurance Activity AA-FCS_SSH_EXT.1.1-ASE-01

The evaluator shall ensure that the selections indicated in the ST are consistent with selections in this and subsequent components. Otherwise, this SFR is evaluated by activities for other SFRs.

Summary

The evaluator examined [ST] sections 6.1.2.13 to 6.1.2.15 where the SSH specific SFRs (FCS_SSH_EXT.1, FCS_SSHC_EXT.1 and FCS_SSHS_EXT.1) are instantiated and searched the SFRs for inconsistencies between themselves, as well as the relevant requirements originating from OSPP, such as FCS_CKM.1, FCS_CKM.2, FSC_COP.1/*. No inconsistencies could be identified.

It should be noted that the [SSHPKGv1.0] allows for the use of the Edwards curve 25519 based algorithms. These algorithms are not covered by the claims in [OSPPv4.2.1] and can thus not be claimed by conforming STs.

The TOE's SSH support is described in the TSS in sections 7.2.2.13 SSH Protocol, 7.2.2.14 SSH Protocol - Client and 7.2.2.15 SSH Protocol - Server. The evaluator examined the consistency of these sections as well as consistency between these sections and sections 6.1.2.13 to 6.1.2.15 in the course of evaluating ASE_TSS.1-2

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

No assurance activities defined.

FCS_SSH_EXT.1.2

TSS Assurance Activities

Assurance Activity AA-FCS_SSH_EXT.1.2-ASE-01

The evaluator shall check to ensure that the authentication methods listed in the TSS are identical to those listed in this SFR component; and, ensure if password-based authentication methods have been selected in the ST then these are also described; and, ensure that if keyboard-interactive is selected, it describes the multifactor authentication mechanisms provided by the TOE.

Summary

Password based and SSH key based authentication is described in the context of FIA_UAU.5. Key based authentication is properly described by referring to the corresponding RFC4252.

Guidance Assurance Activities

Assurance Activity AA-FCS_SSH_EXT.1.2-AGD-01

The evaluator shall check the guidance documentation to ensure the configuration options, if any, for authentication mechanisms provided by the TOE are described.

Summary

The available authentication mechanisms supported by the TOE are described in [ECG], section 6.2 "Authentication" as well as section 6.4 "SSH key-based authentication".

The [ECG] describes in section 3.21.5, that public key based authentication or password based authentication can be used. The available public key based authentication algorithms are described in [ECG] section 3.21.6 "Cryptographic key handling", where the supported public key based authentication algorithms are listed:

RSA-SHA2-256, RSA-SHA2-512, ECDSA-SHA2-NISTP384, ECDSA-SHA2-NISTP521

[ECG], section 3.21.6 contains the following statement

The following cryptographic mechanisms MUST be used for SSH

. The manual page for `sshd_config` for the server and `ssh_config` for the client describes that `PubkeyAcceptedKeyTypes` as the relevant configuration directive.

b

It should be noted that the SSH configuration is generated automatically during the installation process.

Password based authentication can be configured with regard to the following aspects:

- Configuring password policy (section 3.9). That is related to the guidance given to users in section 6.2 "Password policy"
- Locking out users after a number of unsuccessful attempts (section 3.11.5)

Test Assurance Activities

Assurance Activity AA-FCS_SSH_EXT.1.2-ATE-01

- *Test 1: [conditional] If the TOE is acting as SSH Server:*
  - *a) The evaluator shall use a suitable SSH Client to connect to the TOE, enable debug messages in the SSH Client, and examine the debug messages to determine that only the configured authentication methods for the TOE were offered by the server.*
  - *b) [conditional] If the SSH server supports X509 based Client authentication options:*
    - *a) The evaluator shall initiate an SSH session from a client where the username is associated with the X509 certificate. The evaluator shall verify the session is successfully established.*
    - *b) Next the evaluator shall use the same X509 certificate as above but include a username not associated with the certificate. The evaluator shall verify that the session does not establish.*
    - *c) Finally, the evaluator shall use the correct username (from step a above) but use a different X509 certificate which is not associated with the username. The evaluator shall verify that the session does not establish.*
- *Test 2: [conditional] If the TOE is acting as SSH Client, the evaluator shall test for a successful configuration setting of each authentication method as follows:*
  - *a) The evaluator shall initiate a SSH session using the authentication method configured and verify that the session is successfully established.*
  - *b) Next, the evaluator shall use bad authentication data (e.g. incorrectly generated certificate or incorrect password) and ensure that the connection is rejected.*

  Steps a-b shall be repeated for each independently configurable authentication method supported by the server.
- *Test 3: [conditional] If the TOE is acting as SSH Client, the evaluator shall verify that the connection fails upon configuration mismatch as follows:*
  - *a) The evaluator shall configure the Client with an authentication method not supported by the Server.*
  - *b) The evaluator shall verify that the connection fails.*

If the Client supports only one authentication method, the evaluator can test this failure of connection by configuring the Server with an authentication method not supported by the Client. In order to facilitate this test, it is acceptable for the evaluator to configure an authentication method that is outside of the selections in the SFR.

Summary

For the client, the evaluator tested allowed and unallowed authentication mechanisms including wrong credential behavior and observed that the connection was rejected in the latter two cases.

For the server, the evaluator tested that only the autentication methods password an publickey are contained as allowed methods in the connection debug log.

FCS_SSH_EXT.1.3

TSS Assurance Activities

Assurance Activity AA-FCS_SSH_EXT.1.3-ASE-01

The evaluator shall check that the TSS describes how “large packets” are detected and handled.

Summary

Section 7.2.4.13 describes how large packets are detected (passing a threshold of 262144 bytes) and handled (the connection is closed).

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-FCS_SSH_EXT.1.3-ATE-01

- *Test 1: The evaluator shall demonstrate that the TOE accepts the maximum allowed packet size.*
- *Test 2: This test is performed to verify that the TOE drops packets that are larger than size specified in the component.*
  - *a) The evaluator shall establish a successful SSH connection with the peer.*
  - *b) Next the evaluator shall craft a packet that is slightly larger than the maximum size specified in this component and send it through the established SSH connection to the TOE. The packet should not be greater than the maximum packet size + 16 bytes. If the packet is larger, the evaluator shall justify the need to send a larger packet.*
  - *c) Evaluator shall verify that the packet was dropped by the TOE by reviewing the TOE audit log for a dropped packet audit.*

Summary

For the client and server, the evaluator used a max length packet as well as one that is 16 bytes larger by the respective peer and observed that the packet was accepted in the former and rejected in the latter case by the TOE.

FCS_SSH_EXT.1.4

TSS Assurance Activities

Assurance Activity AA-FCS_SSH_EXT.1.4-ASE-01

The evaluator will check the description of the implementation of SSH in the TSS to ensure the encryption algorithms supported are specified. The evaluator will check the TSS to ensure that the encryption algorithms specified are identical to those listed for this component.

Summary

Section 7.2.4.13 describes the supported encryption algorithms (aes128-cbc, aes256-cbc, AES128 GCM, AES256 GCM), which is consistent the SFR FSC_SSH_EXT.1.

Guidance Assurance Activities

Assurance Activity AA-FCS_SSH_EXT.1.4-AGD-01

The evaluator shall check the guidance documentation to ensure that it contains instructions to the administrator on how to ensure that only the allowed mechanisms are used in SSH connections with the TOE.

Summary

[ECG], section 3.21.6 contains the following statement

> *The following cryptographic mechanisms MUST be used for SSH*

. The manual page for `sshd_config` describes that `Ciphers` as the relevant configuration directive.

The available encryption algorithms are described in [ECG] section 3.21.6 "Cryptographic key handling", where the mandated algorithms are listed:

> *AES128-CBC, AES256-CBC, AES128-GCM@openssh.com and AES256-GCM@openssh.com*

Test Assurance Activities

Assurance Activity AA-FCS_SSH_EXT.1.4-ATE-01

The evaluator shall perform the following tests:

- ***Test 1:*** *The evaluator shall use a suitable SSH Client to connect to the TOE and examine the list of server host key algorithms in the SSH_MSG_KEXINIT packet sent from the server to the client to determine that only the configured server authentication methods for the TOE were offered by the server.*
- ***Test 2:*** *The evaluator shall test for a successful configuration setting of each server authentication method as follows. The evaluator shall initiate a SSH session using the authentication method configured and verify that the session is successfully established. Repeat this process for each independently configurable server authentication method supported by the server.*
- ***Test 3:*** *The evaluator shall configure the peer to only allow a mechanism that is not included in the ST selection. The evaluator shall attempt to connect to the TOE and observe that the TOE sends a disconnect message.*

Summary

The evaluator established a successful SSH connection and stored the debug output of the client/server side. From that debug output he observed that all encryption algorithms supported per ST where listed. The further verified that exiting the SSH session indeed terminates the connection. Finally, he verified that if only unsupported algorithms are advertized by the peer the TOE rejects the connection.

FCS_SSH_EXT.1.5

TSS Assurance Activities

Assurance Activity AA-FCS_SSH_EXT.1.5-ASE-01

The evaluator will check the description of the implementation of SSH in the TSS to ensure the hashing algorithms supported are specified. The evaluator will check the TSS to ensure that the hashing algorithms specified are identical to those listed for this component.

Summary

Section 7.2.4.13 describes the supported hahsing algorithms (hmac-sha2-256, hmac-sha2-512, AES GCM), which is consistent the SFR FSC_SSH_EXT.1.

Guidance Assurance Activities

Assurance Activity AA-FCS_SSH_EXT.1.5-AGD-01

The evaluator shall check the guidance documentation to ensure that it contains instructions to the administrator on how to ensure that only the allowed mechanisms are used in SSH connections with the TOE.

Summary

[ECG], section 3.21.6 contains the following statement

The following cryptographic mechanisms MUST be used for SSH

. The manual page for `sshd_config` describes that `MACs` as the relevant configuration directive.

The available integrity protecting authentication algorithms are described in [ECG] section 3.21.6 "Cryptographic key handling", where the mandated algorithms are listed:

HMAC-SHA2-256, HMAC-SHA2-512, AEAD_AES_128_GCM, AEAD_AES_256_GCM

Test Assurance Activities

Assurance Activity AA-FCS_SSH_EXT.1.5-ATE-01

- *Test 1: The evaluator shall use the test data collected in FCS_SSH_EXT.1.4, Test 1 to verify that appropriate mechanisms are advertised.*
- *Test 2: The evaluator shall configure an SSH peer to allow only a hashing algorithm that is not included in the ST selection. The evaluator shall attempt to establish an SSH connection and observe that the connection is rejected.*

Summary

The evaluator observed from the previous debug log that only the supported hash algorithms were present. He also verified that if only unsupported algorithms are advertized by the peer the TOE rejects the connection.

FCS_SSH_EXT.1.6

TSS Assurance Activities

Assurance Activity AA-FCS_SSH_EXT.1.6-ASE-01

The evaluator will check the description of the implementation of SSH in the TSS to ensure the shared secret establishment algorithms supported are specified. The evaluator will check the TSS to ensure that the shared secret establishment algorithms specified are identical to those listed for this component.

Summary

Section 7.2.4.13 describes the supported shared secret establishment algorithms (diffie-hellman-group14-sha256, diffie-hellman-group16-sha512, diffie-hellman-group18-sha512, ecdh-sha2-nistp256, ecdh-sha2-nistp384 or ecdh-sha2-nistp521), which is consistent the SFR FSC_SSH_EXT.1.

Guidance Assurance Activities

Assurance Activity AA-FCS_SSH_EXT.1.6-AGD-01

> *The evaluator shall check the guidance documentation to ensure that it contains instructions to the administrator on how to ensure that only the allowed mechanisms are used in SSH connections with the TOE.*

Summary

[ECG], section 3.21.6 contains the following statement

> *The following cryptographic mechanisms MUST be used for SSH*

. The manual page for `sshd_config` describes that `KexAlgorithms` as the relevant configuration directive.

The available key exchange algorithms are described in [ECG] section 3.21.6 "Cryptographic key handling", where the mandated algorithms are listed:

> *Diffie-Hellman-group14-SHA256, Diffie-Hellman-group16-SHA512, Diffie-Hellman-group18-SHA512, ECDH-SHA2-NISTP256, ECDH-SHA2-NISTP384 or ECDH-SHA2-NISTP521*

Test Assurance Activities

Assurance Activity AA-FCS_SSH_EXT.1.6-ATE-01

- *Test 1: The evaluator shall use the test data collected in FCS_SSH_EXT.1.4, Test 1 to verify that appropriate mechanisms are advertised.*
- *Test 2: The evaluator shall configure an SSH peer to allow only a key exchange method that is not included in the ST selection. The evaluator shall attempt to establish an SSH connection and observe that the connection is rejected.*

Summary

The evaluator observed from the previous debug log that only the supported key exchange algorithms were present. He also verified that if only unsupported algorithms are advertized by the peer the TOE rejects the connection.

FCS_SSH_EXT.1.7

TSS Assurance Activities

Assurance Activity AA-FCS_SSH_EXT.1.7-ASE-01

> *The evaluator will check the description of the implementation of SSH in the TSS to ensure the KDFs supported are specified. The evaluator will check the TSS to ensure that the KDFs specified are identical to those listed for this component.*

Summary

The supported algorithms are described in section 7.2.2.13 by referring to the relevant sections in RFC4253 and RFC5656. The KDFs specified in the TSS as thus identical to those claimed in the corresponding SFR.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

No assurance activities defined.

FCS_SSH_EXT.1.8

TSS Assurance Activities

Assurance Activity AA-FCS_SSH_EXT.1.8-ASE-01

The evaluator shall check the TSS to ensure that if the TOE enforces connection rekey or termination limits lower than the maximum values that these lower limits are identified.

In cases where hardware limitation will prevent reaching data transfer threshold in less than one hour, the evaluator shall check the TSS to ensure it contains:

a) An argument describing this hardware-based limitation and

b) Identification of the hardware components that form the basis of such argument.

For example, if specific Ethernet Controller or Wi-Fi radio chip is the root cause of such limitation, these subsystems shall be identified.

Summary

Section 7.2.2.13 describes the condition for rekeying: "After processing at most 2^30 bytes covering both sent and received data or the last re-key is more than 1 hour ago, the TOE initiates a re-keying, when the option RekeyLimit is set appropriately."

Guidance Assurance Activities

Assurance Activity AA-FCS_SSH_EXT.1.8-AGD-01

The evaluator shall check the guidance documentation to ensure that if the connection rekey or termination limits are configurable, it contains instructions to the administrator on how to configure the relevant connection rekey or termination limits for the TOE.

Summary

The [ECG] in section 3.21.5 does contain respective statements concerning the `RekeyLimit` configuration file directive..

Test Assurance Activities

Assurance Activity AA-FCS_SSH_EXT.1.8-ATE-01

The test harness needs to be configured so that its connection rekey or termination limits are greater than the limits supported by the TOE -- it is expected that the test harness should not be initiating the connection rekey or termination.

- *Test 1: Establish an SSH connection. Wait until the identified connection rekey limit is met. Observed that a connection rekey or termination is initiated. This may require traffic to periodically be sent, or connection keep alive to be set, to ensure that the connection is not closed due to an idle timeout.*
- *Test 2: Establish an SSH connection. Transmit data from the TOE until the identified connection rekey or termination limit is met. Observe that a connection rekey or termination is initiated.*
- *Test 3: Establish an SSH connection. Send data to the TOE until the identified connection rekey limit or termination is met. Observe that a connection rekey or termination is initiated.*

Summary

The evaluator tested the data transfer in both directions and observed that the connection rekeying is performed.

2.1.2.10 SSH Protocol - Client (FCS_SSHC_EXT.1)

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

Assurance Activity AA-FCS_SSHC_EXT.1-AGD-01

The evaluator shall check the guidance documentation to ensure that it contains instructions to the administrator on how to ensure that only the allowed mechanisms are used in SSH connections with the TOE.

Summary

The available cryptographic algorithms are described in [ECG] section 3.21.6 "Cryptographic key handling", where the supported algorithms are listed. The basic conforming SSH configuration including the initial key generation is performed by the automated installation process.

Please refer to AA-FCS_SSH_EXT.1.2-AGD-01 for an assessment of the relevant instructions.

Test Assurance Activities

Assurance Activity AA-FCS_SSHC_EXT.1-ATE-01

The evaluator shall perform the following tests:

- *Test 1: [conditional] If using a local database by associating each host name with its corresponding public key, the evaluator shall configure the TOE with only a single host name and corresponding public key in the local database. The evaluator shall verify that the TOE can successfully connect to the host identified by the host name.*
- *Test 2: [conditional] If using a local database by associating each host name with its corresponding public key, the evaluator shall configure the TOE with only a single host name and non-corresponding public key in the local database. The evaluator shall verify that the TOE fails to connect to a host not identified by the host name.*
- *Test 3: [conditional] If using a local database by associating each host name with its corresponding public key, the evaluator shall try to connect to a host not configured in the local database. The evaluator shall verify that the TOE either fails to connect to a host identified by the host name or there is a prompt provided to store the public key in the local database.*
- *Test 4: [conditional] If using a list of trusted certification authorities, the evaluator shall configure the TOE with only a single trusted certification authority corresponding to the host. The evaluator shall verify that the TOE can successfully connect to the host identified by the host name.*
- *Test 5: [conditional] If using a list of trusted certification authorities, the evaluator shall configure the TOE with only a single trusted certification authority that does not correspond to the host. The evaluator shall verify that the TOE fails to the host identified by the host name.*

Summary

With having a local database of trusted keys, the evaluator tested the case where the database is empty and observed that a prompt for adding the server is provided, followed by a successful connection in case of acceptance.

2.1.2.11 SSH Protocol - Server (FCS_SSHS_EXT.1)

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

Assurance Activity AA-FCS_SSHS_EXT.1-AGD-01

The evaluator shall check the guidance documentation to ensure that it contains instructions to the administrator on how to ensure that only the allowed mechanisms are used in SSH connections with the TOE.

Summary

The available cryptographic algorithms are described in [ECG] section 3.21.6 "Cryptographic key handling", where the supported algorithms are listed. The basic conforming SSH configuration including the initial key generation is performed by the automated installation process.

Please refer to AA-FCS_SSH_EXT.1.2-AGD-01 for an assessment of the relevant instructions.

Test Assurance Activities

Assurance Activity AA-FCS_SSHS_EXT.1-ATE-01

The evaluator shall repeat Test 1 and Test 2 from FCS_SSH_EXT.1.4 for each of the authentication mechanisms supported by the TOE.

Next the evaluator shall configure the remote peer to only allow an authentication mechanism that is not included in the ST selection. The evaluator shall attempt to connect to the TOE and observe that the attempt fails.

Summary

The evaluator tested the SSH server by establishing successful connection with each claimed authentication mechanism and verified the termination of the connection if the session ends. He when verified that the connection was rejected if only unallowed authentication mechanisms were advertized by the client.

2.1.2.12 Storage of Sensitive Data (FCS_STO_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FCS_STO_EXT.1-ASE-01

The evaluator will check the TSS to ensure that it lists all persistent sensitive data for which the OS provides a storage capability. For each of these items, the evaluator will confirm that the TSS lists for what purpose it can be used, and how it is stored. The evaluator will confirm that cryptographic operations used to protect the data occur as specified in FCS_COP.1(1).

The evaluator will also consult the developer documentation to verify that an interface exists for applications to securely store credentials.

Summary

Section 7.2.2.9 FCS_STO_EXT.1 Storage of Sensitive Data states that all partitions can be encrypted. dm-crypt is used for the encryption of the disk partitions. As described in 7.2.2 AES-XTS is used for the disk encryption based on dm-crypt therefore the cryptographic operation stated in FCS_COP.1(1) is used.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

No assurance activities defined.

2.1.2.13 TLS Client Protocol (FCS_TLSC_EXT.1)

FCS_TLSC_EXT.1.1

TSS Assurance Activities

Assurance Activity AA-FCS_TLSC_EXT.1.1-ASE-01

The evaluator will check the description of the implementation of this protocol in the TSS to ensure that the cipher suites supported are specified. The evaluator will check the TSS to ensure that the cipher suites specified include those listed for this component.

Summary

The evaluator determined that section 7.2.2.10 FCS_TLSC_EXT.1 TLS Client Protocol claims to support all cipher suites listed.

Guidance Assurance Activities

Assurance Activity AA-FCS_TLSC_EXT.1.1-AGD-01

The evaluator will also check the operational guidance to ensure that it contains instructions on configuring the OS so that TLS conforms to the description in the TSS.

Summary

The evaluator examined the following section of [ECG] for related guidance to FCS_TLSC_EXT.1:

- Section 3.21.6 "Cryptographic key handling" lists the possible cipher suites to be used with TLS.

Test Assurance Activities

Assurance Activity AA-FCS_TLSC_EXT.1.1-ATE-01

The evaluator will also perform the following tests:

- ***Test 1:*** *The evaluator will 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 will attempt to establish the connection using a server with a server certificate that contains the Server Authentication purpose in the extendedKeyUsage field and verify that a connection is not established. The evaluator will then verify that the client rejects an otherwise valid server certificate that lacks the Server Authentication purpose in the extendedKeyUsage field and a connection is not established. Ideally, the two certificates should be identical except for the extendedKeyUsage field.*
- ***Test 3:*** *The evaluator will 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 will verify that the OS disconnects after receiving the server's Certificate handshake message.*
- ***Test 4:*** *The evaluator will 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 will perform the following modifications to the traffic:*
  - ***Test 5.1:*** *Change the TLS version selected by the server in the Server Hello to a non-supported TLS version (for example 1.3 represented by the two bytes 03 04) and verify that the client rejects the connection.*
  - ***Test 5.2*** *Modify at least one byte in the server's nonce in the Server Hello handshake message, and verify that the client rejects the Server Key Exchange handshake message (if using a DHE or ECDHE cipher suite) or that the server denies the client's Finished handshake message.*
  - ***Test 5.3:*** *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 will verify that the client rejects the connection after receiving the Server Hello*
  - ***Test 5.4:*** *If an ECDHE or DHE ciphersuite is selected, modify the signature block in the Server's Key Exchange handshake message, and verify that the client rejects the connection after receiving the Server Key Exchange message.*
  - ***Test 5.5:*** *Modify a byte in the Server Finished handshake message, and verify that the client sends a fatal alert upon receipt and does not send any application data.*
  - ***Test 5.6:*** *Send a garbled message from the Server after the Server has issued the Change Cipher Spec message and verify that the client denies the connection.*

Summary

Most of the following tests are implemented using a modified TLS servers implementations. In the remaining cases a standard TLS server instance is used.

Test 1: The evaluator generated all necessary certificates, keys, CRLS and chains for this test and connected the TOE to an OpenSSL TLS server configured to only accept the ciphersuites mandated by the Security Target and made sure that the connection was successful.

Test 2: The evaluator generated all necessary certificates, keys, CRLS and chains for this test and connected the TOE to an OpenSSL TLS server that contains the Server Authentication purpose in the extendedKeyUsage field and verified that a connection was established. For the second part of this test, a server certificate has been generated without the Server Authentication purpose in the extendedKeyUsage field. The evaluator made sure that both certificates are identical except for the serverAuth value in the extendedKeyUsage field. The connection was not established.

Test 3: The evaluator generated all necessary certificates, keys, CRLS and chains for this test. The evaluator used an ECDSA certificate while forcing the server to use the TLS_RSA_WITH_AES_128_CBC_SHA ciphersuite. The evaluator verified that the OS disconnects after receiving the server's Certificate handshake message.

Test 4: The evaluator generated all necessary certificates, keys, CRLS and chains for this test. The evaluator used a proxy in order to force the server to use the TLS_NULL_WITH_NULL_NULL cipher suite and verified. that the client denies the connection

Test 5.1: The evaluator generated all necessary certificates, keys, CRLS and chains for this test. The evaluator used a proxy to modify the TLS version during the handshake in the server hello message and verified that the client rejects the connection.

Test 5.2: The evaluator generated all necessary certificates, keys, CRLS and chains for this test. The evaluator used a proxy to modify a byte in the nonce in the server hello message and verifier that the client rejects the server key exchange message.

Test 5.3: The evaluator generated all necessary certificates, keys, CRLS and chains for this test. The evaluator used a proxy to 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 verified that the client rejected the connection after receiving the Server Hello.

Test 5.4: The evaluator generated all necessary certificates, keys, CRLS and chains for this test. The evaluator used a proxy to modify the signature block in the server certificate in the key exchange handshake message, and verified that the client rejects the connection after receiving and verifying the certificate.

Test 5.5: The evaluator generated all necessary certificates, keys, CRLS and chains for this test. The evaluator used a proxy to modify a byte in the server finished message and verified that the client send a fatal alert upon receipt and did not send any application data.

Test 5.6: The evaluator generated all necessary certificates, keys, CRLS and chains for this test. The evaluator used a proxy to send a garble message from the server after the server has issued the change cipher spec message, and verified that the client denied the connection.

FCS_TLSC_EXT.1.2

TSS Assurance Activities

Assurance Activity AA-FCS_TLSC_EXT.1.2-ASE-01

The evaluator will 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 will ensure that this description identifies whether and the manner in which certificate pinning is supported or used by the OS.

Summary

The evaluator determined that 7.2.2.10 FCS_TLSC_EXT.1 TLS Client Protocol states:

> *The TOE verifies that server certificate is valid according to FIA_X509_EXT.1 and that the presented identifier matches the reference identifier according to RFC 6125. The reference identifiers supported are DNS and IP addresses in the SAN. Wildcards are supported. The TOE does not support certificate pinning. The TOE does not establish a trusted channel if the server certificate is invalid.*

Guidance Assurance Activities

Assurance Activity AA-FCS_TLSC_EXT.1.2-AGD-01

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

Summary

According to [ST], section 7.2.2 "Cryptography", TLS is implemented by OpenSSL. That is, a library to be instrumented by some application. Setting the reference identifier to be used for TLS certificate validation can only be performed by the consuming application since the library itself has no configuration file.

Test Assurance Activities

Assurance Activity AA-FCS_TLSC_EXT.1.2-ATE-01

The evaluator will configure the reference identifier according to the AGD guidance and perform the following tests during a TLS connection:

- ***Test 1:*** *The evaluator will present a server certificate that does not contain an identifier in either the Subject Alternative Name (SAN) or Common Name (CN) that matches the reference identifier. The evaluator will verify that the connection fails.*
- ***Test 2:*** *The evaluator will 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 will 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 will present a server certificate that contains a CN that matches the reference identifier and does not contain the SAN extension. The evaluator will 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 will 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 will verify that the connection succeeds.*
- ***Test 5:*** *The evaluator will perform the following wildcard tests with each supported type of reference identifier:*
  - ***Test 5.1:*** *The evaluator will 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:*** *The evaluator will present a server certificate containing a wildcard in the left-most label but not preceding the public suffix (e.g. *. example.com). The evaluator will configure the reference identifier with a single left-most label (e.g. foo.example.com) and verify that the connection succeeds. The evaluator will 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 will configure the reference identifier with two left-most labels (e.g. bar.foo.example.com) and verify that the connection fails.*
  - ***Test 5.3:*** *The evaluator will present a server certificate containing a wildcard in the left-most label immediately preceding the public suffix (e.g. *.com). The evaluator will configure the reference identifier with a single left-most label (e.g. foo.com) and verify that the connection fails. The evaluator will configure the reference identifier with two left-most labels (e.g. bar.foo.com) and verify that the connection fails.*
- ***Test 6:*** *[conditional] If URI or Service name reference identifiers are supported, the evaluator will configure the DNS name and the service identifier. The evaluator will 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 will 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 will present a certificate that does not match the pinned certificate and verify that the connection fails.*

Summary

Test 1: The evaluator generated all necessary certificates, keys, CRLS and chains for this test. The evaluator presented a server certificate that does not contain an identifier in either the Subject Alternative Name (SAN) or Common Name (CN) that matches the reference identifier. The evaluator verified that the connection failed.

Test 2: The evaluator generated all necessary certificates, keys, CRLS and chains for this test. The evaluator presented a server certificate that contained a CN that matched the reference identifier, contained the SAN extension, but did not contain an identifier in the SAN that matched the reference identifier. The evaluator verified that the connection failed. The evaluator repeated this test for each supported SAN type.

Test 3: The evaluator generated all necessary certificates, keys, CRLS and chains for this test. The removed the SAN extension from the certificate and kept the CN. The evaluator verified that the connection succeeded.

Test 4: The evaluator generated all necessary certificates, keys, CRLS and chains for this test. The evaluator set an incorrect CN but kept a correct SAN extension. The evaluator verified that the connection succeeded.

Test 5.1: The evaluator generated all necessary certificates, keys, CRLS and chains for this test. The evaluator generated a server certificate containing a wildcard that is not in the left-most label of the presented identifier and verified that the connection failed.

Test 5.2: The evaluator generated all necessary certificates, keys, CRLS and chains for this test. The evaluator generated a server certificate containing a wildcard in the left-most label but not preceding the public suffix.

Test 5.3: The evaluator generated all necessary certificates, keys, CRLS and chains for this test. The evaluator generated a server certificate containing a wildcard in the left-most label immediately preceding the public suffix. The evaluator configured the reference identifier with a single left-most label and verified that the connection fails. The evaluator configured the reference identifier with two left-most labels and verified that the connection failed.

Test 6: This test is not applicable. URI or Service name reference identifiers are not supported.

Test 7: This test is not applicable. Certificate pinning is not claimed.

FCS_TLSC_EXT.1.3

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-FCS_TLSC_EXT.1.3-ATE-01

The evaluator will use TLS as a function to verify that the validation rules in FIA_X509_EXT.1.1 are adhered to and shall perform the following additional test:

- ***Test 1:*** *The evaluator will demonstrate that a peer using a certificate without a valid certification path results in an authenticate failure. Using the administrative guidance, the evaluator will then load the trusted CA certificate(s) needed to validate the peer's certificate, and demonstrate that the connection succeeds. The evaluator then shall delete one of the CA certificates, and show that the connection fails.*
- ***Test 2:*** *The evaluator will demonstrate that a peer using a certificate which has been revoked results in an authentication failure.*
- ***Test 3:*** *The evaluator will demonstrate that a peer using a certificate which has passed its expiration date results in an authentication failure.*
- ***Test 4:*** *the evaluator will demonstrate that a peer using a certificate which does not have a valid identifier shall result in an authentication failure.*

Summary

Test 1: The evaluator generated all necessary certificates, keys, CRLS and chains for this test. The evaluator removed one CA in the CA chain and verified that the connection failed. The evaluator connected the TOE to an OpenSSL TLS server configured with the trusted CA certificates needed to validate the peer's certificate and verified that the connection was successful.

Test 2: The evaluator generated all necessary certificates, keys, CRLS and chains for this test. The evaluator generated a certificate for the server and then revoked it. The evaluator confirmed that the connection failed.

Test 3: The evaluator generated all necessary certificates, keys, CRLS and chains for this test. The evaluator used a certificate which passed its expiration date and verified that the connection resulted in authentication failure.

Test 4: The evaluator generated all necessary certificates, keys, CRLS and chains for this test. The evaluator used a certificate which did not have a valid identifier and verified that the connection resulted in authentication failure.

2.1.2.14 TLS Client Support for Mutual Authentication (FCS_TLSC_EXT.2)

TSS Assurance Activities

Assurance Activity AA-FCS_TLSC_EXT.2-ASE-01

The evaluator will verify that TSS describes support for the Supported Groups Extension and whether the required behavior is performed by default or may be configured.

Summary

The evaluator determined that section 7.2.2.11 FCS_TLSC_EXT.2 TLS Client Support for Supported Groups Extension states "The TOE, by default, presents the Supported Elliptic Curves Extension in the Client Hello with the following NIST curves: secp256r1, secp384r1, and secp521r1. In addition, the safe primes from RFC7919 are supported as well."

Guidance Assurance Activities

Assurance Activity AA-FCS_TLSC_EXT.2-AGD-01

If the TSS indicates that support for the Supported Groups Extension must be configured to meet the requirement, the evaluator will verify that AGD guidance includes configuration instructions for the Supported Groups Extension.

Summary

Section 6.1.2.11 "TLS Client Protocol" of the ST enlists the allowed supported groups FCS_TLSC_EXT.2.

Section 3.21.6 "Cryptographic key handling" of [ECG] describes TLS cryptographic algorithms.

The TOE implements TLS via the OpenSSL library which needs to be configured by the consuming application. The relevant section of [ECG] quotes the allowed Supported Groups Extension in the Client Hello for TLS. The evaluator verified that the stated groups are consistent with those mentioned by the ST.

Test Assurance Activities

Assurance Activity AA-FCS_TLSC_EXT.2-ATE-01

The evaluator will also perform the following test:

The evaluator will configure a server to perform ECDHE key exchange using each of the TOE's supported curves and shall verify that the TOE successfully connects to the server.

Summary

This optional requirement is not used in the ST and is therefore not applicable.

2.1.2.15 TLS Client Support for Renegotiation (FCS_TLSC_EXT.4)

TSS Assurance Activities

Assurance Activity AA-FCS_TLSC_EXT.4-ASE-01

The evaluator will ensure that the TSS description required per FIA_X509_EXT.2.1 includes the use of client-side certificates for TLS mutual authentication.

Summary

The evaluator determined that section 7.2.2.12 FCS_TLSC_EXT.4 TLS Client Support for Mutual Authentication contains a description of the TOE's use of client side certificates for mutual authentication.

Guidance Assurance Activities

Assurance Activity AA-FCS_TLSC_EXT.4-AGD-01

The evaluator will verify that the AGD guidance required per FIA_X509_EXT.2.1 includes instructions for configuring the client-side certificates for TLS mutual authentication.

Summary

Section 6.1.2.11 "TLS Client Protocol" of the ST enlists the allowed supported groups FCS_TLSC_EXT.2.

Section 3.21.6 "Cryptographic key handling" of [ECG] describes TLS cryptographic algorithms.

The TOE implements TLS via the OpenSSL library which needs to be configured by the consuming application. The relevant section of [ECG] quotes the allowed algorithms to be used with TLS / X.509 certificates. The evaluator verified that the algorithms are consistent with those mentioned by the ST.

Test Assurance Activities

Assurance Activity AA-FCS_TLSC_EXT.4-ATE-01

The evaluator will also perform the following test:

- ***Test 1:*** *The evaluator will establish a connection to a peer server that is not configured for mutual authentication (i.e. does not send Server's Certificate Request (type 13) message). The evaluator observes negotiation of a TLS channel and confirms that the TOE did not send Client's Certificate message (type 11) during handshake.*
- ***Test 2:*** *The evaluator will establish a connection to a peer server with a shared trusted root that is configured for mutual authentication (i.e. it sends Server's Certificate Request (type 13) message). The evaluator observes negotiation of a TLS channel and confirms that the TOE responds with a non-empty Client's Certificate message (type 11) and Certificate Verify (type 15) messages.*

Summary

Test 1: The evaluator establishes a connection to a peer server that is not configured for mutual authentication and verifies that that the TOE does not send the Client's Certificate message.

Test 2: The evaluator establishes a connection to a peer server that is configured for mutual authentication and verifies that that the TOE does send the Client's Certificate message with a client certificate information included.

2.1.3 User data protection (FDP)

2.1.3.1 Acess Controls for Protecting User Data (FDP_ACF_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FDP_ACF_EXT.1-ASE-01

The evaluator will confirm that the TSS comprehensively describes the access control policy enforced by the OS. The description must include the rules by which accesses to particular files and directories are determined for particular users. The evaluator will inspect the TSS to ensure that it describes the access control rules in such detail that given any possible scenario between a user and a file governed by the OS the access control decision is unambiguous.

Summary

The description of the access control policy enforced by the OS is provided in section 7.2.3.1 FDP_ACF_EXT.1 Access Controls for Protecting User Data. The evaluator determined that the rules for file and directory access are described and that the description is in such detail that the decision about access control is unambiguous.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-FDP_ACF_EXT.1-ATE-01

The evaluator will create two new standard user accounts on the system and conduct the following tests:

- ***Test 1:*** *The evaluator will authenticate to the system as the first user and create a file within that user's home directory. The evaluator will then log off the system and log in as the second user. The evaluator will then attempt to read the file created in the first user's home directory. The evaluator will ensure that the read attempt is denied.*
- ***Test 2:*** *The evaluator will authenticate to the system as the first user and create a file within that user's home directory. The evaluator will then log off the system and log in as the second user. The evaluator will then attempt to modify the file created in the first user's home directory. The evaluator will ensure that the modification is denied.*
- ***Test 3:*** *The evaluator will authenticate to the system as the first user and create a file within that user's user directory. The evaluator will then log off the system and log in as the second user. The evaluator will then attempt to delete the file created in the first user's home directory. The evaluator will ensure that the deletion is denied.*
- ***Test 4:*** *The evaluator will authenticate to the system as the first user. The evaluator will attempt to create a file in the second user's home directory. The evaluator will ensure that the creation of the file is denied.*
- ***Test 5:*** *The evaluator will authenticate to the system as the first user and attempt to modify the file created in the first user's home directory. The evaluator will ensure that the modification of the file is accepted.*
- ***Test 6:*** *The evaluator will authenticate to the system as the first user and attempt to delete the file created in the first user's directory. The evaluator will ensure that the deletion of the file is accepted.*

Summary

Test 1: The evaluator authenticated to the system as the first user and create a file within that user's home directory (user1/user1.txt). The evaluator then logged off the system and logged in as the second user. The evaluator attempted to read the file created in the first user's home directory. The evaluator verified that the read attempt is denied (Permission denied).

Test 2: The evaluator authenticated to the system as the first user and created a file within that user's home directory. The evaluator logged off the system and log in as the second user. The evaluator then attempted to modify the file created in the first user's home directory. The evaluator verified that the read attempt is denied (Permission denied).

Test 3: The evaluator authenticated to the system as the first user and created a file within that user's user directory. The evaluator then logged off the system and loged in as the second user. The evaluator attempted to delete the file created in the first user's home directory. The evaluator verified that the read attempt is denied (Permission denied).

Test 4: The evaluator authenticated to the system as the first user. The evaluator attempted to create a file in the second user's home directory. The evaluator verified that the creation of the file is denied (Permission denied).

Test 5: The evaluator authenticated to the system as the first user and attempted to modify the file created in the first user's home directory. The evaluator ensured that the modification of the file was accepted.

Test 6: The evaluator authenticated to the system as the first user and attempted to delete the file created in the first user's directory. The evaluator ensured that the deletion of the file was accepted.

2.1.4 Identification and authentication (FIA)

2.1.4.1 Authentication Failure Handling (FIA_AFL.1)

FIA_AFL.1.1

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-FIA_AFL.1.1-ATE-01

The evaluator will set an administrator-configurable threshold for failed attempts, or note the ST-specified assignment. The evaluator will then (per selection) repeatedly attempt to authenticate with an incorrect password, PIN, or certificate until the number of attempts reaches the threshold. Note that the authentication attempts and lockouts must also be logged as specified in FAU_GEN.1.

Summary

The evaluator set an threshold (deny=5 value) for failed attempts and attempted to login via SSH and local console. The evaluator verified that it was not be possible to login when the failure count reached 5. The evaluator also examined audit records to verify that appropriate account lockout audit records were generated.

FIA_AFL.1.2

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-FIA_AFL.1.2-ATE-01

- ***Test 1:*** *The evaluator will attempt to authenticate repeatedly to the system with a known bad password. Once the defined number of failed authentication attempts has been reached the evaluator will ensure that the account that was being used for testing has had the actions detailed in the assignment list above applied to it. The evaluator will ensure that an event has been logged to the security event log detailing that the account has had these actions applied.*
- ***Test 2:*** *The evaluator will attempt to authenticate repeatedly to the system with a known bad certificate. Once the defined number of failed authentication attempts has been reached the evaluator will ensure that the account that was being used for testing has had the actions detailed in the assignment list above applied to it. The evaluator will ensure that an event has been logged to the security event log detailing that the account has had these actions applied.*
- ***Test 3:*** *The evaluator will attempt to authenticate repeatedly to the system using both a bad password and a bad certificate. Once the defined number of failed authentication attempts has been reached the evaluator will ensure that the account that was being used for testing has had the actions detailed in the assignment list above applied to it. The evaluator will ensure that an event has been logged to the security event log detailing that the account has had these actions applied.*

Summary

Test 1: The evaluator set an threshold (deny=5 value) for failed attempts and attempted to login via SSH and local console. The evaluator verified that it was not be possible to login when the failure count reached 3. The evaluator also examined audit records to verify that appropriate account lockout audit records were generated.

Test 2: This test is not applicable, certificate-based authentication is not supported by the TOE.

2.1.4.2 Multiple Authentication Mechanisms (FIA_UAU.5)

FIA_UAU.5.1

TSS Assurance Activities

Assurance Activity AA-FIA_UAU.5.1-ASE-01

If user name 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 OS can interface.

Summary

As the [ST] states in section 7.2.4.2 "The TOE supports authentication based on username/password." and "The TOE supports authentication with SSH-keys." and not "user name and PIN that releases an asymmetric key". The evaluator concluded that this work unit is not applicable and therefore satisfied.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-FIA_UAU.5.1-ATE-01

*If **user name and password authentication** is selected, the evaluator will configure the OS with a known user name and password and conduct the following tests:*

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

Summary

Test 1: The evaluator performed multiple attempts to authenticate to the OS using the known user name and password. The evaluator ensured that the authentication attempts were successful.

Test 2: The evaluator performed multiple attempts to authenticate to the OS using the known user name but an incorrect password. The evaluator ensured that the authentication attempt were unsuccessful.

Assurance Activity AA-FIA_UAU.5.1-ATE-02

*[If **user name 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 OS can interface.]*

The evaluator will then conduct the following tests:

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

Summary

Test 1: The evaluator attempted to authenticate to the OS using the known user name and PIN (password) as well as SSH public key based authentication with correct credentials. The evaluator ensured that the authentication attempt is successful.

Test 2: The evaluator attempted to authenticate to the OS using the known user name and incorrect PIN (password) as well as SSH public key based authentication with incorrect credentials. The evaluator ensured that the authentication attempt is unsuccessful.

Assurance Activity AA-FIA_UAU.5.1-ATE-03

*If **X.509 certificate authentication** is selected, the evaluator will generate an X.509v3 certificate for a user with the Client Authentication Enhanced Key Usage field set. The evaluator will provision the OS for authentication with the X.509v3 certificate. The evaluator will ensure that the certificates are validated by the OS as per FIA_x509_EXT.1.1 and then conduct the following tests:*

- ***Test 1:** The evaluator will attempt to authenticate to the OS 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 OS with this certificate. The evaluator will ensure that the authentication attempt is unsuccessful.*

Summary

This test is not applicable, the TOE does not implemented such functionality.

FIA_UAU.5.2

TSS Assurance Activities

Assurance Activity AA-FIA_UAU.5.2-ASE-01

The evaluator will ensure that the TSS describes each mechanism provided to support user authentication and the rules describing how the authentication mechanism(s) provide authentication.

Summary

The authentication mechanisms (username and password und SSH keys) are described in section 7.2.4.2 FIA_UAU.5 Multiple Authentication Mechanisms (Refined).

The evaluator determined that the provided information describes how the authentication mechanisms provide authentication.

Guidance Assurance Activities

Assurance Activity AA-FIA_UAU.5.2-AGD-01

The evaluator will verify that configuration guidance for each authentication mechanism is addressed in the AGD guidance.

Summary

Test Assurance Activities

Assurance Activity AA-FIA_UAU.5.2-ATE-01

- ***Test 1:*** *For each authentication mechanism selected, the evaluator will enable that mechanism and verify that it can be used to authenticate the user at the specified authentication factor interfaces.*
- ***Test 2:*** *For each authentication mechanism rule, the evaluator will ensure that the authentication mechanism(s) behave as documented in the TSS.*

Summary

The evaluator performed multiple authentication mechanism related tests. The evaluator attempted to authenticate to the OS using the known user name and PIN (password) as well as SSH public key based authentication with correct and incorrect credentials.

2.1.4.3 X.509 Certificate Validation (FIA_X509_EXT.1)

FIA_X509_EXT.1.1

TSS Assurance Activities

Assurance Activity AA-FIA_X509_EXT.1.1-ASE-01

The evaluator will 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.

Summary

The validity checks of certificates is described in section 7.2.4.3 FIA_X509_EXT.1 X.509 Certificate Validation. This section also contains a reference to the description of the certificate path validation algorithm in [RFC5280] to which the TOE claims conformance.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-FIA_X509_EXT.1.1-ATE-01

[TD0525] The tests described must be performed in conjunction with the other certificate services evaluation activities, including the functions in FIA_X509_EXT.2.1. The tests for the extendedKeyUsage rules are performed in conjunction with the uses that require those rules. The evaluator will create a chain of at least four certificates: the node certificate to be tested, two Intermediate CAs, and the self-signed Root CA.

- *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 will demonstrate that validating an expired certificate results in the function failing.*
- *Test 3: The evaluator will test that the OS 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 shall be performed for each method. The evaluator will test revocation of the node certificate and revocation of the intermediate CA certificate (i.e. the intermediate CA certificate should be revoked by the root CA). If OCSP stapling per RFC 6066 is the only supported revocation method, testing revocation of the intermediate CA certificate is omitted. The evaluator will ensure that a valid certificate is used, and that the validation function succeeds. The evaluator then attempts the test with a certificate that has been revoked (for each method chosen in the selection) to ensure when the certificate is no longer valid that the validation function fails.*
- *Test 4: If any OCSP option is selected, the evaluator shall configure the OCSP server or use a man-in-the-middle tool to present a 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: The evaluator shall modify any byte in the first eight bytes of the certificate and demonstrate that the certificate fails to validate. (The certificate will fail to parse correctly.)*
- *Test 6: The evaluator shall modify any byte in the last byte of the certificate and demonstrate that the certificate fails to validate. (The signature on the certificate will not validate.)*
- *Test 7: The evaluator shall modify any byte in the public key of the certificate and demonstrate that the certificate fails to validate. (The signature of the certificate will not validate.)*
- *Test 8a: (Conditional on support for EC certificates as indicated in FCS_COP.1(3)). 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 8b: (Conditional on support for EC certificates as indicated in FCS_COP.1(3)). The evaluator shall replace the intermediate certificate in the certificate chain for Test 8a 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 8a, 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

Test 1: The evaluator demonstrated that validating a certificate without a valid certification path results in the function failing. The evaluator loaded a certificate or certificates as trusted CAs needed to validate the certificate to be used in the function, and demonstrated that the function succeeded. The evaluator deleted one of the certificates, and showed that the function fails. He also modified the issuing certificates by setting CA=False, omitting the CA signing bit, and setting a path length field value that is less than the actual certificate path.

Test 2: The evaluator demonstrated that validating an expired certificate resulted in the function failing.

Test 3: The evaluator tested that the OS could properly handle revoked CRL certificates. The evaluator tested revocation of the node certificate and revocation of the revocation of the intermediate CA certificate (i.e. the intermediate CA certificate should be revoked by the root CA). If OCSP stapling per RFC 6066 is the only supported revocation method, testing revocation of the intermediate CA certificate is omitted. The evaluator will ensure that a valid certificate is used, and that the validation function succeeds. The evaluator then attempts the test with a certificate that has been revoked to ensure when the certificate is no longer valid that the validation function fails.

Test 4: The evaluator configured the CA to sign a CRL with a certificate that does not have the cRLsign key usage set, and verified that validation of the CRL fails.

Test 5: The evaluator modified any byte in the first eight bytes of the certificate and verified that the certificate failed to validate.

Test 6: The evaluator modified any byte in the last byte of the certificate (the signature) and verified that the certificate failed to validate.

Test 7: The evaluator modified any byte in the public key of the certificate and verified that the certificate fails to validate.

FIA_X509_EXT.1.2

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-FIA_X509_EXT.1.2-ATE-01

The tests described must be performed in conjunction with the other certificate services evaluation activities, including the functions in FIA_X509_EXT.2.1. The evaluator will create a chain of at least four certificates: the node certificate to be tested, two Intermediate CAs, and the self-signed Root CA.

- ***Test 1:*** *The evaluator will construct a certificate path, such that the certificate of the CA issuing the OS's certificate does not contain the basicConstraints extension. The validation of the certificate path fails.*
- ***Test 2:*** *The evaluator will construct a certificate path, such that the certificate of the CA issuing the OS's certificate has the CA flag in the basicConstraints extension not set. The validation of the certificate path fails.*
- ***Test 3:*** *The evaluator will construct a certificate path, such that the certificate of the CA issuing the OS's certificate has the CA flag in the basicConstraints extension set to TRUE. The validation of the certificate path succeeds.*

Summary

Test 1: The evaluator constructed a certificate path, such that the certificate of the CA issuing the OS's certificate did not contain the basicConstraints extension. The validation of the certificate path failed.

Test 2: The evaluator constructed a certificate path, such that the certificate of the CA issuing the OS's certificate had the CA flag in the basicConstraints extension not set. The validation of the certificate path failed.

Test 3: The evaluator constructed a certificate path, such that the certificate of the CA issuing the OS's certificate had the CA flag in the basicConstraints extension set to TRUE. The validation of the certificate path succeeded.

2.1.4.4 X.509 Certificate Authentication (FIA_X509_EXT.2)

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-FIA_X509_EXT.2-ATE-01

The evaluator will acquire or develop an application that uses the OS TLS mechanism with an X.509v3 certificate. The evaluator will then run the application and ensure that the provided certificate is used to authenticate the connection.

The evaluator will repeat the activity for any other selections listed.

Summary

The evaluator performed multiple X.509v3 certificate related tests. Please refer to FIA_X509 test described above.

2.1.5 Security management (FMT)

2.1.5.1 Management of Security Functions and Behavior (FMT_MOF_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FMT_MOF_EXT.1-ASE-01

The evaluator will verify that the TSS describes those management functions that are restricted to Administrators, including how the user is prevented from performing those functions, or not able to use any interfaces that allow access to that function.

Summary

7.2.5.2 FMT_SMF_EXT.1 Specification of Management Functions describes that verification checks and permission bits are used to prevent unauthorized users to access management functions that are restricted to authorized administrators.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-FMT_MOF_EXT.1-ATE-01

- ***Test 1:*** *For each function that is indicated as restricted to the administrator, the evaluation shall perform the function as an administrator, as specified in the Operational Guidance, and determine that it has the expected effect as outlined by the Operational Guidance and the SFR. The evaluator will then perform the function (or otherwise attempt to access the function) as a non-administrator and observe that they are unable to invoke that functionality.*

Summary

For each function that is indicated as restricted to the administrator, the evaluator performed the function as an administrator and determined that it had the expected effect. The evaluator then performed the function as a non-administrator and observed that the evaluator was unable to invoke that functionality. That included audit configuration, authentication configuration, firewall, remote auditing, audit configuration, NTP, software updates, and NIC (WiFi) configuration. The evaluators tested the various ways to manage the TOE. In case of configuration files, the evaluator verified that they existed, in which case management operation could be executed by simply a modification of that file.

2.1.5.2 Extended: Specification of Management Functions (FMT_SMF_EXT.1)

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

Assurance Activity AA-FMT_SMF_EXT.1-AGD-01

The evaluator will verify that every management function captured in the ST is described in the operational guidance and that the description contains the information required to perform the management duties associated with the management function.

Summary

Section 6.1.5.2 "Specification of Management Functions" of the ST enlists the management function FMT_SMF_EXT.1.

Based on the table from the ST, the evaluator identified parts of the guidance that contain the relevant description. The results of this activity are summarized in the table below. Note that it omits management functions that have been marked as unsupported "N" for both, User and Administrator, in the corresponding table of the ST.

Table 1: Management duties supported by the TOE.

Management Function	Reference/Evaluator Comment
Enable/disable screen lock	The configuration of automatic screen locking is described in section 3.19 "Screen saver configuration" of [ECG].

Management Function	Reference/Evaluator Comment
Configure screen lock inactivity timeout	See "Enable/disable screen lock" above.
Configure local audit storage capacity	Section 4.3.5 "Storage of audit records" of [ECG] hints at the `auditd` configuration file to set parameters that influence the space consumed by audit logs. A detailed description of available configuration options is provided by https://documentation.suse.com/sles/15-SP4/html/SLES-all/cha-audit-setup.html .
Configure minimum password length	Password configuration options, including minimum number of characters, special characters, numeric characters, uppercase and lowercase characters are described in section 3.9 "Configuring password policy" of [ECG].
Configure minimum number of special characters in password	See "Configure minimum password length" above.
Configure minimum number of numeric characters in password	See "Configure minimum password length" above.
Configure minimum number of uppercase characters in password	See "Configure minimum password length" above.
Configure minimum number of lowercase characters in password	See "Configure minimum password length" above.
Configure lockout policy for unsuccessful authentication attempts through timeouts between attempts	Section 3.11.5 "Locking and unlocking of user accounts" of [ECG] describes the lockout of users by `pam_tally2` and refers to the man page `pam_tally2(8)` which contains all details.
Configure host-based firewall	Section 3.17 "Firewall configuration" of [ECG] states that `firewall-cmd` is used to control the firewall. Its configuration is described in detail by https://documentation.suse.com/sles/15-SP4/html/SLES-all/cha-security-firewall.html .
Configure name/address of audit/logging server to which to send audit/logging records	Section 4.3 of [ECG] refers to the man page `audisp-remote(8)` for information regarding remote auditing. Latter, in turn, refers to `audisp-remote.conf(5)`, which describes the configuration required for send the logs to a logging server.
Configure audit rules	Section 4.3.2 "Selecting the events to be audited" of [ECG] hints at the configuration files used to configure audit rules. More details are described at https://documentation.suse.com/sles/15-SP4/html/SLES-all/cha-audit-scenarios.html .
Configure name/address of network time server	The configuration of the NTP server during installation is described in section 3.17 "Configuring time synchronization with NTP" of [ECG], which describes that `chrony` is used to synchronize the time. At a later point in time, it may be configured as described in https://documentation.suse.com/sles/15-SP4/html/SLES-all/cha-ntp.html .
Enable/disable automatic software update	Section 3.20.2 "Automatic update" of [ECG] describes that `yast2 online_update_configuration` is used to configure automatic updates.

Management Function	Reference/Evaluator Comment
Configure WiFi interface	Section 3.10 "Network configuration" of [ECG] and contained references. https://documentation.suse.com/sles/15-SP4/html/SLES-all/cha-nm.html
Enable/disable network interface cards	See "Configure WiFi interface" above.

Test Assurance Activities

Assurance Activity AA-FMT_SMF_EXT.1-ATE-01

The evaluator will test the OS's ability to provide the management functions by configuring the operating system and testing each option selected in FMT_SFM_EXT.1.1. The evaluator is expected to test these functions in all the ways in which the ST and guidance documentation state the configuration can be managed.

Summary

For each function that is indicated as restricted to the administrator, the evaluator performed the function as an administrator and determined that it had the expected effect. The evaluator then performed the function as a non-administrator and observed that the evaluator was unable to invoke that functionality. That included audit configuration, authentication configuration, firewall, remote auditing, audit configuration, NTP, software updates, and NIC (WiFi) configuration. The evaluators tested the various ways to manage the TOE. In case of configuration files, the evaluator verified that they existed, in which case management operation could be executed by simply a modification of that file.

2.1.6 Protection of the TSF (FPT)

2.1.6.1 Access Controls (FPT_ACF_EXT.1)

FPT_ACF_EXT.1.1

TSS Assurance Activities

Assurance Activity AA-FPT_ACF_EXT.1.1-ASE-01

The evaluator will confirm that the TSS specifies the locations of kernel drivers/modules, security audit logs, shared libraries, system executables, and system configuration files. Every file does not need to be individually identified, but the system's conventions for storing and protecting such files must be specified.

Summary

The evaluator determined that the locations of kernel drivers/modules, security audit logs, shared libraries, system executables, and system configuration files is provided in section 7.1.20 as follows.

- Kernel / initial RAM disk / boot loader configuration: /boot
- Kernel modules: /lib/modules
- Shared libraries: /usr/lib, /usr/lib64, /lib, /lib64
- Applications: /bin, /sbin, /usr/bin, /usr/sbin
- System configuration data: /etc
- Security audit logs: /var/log/audit

- TSF-data: /var except /var/tmp which is a link to /tmp
- System-wide credentials repositories: /etc/group (world-readable), /etc/passwd (world-readable), /etc/gshadow, /etc/shadow, /etc/security/opasswd

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-FPT_ACF_EXT.1.1-ATE-01

The evaluator will create an unprivileged user account. Using this account, the evaluator will ensure that the following tests result in a negative outcome (i.e., the action results in the OS denying the evaluator permission to complete the action):

- ***Test 1:*** *The evaluator will attempt to modify all kernel drivers and modules.*
- ***Test 2:*** *The evaluator will attempt to modify all security audit logs generated by the logging subsystem.*
- ***Test 3:*** *The evaluator will attempt to modify all shared libraries that are used throughout the system.*
- ***Test 4:*** *The evaluator will attempt to modify all system executables.*
- ***Test 5:*** *The evaluator will attempt to modify all system configuration files.*
- ***Test 6:*** *The evaluator will attempt to modify any additional components selected.*

Summary

Test 1: The evaluator attempted to modify all kernel drivers and modules (/lib/*). The OS denied permission to complete the action.

Test 2: The evaluator attempted to modify all security audit logs generated by the logging subsystem (e.g. /var/log/audit/*). The OS denied permission to complete the action.

Test 3: The evaluator attempted to modify all shared libraries that were used throughout the system (e.g. /lib/* , /lib64/*, /usr/lib/*, /usr/lib64/*). The OS denied permission to complete the action.

Test 4: The evaluator attempted to modify all system executables (e.g. /bin/*, /usr/bin/*). The OS denied permission to complete the action.

Test 5:The evaluator attempted to modify all system configuration files (e.g. /etc/*,). The OS denied permission to complete the action.

Test 6: The evaluator attempted to modify any additional components selected (e.g. /boot/*). The OS denied permission to complete the action.

In some cases, files in /etc seemed to be modifiable by non-root. However, they are actually symlinks to other files that are either not writable, or they point to devices (e.g. /dev/random and /dev/null) where write access is ok.

FPT_ACF_EXT.1.2

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-FPT_ACF_EXT.1.2-ATE-01

The evaluator will create an unprivileged user account. Using this account, the evaluator will ensure that the following tests result in a negative outcome (i.e., the action results in the OS denying the evaluator permission to complete the action):

- ***Test 1:** The evaluator will attempt to read security audit logs generated by the auditing subsystem*
- ***Test 2:** The evaluator will attempt to read system-wide credential repositories*
- ***Test 3:** The evaluator will attempt to read any other object specified in the assignment*

Summary

Test 1: Test 1: The evaluator attempted to read security audit logs generated by the auditing subsystem (/var/log/audit/). The OS denied permission to complete the action.

Test 2: The evaluator attempted to read system-wide credential repositories (/etc/shadow, /etc/security/opasswd). The OS denied permission to complete the action.

Test 3: The evaluator attempted to read any other object specified in the assignment. The OS denied permission to complete the action.

2.1.6.2 Address Space Layout Randomization (FPT_ASLR_EXT.1)

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-FPT_ASLR_EXT.1-ATE-01

The evaluator will select 3 executables included with the TSF. If the TSF includes a web browser it must be selected. If the TSF includes a mail client it must be selected. For each of these apps, the evaluator will launch the same executables on two separate instances of the OS on identical hardware and compare all memory mapping locations. The evaluator will ensure that no memory mappings are placed in the same location. If the rare chance occurs that two mappings are the same for a single executable and not the same for the other two, the evaluator will repeat the test with that executable to verify that in the second test the mappings are different. This test can also be completed on the same hardware and rebooting between application launches.

Summary

The evaluator verified that whether no memory mappings are placed in the same location for different applications. The evaluator tested each of the programs cat, perl, and awk twice and compared their memory mappings.

2.1.6.3 Stack Buffer Overflow Protection (FPT_SBOP_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FPT_SBOP_EXT.1-ASE-01

For stack-based OSes, the evaluator will determine that the TSS contains a description of stack-based buffer overflow protections used by the OS. These are referred to by a variety of terms, such as stack cookie, stack guard, and stack canaries. The TSS must include a rationale for any binaries that are not protected in this manner.

Summary

The evaluator determined that the TOE implements stack canaries for each executable and its libraries (section 7.2.6.3). There TSS lists exemptions of binaries for the TOE's different hardware platforms.

Assurance Activity AA-FPT_SBOP_EXT.1-ASE-02

For OSes that store parameters/variables separately from control flow values, the evaluator will verify that the TSS describes what data structures control values, parameters, and variables are stored. The evaluator will also ensure that the TSS includes a description of the safeguards that ensure parameters and variables do not intermix with control flow values.

Summary

The [ST] selects "employ stack-based buffer overflow protections" and does not store parameters/variables separately from flow values. Based on the previous elaboration this work unit is considered to be not applicable and therefore satisfied.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-FPT_SBOP_EXT.1-ATE-01

The evaluator will also preform the following test:

- ***Test 1:** The evaluator will inventory the kernel, libraries, and application binaries to determine those that do not implement stack-based buffer overflow protections. This list should match up with the list provided in the TSS.*

Summary

The evaluator analyzed the kernel, libraries, and application binaries to determine those that did not implement stack-based buffer overflow protections. The evaluator checked the binary flags to identify whether binaries had protection. All setuid programs had the stack protection enabled. All programs not having the stack protection were mentioned in the ST.

2.1.6.4 Boot Integrity (FPT_TST_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FPT_TST_EXT.1-ASE-01

The evaluator will verify that the TSS section of the ST includes a comprehensive description of the boot procedures, including a description of the entire bootchain, for the TSF. The evaluator will ensure that the OS cryptographically verifies each piece of software it loads in the bootchain to include bootloaders and the kernel. Software loaded for execution directly by the platform (e.g. first-stage bootloaders) is out of scope. For each additional category of executable code verified before execution, the evaluator will verify that the description in the TSS describes how that software is cryptographically verified.

The evaluator will verify that the TSS contains a description of the protection afforded to the mechanism performing the cryptographic verification.

Summary

The bootchain is described in section 7.2.8.4 of the [ST]. The integrity of the kernel binary file and the bootloader is verified by signature validation through the secure boot process for x86, IBM Z and ARM architectures.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-FPT_TST_EXT.1-ATE-01

The evaluator will also perform the following tests:

- ***Test 1:*** *The evaluator will perform actions to cause TSF software to load and observe that the integrity mechanism does not flag any executables as containing integrity errors and that the OS properly boots.*
- ***Test 2:*** *The evaluator will modify a TSF executable that is part of the bootchain verified by the TSF (i.e. Not the first-stage bootloader) and attempt to boot. The evaluator will ensure that an integrity violation is triggered and the OS does not boot (Care must be taken so that the integrity violation is determined to be the cause of the failure to load the module, and not the fact that in such a way to invalidate the structure of the module.).*
- ***Test 3: [TD0493] [conditional]*** *If the ST author indicates that the integrity verification is performed using a public key in an X509 certificate, the evaluator will verify that the boot integrity mechanism includes a certificate validation according to in FIA_X509_EXT.1 for all certificates in the chain from the certificate used for boot integrity to a certificate in the trust store that are not themselves in the trust store. This means that, for each X509 certificate in this chain that is not a trust store element, the evaluator must ensure that revocation information is available to the TOE during the bootstrap mechanism (before the TOE becomes fully operational).*

Summary

Test 1: The evaluator performed actions to cause TSF software to load and observe that the integrity mechanism did not flag any executables as containing integrity errors and that the OS properly boots.

Test 2: The evaluator modified a TSF executable that is part of the bootchain verified by the TSF (i.e. Not the first-stage bootloader) and attempted to boot. The evaluator observed that the integrity violation is triggered and the OS does not boot.

The modification for all tested platforms comprised of a change of a sequence of bytes of the corresponding kernel image. The byte sequence was chosen such that it represents a part of a string and not any object code to not trigger a different boot error.

Test 3: This test is not applicable for the TOE as the there is only one certificate with the related key in the hardware-backed trust store.

2.1.6.5 Trusted Update (FPT_TUD_EXT.1)

FPT_TUD_EXT.1.1

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-FPT_TUD_EXT.1.1-ATE-01

[TD0463] The evaluator will check for an update using procedures described in the documentation and verify that the OS provides a list of available updates. Testing this capability may require installing and temporarily placing the system into a configuration in conflict with secure configuration guidance which specifies automatic update.

The evaluator is also to ensure that the response to this query is authentic by using a digital signature scheme specified in FCS_COP.1(3). The digital signature verification may be performed as part of a network protocol as described in FTP_ITC_EXT.1. If the signature verification is not performed as part of a trusted channel, the evaluator shall send a query response with a bad signature and verify that the signature verification fails. The evaluator shall then send a query response with a good signature and verify that the signature verification is successful.

Summary

The evaluator verified the TOE trusted update operation using the zypper command, which is the package manager tool of the TOE. New or updated packages can be listed and installed. The authenticity is not performed as part of a trusted channel, but through package signatures that are tested as part of AA-FPT_TUD_EXT.1.2-ATE-01.

FPT_TUD_EXT.1.2

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-FPT_TUD_EXT.1.2-ATE-01

For the following tests, the evaluator will initiate the download of an update and capture the update prior to installation. The download could originate from the vendor's website, an enterprise-hosted update repository, or another system (e.g. network peer). All supported origins for the update must be indicated in the TSS and evaluated.

- ***Test 1:** The evaluator will ensure that the update has a digital signature belonging to the vendor prior to its installation. The evaluator will modify the downloaded update in such a way that the digital signature is no longer valid. The evaluator will then attempt to install the modified update. The evaluator will ensure that the OS does not install the modified update.*
- ***Test 2:** The evaluator will ensure that the update has a digital signature belonging to the vendor. The evaluator will then attempt to install the update (or permit installation to continue). The evaluator will ensure that the OS successfully installs the update.*

Summary

The evaluator verified the TOE trusted update operation with successful and unsuccessful output using the zypper command.

Test 1: The TOE trusted update was tested with having the correct unmodified package installed. The update was successful.

Test 2: The TOE trusted update was tested with trying to install corrupted package. The update was rejected.

2.1.6.6 Trusted Update for Application Software (FPT_TUD_EXT.2)

FPT_TUD_EXT.2.1

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-FPT_TUD_EXT.2.1-ATE-01

[TD0463] The evaluator will check for updates to application software using procedures described in the documentation and verify that the OS provides a list of available updates. Testing this capability may require temporarily placing the system into a configuration in conflict with secure configuration guidance which specifies automatic update.

The evaluator is also to ensure that the response to this query is authentic by using a digital signature scheme specified in FCS_COP.1(3). The digital signature verification may be performed as part of a network protocol occurs as described in FTP_ITC_EXT.1. If the signature verification is not performed as part of a trusted channel, the evaluator shall send a query response with a bad signature and verify that the signature verification fails. The evaluator shall then send a query response with a good signature and verify that the signature verification is successful.

Summary

The evaluator used zypper for listing and applying updates as described in the guidance documentation and verified that the OS provides a list of available updates. Please note that the test that was performed for FPT_TUD_EXT.1 applies here as well, because the update mechanisms are the same.

FPT_TUD_EXT.2.2

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-FPT_TUD_EXT.2.2-ATE-01

The evaluator will initiate an update to an application. This may vary depending on the application, but it could be through the application vendor's website, a commercial app store, or another system. All origins supported by the OS must be indicated in the TSS and evaluated. However, this only includes those mechanisms for which the OS is providing a trusted installation and update functionality. It does not include user or administrator-driven download and installation of arbitrary files.

- ***Test 1:*** *The evaluator will ensure that the update has a digital signature which chains to the OS vendor or another trusted root managed through the OS. The evaluator will modify the downloaded update in such a way that the digital signature is no longer valid. The evaluator will then attempt to install the modified update. The evaluator will ensure that the OS does not install the modified update.*
- ***Test 2:*** *The evaluator will ensure that the update has a digital signature belonging to the OS vendor or another trusted root managed through the OS. The evaluator will then attempt to install the update. The evaluator will ensure that the OS successfully installs the update.*

Summary

Test 1: The evaluator used zypper for updates as described in the guidance documentation and verified that the OS provides a list of available updates. The evaluator tested attempted to installed updates when the digital signature was no longer valid and ensured that the OS did not installed the modified update. Please note that the test that was performed for FPT_TUD_EXT.1 applies here as well, because the update mechanism are the same.

Test 2: The evaluator used zypper for updates as described in the guidance documentation and verified that the OS provides a list of available updates. The evaluator ensured that the update had a digital signature belonging to the OS vendor and verified that the OS successfully installed the update.

2.1.7 TOE access (FTA)

2.1.7.1 Default TOE Access Banners (FTA_TAB.1)

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-FTA_TAB.1-ATE-01

The evaluator will configure the OS, per instructions in the OS manual, to display the advisory warning message "TEST TEST Warning Message TEST TEST". The evaluator will then log out and confirm that the advisory message is displayed before logging in can occur.

Summary

This optional requirement is not used in the ST and is therefore not applicable.

2.1.8 Trusted path/channels (FTP)

2.1.8.1 Trusted Channel Communication (FTP_ITC_EXT.1)

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-FTP_ITC_EXT.1-ATE-01

The evaluator will configure the OS to communicate with another trusted IT product as identified in the second selection. The evaluator will monitor network traffic while the OS performs communication with each of the servers identified in the second selection. The evaluator will ensure that for each session a trusted channel was established in conformance with the protocols identified in the first selection.

Summary

The evaluator configure the OS to communicate with management server using TLS and SSH. Please see extensive testing description of TLS and SSH testing performed above.

2.1.8.2 Trusted Path (FTP_TRP.1)

TSS Assurance Activities

Assurance Activity AA-FTP_TRP.1-ASE-01

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

Summary

The evaluator determined that section 7.2.8.2 specifies that "This trusted path based on SSH is used to allow remote administrators to securely access the TOE for administration.". The protection and establishment of the SSH communication is further described in section 7.2.2.13 SSH Protocol.

The evaluator concluded that the protocols listed in the requirements (SSH) are consistent with the one (SSH) specified in section 7.2.8.2

Guidance Assurance Activities

Assurance Activity AA-FTP_TRP.1-AGD-01

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

Summary

Administrators can login to the TOE via the local serial terminal or SSH. The evaluator examined [ECG].

SSH authentication has already been discussed in the previous work unit AA-FIA_UAU.5.2-AGD-01 and relevant parts of the guidance been identified.

Section 3.12 "Using serial terminals" describes the use of serial terminals, but this is a means of local administrative access not to be discussed here.

Test Assurance Activities

Assurance Activity AA-FTP_TRP.1-ATE-01

The evaluator will also perform the following tests:

- ***Test 1:*** *The evaluator will ensure that communications using each 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 will follow the operational guidance to ensure that there is no available interface that can be used by a remote user to establish a remote administrative sessions without invoking the trusted path.*
- ***Test 3:*** *The evaluator will ensure, for each method of remote administration, the channel data is not sent in plaintext.*
- ***Test 4:*** *The evaluator will ensure, for each method of remote administration, modification of the channel data is detected by the OS.*

Summary

Test 1: Remote OS administration is offered via SSH. The evaluator tested communications using SSH remote administration method for remote users. Please refer to test cases that describe SSH.

Test 2: Remote OS administration is offered via SSH. The evaluator followed the operational guidance to ensure that there were no available interface that could be used by a remote user to establish a remote administrative sessions without invoking SSH.

Test 3: Remote OS administration is offered via SSH. The evaluator ensured that SSH data was not sent in plaintext.

Test 4: Remote OS administration is offered via SSH. The evaluator ensured that modification of the channel (SSH) data was detected by the OS.

This modification was implemented through using a modified SSH client which flipped a byte after a certain number of sent packets.

2.2 Security Assurance Requirements

2.2.1 Life-cycle support (ALC)

2.2.1.1 Labelling of the TOE (ALC_CMC.1)

Assurance Activity AA-ALC_CMC.1-ALC-01

The evaluator will 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. Further, the evaluator will check the AGD guidance and OS samples received for testing to ensure that the version number is consistent with that in the ST. If the vendor maintains a web site advertising the OS, the evaluator will examine the information on the web site to ensure that the information in the ST is sufficient to distinguish the product.

Summary

The [ST], section 1.2 identifies the TOE as "SUSE Linux Enterprise Server 15 SP4". The evaluator examined the ISO images the TOE is shipped as:

- `SLE-15-SP4-Full-x86_64-QU3-Media1.iso` (for the Intel x86_64 and the AMD64 platform)
- `SLE-15-SP4-Full-ppc64le-QU3-Media1.iso` (for the Power 10 platform)
- `SLE-15-SP4-Full-s390x-QU3-Media1.iso` (for the z/Architecture)
- `SLE-15-SP4-Full-aarch64-QU3-Media1.iso` (for the ARM architecture)

IN these ISO images, the version of the TOE is clearly represented as "**SLE-15-SP4**", where SLE is the abbreviation for "SUSE Linux Enterprise".

2.2.1.2 TOE CM coverage (ALC_CMS.1)

Assurance Activity AA-ALC_CMS.1-ALC-01

The "evaluation evidence required by the SARs" in this PP is limited to the information in the ST coupled with the guidance provided to administrators and users under the AGD requirements. By ensuring that the OS is specifically identified and that this identification is consistent in the ST and in the AGD guidance (as done in the evaluation activity for ALC_CMC.1), the evaluator implicitly confirms the information required by this component. Life-cycle support is targeted aspects of the developer's life-cycle and instructions to providers of applications for the developer's devices, rather than an in-depth examination of the TSF manufacturer's development and configuration management process. This is not meant to diminish the critical role that a developer's practices play in contributing to the overall trustworthiness of a product; rather, it's a reflection on the information to be made available for evaluation.

The evaluator will ensure that the developer has identified (in guidance documentation for application developers concerning the targeted 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(s) are invoked (e.g., compiler and linker flags). The evaluator will ensure that this documentation also includes an indication of whether such protections are on by default, or have to be specifically enabled. The evaluator will 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

The developer provided [ECG], which in chapter 5 provides guidance for "Application Developers":

> " *When creating application running on SLES the application developers can use the included gcc compiler and linker. When invoking gcc, the best practices for secure development should be followed by developers:*
>
> - *Include the enabling of stack smashing protections through the following compiler flags:*

- *`-fstack-protector-strong --param=ssp-buffer-size=4`*

- *Include the enabling of ASLR through the following compiler and linker flags:*
  - *`-fpie -Wl,-pie`*

"

The evaluator determines that the developer provides guidance to developers on how to specifically enable buffer overflow prevention for their own programs. The evaluator further determines that [ECG], titled "Common Criteria NIAP Evaluated Configuration Guide for **SUSE LINUX Enterprise Server 15 SP4**" allows readers of this guide to specifically relate that document to the TOE, which is called "**SUSE Linux Enterprise Server Version 15 SP4**" and thus the TSF related to and provided by it.

2.2.1.3 Extension: Timely Security Updates (ALC_TSU_EXT.1)

Assurance Activity AA-ALC_TSU_EXT.1-ALC-01

The evaluator will verify that the TSS contains a description of the timely security update process used by the developer to create and deploy security updates. The evaluator will verify that this description addresses the entire application. The evaluator will also verify that, in addition to the OS developer's process, any third-party processes are also addressed in the description. The evaluator will also verify that each mechanism for deployment of security updates is described.

The evaluator will verify that, for each deployment mechanism described for the update process, the TSS lists a time between public disclosure of a vulnerability and public availability of the security update to the OS patching this vulnerability, to include any third-party or carrier delays in deployment. The evaluator will verify that this time is expressed in a number or range of days.

The evaluator will verify that this description includes the publicly available mechanisms (including either an email address or website) for reporting security issues related to the OS. The evaluator shall verify that the description of this mechanism includes a method for protecting the report either using a public key for encrypting email or a trusted channel for a website.

Summary

The developer provided a description of the timely security updates in section 7.1.1 of [ST]. The description provides the following information:

"

The entire TOE (and in fact the entire SLES distribution) is subject to an extensive update process. The update process starts when SUSE is informed about defects. Depending on the severity (security incidents are considered to be severe), fixes are developed, tested and released with updated RPM packages.

Guaranteed response times depend on the selected service level agreement which is outlined in https://www.suse.com/support/handbook/.

SUSE generally does not disclose, discuss, or confirm security issues until a full investigation is complete and any necessary patches or releases are available. Further details about the security release process can be obtained at https://www.suse.com/support/security/ Once an issue has been confirmed and a patch has been made available, references containing technical details on the patches are made available and Common Vulnerabilities and Exposures (CVEs), etc. are released.

SUSE distributes information about security issues in its products through its "SUSE Update Advisory" page. (https://www.suse.com/support/update/)

The entire update process is handled by SUSE and covers all packages shipped as part of the SLES distribution from which the TOE is a subset.

Potential security vulnerabilities can be reported by following the procedures on the "SUSE Security Contacts" page (https://www.suse.com/support/security/contact/). This includes sending an email to "security@suse.com" or "security@suse.de" and includes the ability to encrypt information using the SUSE Security Team PGP key.

"

The evaluator determines, that only SUSE handles the update process. The description provided covers the deployment of the security updates ("patches").

The webpage https://www.suse.com/support/handbook/ in section "Hours of Coverage and Target Response Times" lists the response time in days for incidents of different severity classes (which are also defined in that document) as well as different support contracts.

The evaluator determines that on https://www.suse.com/support/security/contact/, where email addresses (`security@{suse.com,suse.de}`) and GPG keys for protecting the communication are provided.

2.2.2 Guidance documents (AGD)

2.2.2.1 Operational user guidance (AGD_OPE.1)

Assurance Activity AA-AGD_OPE.1-AGD-01

Some of the contents of the operational guidance are verified by the guidance evaluation activities in Section 5.1 of [OSPPv4.2.1] *and evaluation of the OS according to the* [CEM].

If cryptographic functions are provided by the OS, the operational guidance shall contain instructions for configuring the cryptographic engine associated with the evaluated configuration of the OS. It shall provide a warning to the administrator that use of other cryptographic engines was not evaluated nor tested during the CC evaluation of the OS. The documentation must describe the process for verifying updates to the OS by verifying a digital signature – this may be done by the OS or the underlying platform. The evaluator will verify that this process includes the following steps: Instructions for obtaining the update itself. This should include instructions for making the update accessible to the OS (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. The OS will likely contain security functionality that does not fall in the scope of evaluation under this PP. The operational guidance shall make it clear to an administrator which security functionality is covered by the evaluation activities.

Summary

As the evaluator noted in previous assurance activities, the TOE's OpenSSL is a cryptographic library that requires an application in order to configure it. Also, [ST] makes no explicit statements about which applications are covered by the evaluation, as those are considered outside the TOE scope. The evaluator notes that the FIPS 140-2 compliant TOE enforce that only FIPS-approved algorithms/functions are used in the evaluated configuration. According to a brief note in chapter 2 "installation" of [ECG].

As the TOE scope includes OpenSSL, the TOE has a number of functions that are, in general, outside of the evaluated configuration. [ECG] section 3.21.6 "Cryptographic key handling" counters the accidental misuse of other functions by inclusion of a number of explicit instructions to restrict the use to supported functions. In particular, it states:

- *The following cryptographic mechanisms MUST be used for SSH: [...]*
- *The following cipher suites MUST be used: [...]*
- *You MUST ensure that the exponent of an RSA certificate is at least 2^16 + 1 i.e. 65537 or larger. [...]*

- *The cryptographic key establishment MUST be implemented using one of the following cryptographic key establishment methods: [...]*
- *Asymmetric cryptographic keys MUST be generated using one of the following cryptographic key generation algorithms: [...]*

The meaning of "MUST" is explained in section 1.2 "How to use this document" of [ECG].

[ST] section 7.2.6.5 "Trusted Update" specifies the update mechanism based in the TSS. Updates are performed manually using the `zypper` tool which deals with related low-level functions such as RPM. It represents the modern SuSE standard solution for package management and updates on the command line. As a table of FMT_SMF_EXT.1.1 details, automatic updates are not supported in the evaluated configuration.

Section 3.20 "Update configuration" of [ECG] provides related guidance for trusted updates. The man pages `zypper(8)`.

Section 3.20 of [ECG] is rather brief, which the evaluator determines tolerable, since details are available in the `zypper` man page and the procedures is established standard (for SuSE products). It states the reference to `zypper` and that the trusted updates are realized by means of storage of a certificate received during installation with help of which RPM packages are verified. No specific configuration is needed for that.

As a standard procedure for Linux distributions, the RPM packages are digitally signed and verified/enforced by default (via `rpm`) unless explicitly specified with `--nosignature` which omits verification of the RPM package or header signatures when reading the RPM package. Additionally, the signatures of each installed RPM package can be viewed with `rpm -qi` which shows the key that was used for signing the package as well as the signature ciphers.

However, this verification process takes place implicitly when updates are installed with `zypper` and no further documentation is expected by the evaluator regarding this aspect.

The evaluator verified the available guidance including [ECG] and [MANPAGES] against the security functionality claimed in [ST] and determined that all aspects are covered.

2.2.2.2 Preparative procedures (AGD_PRE.1)

Assurance Activity AA-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 claimed for the TOE in the ST.

Summary

According to [ST] section 1.4.4 "Required Hardware and Software", the evaluation supports the following platforms:

- x86 64bit Intel Xeon processors:
  - Delta D20x-M1-PC-32-8-96GB-1TB-2x1G
  - Micro Architecture: Cascade Lake
- x86 64bit AMD EPYC processors:
  - Gigabyte R181-Z90
  - Micro Architecture: AMD EPYC 3rd Generation
- IBM Z System based on z/Architecture processors:

    - IBM Z System z15
    - Micro Architecture: z15
- IBM POWER processors:
    - IBM POWER 10
    - Micro Architecture: POWER 10
- System based on ARM64 processors:
    - Gigabyte R181-T90
    - Micro Architecture: ARMv8.2-A

The evaluator checked [ECG] section 1.3.1 "Hardware requirements" to determine that it states the same platforms.

There are a number of aspects with respect to which specific documentation is expected and identified in [ECG]:

- There are different installer images depending on the architecture of the platform, as described in section 2.2.2 "Obtaining of installation images" of [ECG].
- Specific configuration steps may be needed depending on the architecture of the platform, as described in sections 2.3.2 "x86 Configuration", 2.3.2 "ARM64 System Configuration", and 2.3.3 "IBM Z System Configuration" and 2.3.4 "IBM POWER System Configuration" of [ECG].
- Cryptographic implementations can differ depending on the architecture of the platform, as described in 3.21.{1,2,3} "OpenSSL on $PLATFORM Architecture" of [ECG]. These sections describe the specific configurations or settings for OpenSSL on that platform.

The evaluator considered the dependencies of the guidance on aspects of the platform and found no other details that would be necessary to mention.

2.2.3 Tests (ATE)

2.2.3.1 Independent testing - conformance (ATE_IND.1)

Assurance Activity AA-ATE_IND.1-ATE-01

The evaluator will prepare a test plan and report documenting the testing aspects of the system, including any application crashes during testing. The evaluator shall determine the root cause of any application crashes and include that information in the report. The test plan covers all of the testing actions contained in the [CEM] *and the body of this PP's Evaluation Activities*

While it is not necessary to have one test case per test listed in an evaluation activity, the evaluator 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 evaluator is 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) should be provided that the driver or tool will not adversely affect the performance of the functionality by the OS and its platform.

This also includes the configuration of the cryptographic engine to be used. The cryptographic algorithms implemented by this engine are those specified by this PP and used by the cryptographic protocols being evaluated (IPsec, TLS). 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 following is an summary of the information that is covered in the test plan:

Configuration

The TOE configuration covered the TOE version SUSE Linux Enterprise Server 15 SP4 on the following platforms (marketing name, processor, and architecture in brackets):

- Intel (Delta D20x-M1-PC-32-8-96GB-1TB-2x1G, x86_64, Cascade Lake)
- AMD (Gigabyte R181-Z90, x86_64, AMD EPYC 3rd Generation)
- ARM (Gigabyte R181-T90, aarch64, ARMv8.2-A)
- IBM Power (IBM Power10 9080-HEX, IBM Power, Power 10)
- IBM Z (IBM Z System z15, IBM System Z, z15)

All tests were performed on all platforms.

The detailed configuration followed the setup requirements in the evaluated configuration guide [ECG], notably:

- the RPM package versions that have to be installed in addition to the base installation (section 2.3)
- cipher configurations and SSH rekeying configurations (section 3.21.5)
- platform-specific configurations (section 2.3.1 to 2.3.4)

General test structure

All tests are specified, or references to test scripts are given. The test information has an introductory paragraph that refers to the ST SFR under test. The test plan also contains a separate section for each test, where the test result is documented. Chapter 2 summaries the tests results.

2.2.4 Vulnerability assessment (AVA)

2.2.4.1 Vulnerability survey (AVA_VAN.1)

Assurance Activity AA-AVA_VAN.1-AVA-01

The evaluator will 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 find vulnerabilities that have been found in similar applications with a particular focus on network protocols the application uses and document formats it parses. 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. 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 performed a search in publicly available information for potential vulnerabilities:

- CVE: https://cve.mitre.org/cve and https://nvd.nist.gov/vuln. These have been used for searching more product and version-specific weaknesses
- Google: the standard search engine has been used for general weaknesses of the involved technologies, but also for followup searches on CVE entries, where a CVE entry itself did not provide enough information to come to a conclusion on the applicability and exploitability of a vulnerability
- SUSE support page: https://www.suse.com/support/security lists secure configuration, and vulnerability information and how to respond to it

The evaluator used the following search terms:

- Linux
- SLES
- OpenSSL
- OpenSSH
- names of the rpm packages in the installed system (always searched in combination with SLES and/or Linux to avoid false positives)

In summary, no exploitable vulnerabilities have been identified.

A Appendixes

A.1 References

AIS14 **Anforderungen an Aufbau und Inhalt von Einzelprüfberichten für Evaluationen nach CC**
Version 7
Date 2010-08-03
Location https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Zertifizierung/Interpretationen/AIS_14_pdf.pdf?__blob=publicationFile

AIS19 **Anforderungen an Aufbau und Inhalt der Zusammenfassung des ETR (Evaluation Technical Report) für Evaluationen nach CC (Common Criteria)**
Version 9
Date 2014-11-03
Location https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Zertifizierung/Interpretationen/AIS_19_pdf.pdf?__blob=publicationFile

AIS20 **Funktionalitätsklassen und Evaluationsmethodologie für deterministische Zufallszahlengeneratoren (Functionality Classes and Evaluation Methodology for Deterministic RNGs)**
Version 3
Date 2013-05-15
Location https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Zertifizierung/Interpretationen/AIS_20_pdf.pdf?__blob=publicationFile

AIS23 **Zusammentragen von Nachweisen der Entwickler (Collection of Developer Evidence)**
Version 4
Date 2017-03-15
Location https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Zertifizierung/Interpretationen/AIS_23_pdf.pdf?__blob=publicationFile

AIS32 **CC-Interpretationen im deutschen Zertifizierungsschema**
Version 7
Date 2011-06-08
Location https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Zertifizierung/Interpretationen/AIS_32_pdf.pdf?__blob=publicationFile

BSIAAA **Anforderungen an Antragsteller zur Anerkennung als Prüfstelle im Bereich Common Criteria, CC-Prüfstellen**
Version 1.5
Date 2023-06-01
Location https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Zertifizierung/CC-Pruefstellen.pdf?__blob=publicationFile&v=4

BSIRNG **Documentation and Analysis of the Linux Random Number Generator**
Version 5.4
Date received 2023-03-14
File name ear/LinuxRNG-Analysis-v5.4.pdf

CC **Common Criteria for Information Technology Security Evaluation**
Version 3.1R5
Date April 2017
Location http://www.commoncriteriaportal.org/files/ccfiles/CCPART1V3.1R5.pdf
Location http://www.commoncriteriaportal.org/files/ccfiles/CCPART2V3.1R5.pdf
Location http://www.commoncriteriaportal.org/files/ccfiles/CCPART3V3.1R5.pdf

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

CC-EP **Programm [CC-Prüfstellen]: CC-Evaluierungsprozess**
Version 1.3
Date 2023-07-01
Location https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Zertifizierung/Programm_CC-Evaluierungsprozess.pdf?__blob=publicationFile&v=2

CCEVS-EDAC **Entropy Documentation and Assessment Clarification Annex**
Author(s) NVLAP
Version 1.0
Date 2015-05-08
Location https://www.niap-ccevs.org/Documents_and_Guidance/ccevs/Entropy%20Documentation%20and%20Assessment%20Clarification.pdf

CCEVS-TD0365 **FCS_CKM_EXT.4 selections**
Date 2018-10-12
Location https://www.niap-ccevs.org/Documents_and_Guidance/view_td.cfm?TD=0365

CCEVS-TD0386 **Platform-Provided Verification of Update**
Date 2019-02-07
Location https://www.niap-ccevs.org/Documents_and_Guidance/view_td.cfm?TD=0386

CCEVS-TD0441 **Updated TLS Ciphersuites for OS PP**
Date 2019-08-21
Location https://www.niap-ccevs.org/Documents_and_Guidance/view_td.cfm?TD=0441

CCEVS-TD0463 **Clarification for FPT_TUD_EXT**
Date 2019-11-12
Location https://www.niap-ccevs.org/Documents_and_Guidance/view_td.cfm?TD=0463

CCEVS-TD0493 **X.509v3 certificates when using digital signatures for Boot Integrity**
Date 2020-03-04
Location https://www.niap-ccevs.org/Documents_and_Guidance/view_td.cfm?TD=0493

CCEVS-TD0496 **GPOS PP adds allow-with statement for VPN Client V2.1**
Date 2020-01-29
Location https://www.niap-ccevs.org/Documents_and_Guidance/view_td.cfm?TD=0496

CCEVS-TD0501 **Cryptographic selections and updates for OS PP**
Date 2020-09-03
Location https://www.niap-ccevs.org/Documents_and_Guidance/view_td.cfm?TD=0501

CCEVS-TD0525 **Updates to Certificate Revocation (FIA_X509_EXT.1)**
Date 2020-07-01
Location https://www.niap-ccevs.org/Documents_and_Guidance/view_td.cfm?TD=0525

CCEVS-TD0578 **SHA-1 is no longer mandatory**
Date 2021-02-12
Location https://www.niap-ccevs.org/Documents_and_Guidance/view_td.cfm?TD=0578

CCEVS-TD0630 **FCS_COP.1 requirements for Secure Shell**
Date 2022-06-17
Location https://www.niap-ccevs.org/Documents_and_Guidance/view_td.cfm?TD=0630

CCEVS-TD0649 **Conformance claims for OS PP v4.2.1**
Date 2022-06-17
Location https://www.niap-ccevs.org/Documents_and_Guidance/view_td.cfm?TD=0649

CCEVS-TD0666 **Ambiguous intent of FCS_SSHS_EXT.1 tests**
Date 2022-09-13
Location https://www.niap-ccevs.org/Documents_and_Guidance/view_td.cfm?TD=0666

CCEVS-TD0682 **Addressing Ambiguity in FCS_SSHS_EXT.1 Tests**
Date 2022-12-13
Location https://www.niap-ccevs.org/Documents_and_Guidance/view_td.cfm?TD=0682

CCEVS-TD0694 **FCS_SSH_EXT.1.3 Inconsistency**
Date 2022-12-14
Location https://www.niap-ccevs.org/Documents_and_Guidance/view_td.cfm?TD=0694

CCEVS-TD0695 **Choice of 128 or 256 bit size in AES-CTR in SSH Functional Package.**
Date 2022-12-14
Location https://www.niap-ccevs.org/Documents_and_Guidance/view_td.cfm?TD=0695

CCEVS-TD0715 **Updates to FIA_X509_EXT.1 for Exception Processing and Test Conditions**
Date 2023-04-04
Location https://www.niap-ccevs.org/Documents_and_Guidance/view_td.cfm?TD=0715

CCEVS-TD0777 **Clarification to Selections for Auditable Events for FCS_SSH_EXT.1**
Date 2023-08-23
Location https://www.niap-ccevs.org/Documents_and_Guidance/view_td.cfm?TD=0777

CEM **Common Methodology for Information Technology Security Evaluation**
Version 3.1R5
Date April 2017
Location http://www.commoncriteriaportal.org/files/ccfiles/CEMV3.1R5.pdf

EAR **ETR-Part Deterministic Random Number Generator and Entropy Source Analysis for the SUSE Linux Enterprise Server (SLES 15 SP4)**
Version 1.2
Date received 2023-11-21
File name ear/BSI-DSZ-CC-1213-DRNG-EAR_231121_v1.2.pdf

ECG **Common Criteria Evaluated Configuration Guide for SUSE LINUX Enterprise Server 15 SP4 (NIAP)**
Version 4.0
Date 2023-11-10
File name agd/SLES15-SP4-NIAP-Configuration-Guide-v4.0.pdf

GER **Guidelines for Evaluation Reports according to Common Criteria version 3.1**
Version 2.0
Date 2010-07-01

JIL01 **Joint Interpretation Library: Collection of Developer Evidence**
Version 1.5
Date January 2012

MANPAGES **SLES Manual Pages**
Date received 2023-07-26
File name adv/fsp/man-pages-2023-07-26.zip

OSPPv4.2.1 **Protection Profile for General Purpose Operating Systems Version 4.2.1**
Version 4.2.1
Date 2019-04-22
Location https://www.niap-ccevs.org/MMO/PP/PP_OS_V4.2.1.pdf

RFC5280 **Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile**
Author(s) D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, W. Polk
Date 2008-05-01
Location http://www.ietf.org/rfc/rfc5280.txt

SSHPKGv1.0	**Functional Package for Secure Shell (SSH)**	
	Version	1.0
	Date	2021-05-13
	Location	https://www.niap-ccevs.org/MMO/PP/pkg_ssh_v1.0.pdf
ST	**SUSE Linux Enterprise Server 15 SP4 Security Target**	
	Version	1.4
	Date	2023-12-05
	File name	ase/st-sles-15sp4-v1.4.pdf
VB-Prod	**Verfahrensbeschreibung zur Zertifizierung von Produkten (VB-Produkte)**	
	Version	4.0
	Date	2023-06-01
	Location	https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Zertifizierung/VB-Produkte.pdf?__blob=publicationFile&v=2

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.