F5 BIG-IP® 16.1.3.1 including APM Assurance Activity Report

Version:	1.0
Date:	2023-01-27
Status:	RELEASED
Classification:	Public
Filename:	CSEC2021012_AAR_230127_v1.0
Product:	F5 BIG-IP® 16.1.3.1 including APM
Sponsor:	F5, Inc.
Evaluation Facility:	atsec information security AB
Certification ID:	CSEC2021012
Certification Body:	CSEC
Author(s):	King Ables
Quality Assurance:	Trang Huynh

atsec information security AB
Svärdvägen 23
S-182 33 Danderyd

Phone: +46-8-55 110 400
www.atsec.com

(Evaluation facility with accreditation number 1937 is accredited by SWEDAC as a Testing laboratory according to ISO/IEC 17025)

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	2023-01-20	King Ables	First version.	

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]; and
- "Supporting Document - Evaluation Activities for Network Device cPP" [NDcPPv2.2-SD]

, as specified in the Security Target [ST].

The following scheme documents and interpretations have been considered:

- [CCEVS-TD0527]: "Updates to Certificate Revocation Testing (FIA_X509_EXT.1)", version as of 2020-07-01.
- [CCEVS-TD0536]: "NIT Technical Decision for Update Verification Inconsistency", version as of 2020-07-13.
- [CCEVS-TD0537]: "NIT Technical Decision for Incorrect reference to FCS_TLSC_EXT.2.3", version as of 2020-07-13.
- [CCEVS-TD0538]: "NIT Technical Decision for Outdated link to allowed-with list", version as of 2020-07-13.
- [CCEVS-TD0547]: "NIT Technical Decision for Clarification on developer disclosure of AVA_VAN", version as of 2020-10-15.
- [CCEVS-TD0555]: "NIT Technical Decision for RFC Reference incorrect in TLSS Test", version as of 2020-11-06.
- [CCEVS-TD0556]: "NIT Technical Decision for RFC 5077 question", version as of 2020-11-06.
- [CCEVS-TD0563]: "NiT Technical Decision for Clarification of audit date information", version as of 2021-01-28.
- [CCEVS-TD0564]: "NiT Technical Decision for Vulnerability Analysis Search Criteria", version as of 2021-01-28.
- [CCEVS-TD0569]: "NIT Technical Decision for Session ID Usage Conflict in FCS_DTLSS_EXT.1.7", version as of 2021-01-28.
- [CCEVS-TD0570]: "NiT Technical Decision for Clarification about FIA_AFL.1", version as of 2021-01-29.
- [CCEVS-TD0571]: "NiT Technical Decision for Guidance on how to handle FIA_AFL.1", version as of 2021-01-29.

- [CCEVS-TD0572]: "NiT Technical Decision for Restricting FTP_ITC.1 to only IP address identifiers", version as of 2021-01-29.
- [CCEVS-TD0591]: "NIT Technical Decision for Virtual TOEs and hypervisors", version as of 2021-05-21.
- [CCEVS-TD0592]: "NIT Technical Decision for Local Storage of Audit Records", version as of 2021-05-21.
- [CCEVS-TD0631]: "NIT Technical Decision for Clarification of public key authentication for SSH Server", version as of 2022-03-21.
- [CCEVS-TD0632]: "NIT Technical Decision for Consistency with Time Data for vNDs", version as of 2022-03-21.
- [CCEVS-TD0634]: "NIT Technical Decision for Clarification required for testing IPv6", version as of 2022-03-21.
- [CCEVS-TD0635]: "NIT Technical Decision for TLS Server and Key Agreement Parameters", version as of 2022-03-21.
- [CCEVS-TD0638]: "TD0638: NIT Technical Decision for Key Pair Generation for Authentication ", version as of 2022-08-05.
- [CCEVS-TD0670]: "TD0670: NIT Technical Decision for Mutual and Non-Mutual Auth TLSC Testing", version as of 2022-09-16.
- [CSEC-EP002]: "Evaluation and Certification", version 34.0 as of 2021-10-26.
- [CSEC-EP188]: "Scheme Crypto Policy", version 12.0 as of 2021-10-26.

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.

The TOE contains a single implementation of the OpenSSL module. Separate instances of the OpenSSL module run on both the Control Plane and the Data Plane. In the CAVP certificates, the Data Plane instance is called "TMM Data Plane" where TMM is an acronym for Traffic Management Microkernel. Because the OpenSSL module runs in both a virtualized environment (i.e., Virtual Clustered Multiprocessing™ a.k.a. vCMP®) and a non-virtualized environment, two sets of CAVP certificates are required for each plane. Another set of CAVP certificates is required for the Virtualized Environment (VE) implementation.

Table 1 provides the CAVP operational environment for each TOE virtual environment.

Table 1: CAVP operational environments

Software	Platform	CPU	Virtual Environment
Big-IP 16.1.3.1	F5 i4800	Intel Broadwell D-1518	N/A
		Intel Broadwell E5-1630v4	N/A
		Intel Broadwell E5-1650v4	N/A
		Intel Broadwell E5-1660v4	N/A
		Intel Broadwell E5-2680v4	N/A
		Intel Broadwell E5-2695v4	N/A
		Intel Haswell E5-2658v3	N/A
		Intel Ivy Bridge E5-2658v2	N/A
Big-IP vCMP Hypervisor 16.1.3.1	F5 i5800	Intel Broadwell D-1518	N/A
		Intel Broadwell E5-1630v4	N/A
		Intel Broadwell E5-1650v4	N/A
		Intel Broadwell E5-1660v4	N/A
		Intel Broadwell E5-2680v4	N/A
		Intel Broadwell E5-2695v4	N/A
		Intel Haswell E5-2658v3	N/A
		Intel Ivy Bridge E5-2658v2	N/A
BIG-IP Virtual Edition (VE) 16.1.3.1	Dell PowerEdge R630	Intel Xeon E5-2660v3 (Haswell)	Hyper-V version 10.0 on Windows Server 2019
	Dell PowerEdge M630	Intel Xeon E5-2590v4 (Broadwell)	VMWare ESXi 6.5.0

Software	Platform	CPU	Virtual Environment
			KVM on Ubuntu 20.04

Table 2 provides a mapping between cryptographic algorithms specified by SFRs in [ST] and Cryptographic Algorithm Validation Program (CAVP) certificates.

Table 2: Mapping of SFRs to CAVP certificates

SFR	Algorithm and [Standard]	Options	Control Plane (OpenSSL) CAVP		Data Plane (TMM) CAVP		Virtual Edition (VE) CAVP	
			Device	vCMP®	Device	vCMP®	AES-NI & SHA SSSE3	Assembler
FCS_CKM.1	RSA KeyGen (186-4) [FIPS 186-4]	Modulo 2048, Modulo 3072	A2594	A2777				A2762
	ECDSA KeyGen (186-4) [FIPS 186-4]	P-256, P-384	A2594	A2777	A2671	A2778		A2762
	ECDSA KeyVer (186-4) [FIPS 186-4]	P-256, P-384	A2594	A2777	A2671	A2778		A2762
FCS_CKM.2	RSA KeyEstab [1] [NIST SP 800-56B]	Modulo 2048, Modulo 3072						
	ECC KeyEstab (KAS-ECC Component) [NIST SP 800-56A]	P-256, P-384	A2594	A2777	A2671	A2778		A2762
FCS_COP.1 /DataEncryption	AES **AES:**[FIPS 197]; **CBC:**[NIST SP 800-38A]; **GCM:**[NIST SP 800-38D]	Encrypt & decrypt: AES-CBC-128, AES-CBC-256, AES-GCM-128, AES-GCM-256	A2594	A2777	A2671	A2778	A2711	
FCS_COP.1 /SigGen	RSA SigGen (186-4) [FIPS 186-4]	RSASSA-PKCS1v1.5: Modulo 2048 with SHA-256, SHA-384; Modulo 3072 with SHA-256, SHA-384	A2594	A2777	A2671	A2778		A2762

[1] CAVP does not support the testing of RSA key establishment.

SFR	Algorithm and [Standard]	Options	Control Plane (OpenSSL) CAVP		Data Plane (TMM) CAVP		Virtual Edition (VE) CAVP	
			Device	vCMP®	Device	vCMP®	AES-NI & SHA SSSE3	Assembler
	RSA SigVer (186-4) [FIPS 186-4]	RSASSA-PKCS1v1.5: Modulo 2048 with SHA-1, SHA-256, SHA-384; Modulo 3072 with SHA-1, SHA-256, SHA-384	A2594	A2777	A2671	A2778		A2762
	ECDSA SigGen (186-4) [FIPS 186-4]	P-256 with SHA-256, SHA-384; P-384 with SHA-256, SHA-384	A2594	A2777	A2671	A2778		A2762
	ECDSA SigVer (186-4) [FIPS 186-4]	P-256 with SHA-256, SHA-384; P-384 with SHA-256, SHA-384	A2594	A2777	A2671	A2778		A2762
FCS_COP.1 /Hash	SHS (byte-oriented) [FIPS 180-4]	SHA-1	A2594	A2777	A2671	A2778	A2711	
		SHA-256, SHA-384	A2594	A2777	A2671	A2778		A2762
FCS_COP.1 /KeyedHash	HMAC [FIPS 198-1]	HMAC-SHA-1	A2594	A2777	A2671	A2778	A2711	
		HMAC-SHA-256, HMAC-SHA-384	A2594	A2777	A2671	A2778		A2762
FCS_RBG_EXT.1	CTR_DRBG(AES) [NIST SP 800-90A Rev. 1]	AES-256	A2594	A2777	A2671	A2778	A2711	

2.1 Security Functional Requirements

2.1.1 Security audit (FAU)

2.1.1.1 Audit Data Generation (FAU_GEN.1)

TSS Assurance Activities

Assurance Activity AA-FAU_GEN.1-ASE-01

For the administrative task of generating/import of, changing, or deleting of cryptographic keys as defined in FAU_GEN.1.1 item c), the TSS should identify what information is logged to identify the relevant key.

Summary

Chapter 7 of [ST] contains the TSS. Section 7.1 *Security Audit* describes auditing.

The evaluator examined the TSS which states that for audit records logging the administrative task of generating/importing of, changing, or deleting of cryptographic keys, the certificate key file object name is logged to identified the relevant key.

Assurance Activity AA-FAU_GEN.1-ASE-02

For distributed TOEs the evaluator shall examine the TSS to ensure that it describes which of the overall required auditable events defined in FAU_GEN.1.1 are generated and recorded by which TOE components. The evaluator shall ensure that this mapping of audit events to TOE components accounts for, and is consistent with, information provided in Table 1, as well as events in Tables 2, 4, and 5 (where applicable to the overall TOE). This includes that the evaluator shall confirm that all components defined as generating audit information for a particular SFR should also contribute to that SFR as defined in the mapping of SFRs to TOE components, and that the audit records generated by each component cover all the SFRs that it implements.

Summary

According to the Security Target [ST] , the TOE is not distributed therefore this Evaluation Activity is not applicable.

Guidance Assurance Activities

Assurance Activity AA-FAU_GEN.1-AGD-01

The evaluator shall check the guidance documentation and ensure that it provides an example of each auditable event required by FAU_GEN.1 (i.e. at least one instance of each auditable event, comprising the mandatory, optional and selection-based SFR sections as applicable, shall be provided from the actual audit record).

The evaluator shall also make a determination of the administrative actions related to TSF data related to configuration changes. The evaluator shall examine the guidance documentation and make a determination of which administrative commands, including subcommands, scripts, and configuration files, are related to the configuration (including enabling or disabling) of the mechanisms implemented in the TOE that are necessary to enforce the requirements specified in the cPP. The evaluator shall document the methodology or approach taken while determining which actions in the administrative guide are related to TSF data related to configuration changes. The evaluator may perform this activity as part of the activities associated with ensuring that the corresponding guidance documentation satisfies the requirements related to it.

Summary

Section 9.2 "Sample Event Records -- TMOS, AFM" of [ECG] contains a list of auditable events corresponding to those identified in [ST] along with a description of the audit record format. The following table identifies the audits claimed in the FAU_GEN.1 requirements of [ST] and the corresponding audit samples described in section 9.2 of [ECG] .

SFR	Auditable Events	Additional Audit Record Contents	Relevant Audit Events
FAU_GEN.1	Startup of audit	No additional information	Section 9.2.1 describes audit records demonstrating start up of the syslog and management control program daemon (mcpd) audit mechanisms.

SFR	Auditable Events	Additional Audit Record Contents	Relevant Audit Events
	Shutdown of audit	No additional information	Section 9.2.2 describes audit records demonstrating shut down of the mcpd and tmsh audit mechanisms.
	Administrative login and logout	User account/name	Section 9.2.3.1 describes audit records demonstrating administrator login and logout. The audit records contain the administrator ID (e.g., admin).
	Security related configuration changes	Change description	Section 9.2.3.3 describes audit records demonstrating security-related configurations for each user interface (tmsh, GUI, iControl, iControl REST) and mcpd logging.' The audit records provides a description of the change, for example, the audit record for GUI in section 9.2.3.3.3 shows changing the DB variable value from "log.mcdp.level" to "warning".
	Generating/ import of, changing, or deleting of cryptographic keys	Action itself, unique key name or key reference	Section 9.2.3.4 describes audit records demonstrating generating, importing, modifying, and deleting cryptographic keys.
	Resetting passwords	User account	Section 9.2.3.5 describes audit records for resetting passwords in which all audit records contain the user identifier.
	Starting and stopping services	None	Sections 9.2.3.6 and 9.2.3.7 describes audit records for starting of the config service and stopping of the big3d service.
FAU_STG_EXT.3/LocSpace	Low storage space for audit events.	No additional information	Section 9.2.4 describes audit record for warning for low audit storage space. This audit record warns that the log disk usage is at least 80%.
FCS_HTTPS_EXT.1	Failure to establish a HTTPS Session	Reason for failure	Section 9.2.5 describes 3 audit records from /var/log/audit demonstrating HTTPS session requests failed because the admin user has “nologin” specified in the BIG-IP configuration, and so login is denied for the first case. In the third case, the error message is returned from mod_auth_pam(), which means that the login authentication failed. This section also describes 3 audit records from /var/log/ltm demonstrating the following failures: • SSL Handshake failed for TCP • SSL Handshake succeeded for TCP • SSL Connection terminated for TCP

SFR	Auditable Events	Additional Audit Record Contents	Relevant Audit Events
FCS_SSHS_EXT.1	Failure to establish an SSH Session	Reason for failure	Section 9.2.6 describes 3 audit records demonstrating failed SSH session establishment with the description of the failures: the entries are coming from pam_audit(). In the first two, the SSH session is not established because user root is not allowed to log in when Appliance Mode is licensed (as it must be for the Common Criteria configuration. In the third case, the user “asdf” doesn't exist.
FCS_TLSC_EXT.1[1]-[2]	Failure to establish an TLS Session	Reason for failure	Section 9.2.7 describes audit records for failure to establish a TLS data plan session (BIG-IP as client) The reason for failure include peer certificate verification error, connection error, connection termination, and SSL handshake failed.
FCS_TLSC_EXT.2	Failure to establish an TLS Session	Reason for failure	Section 9.2.7 describes audit records for failure to establish a TLS data plan session (BIG-IP as client) The reason for failure include peer certificate verification error, connection error, connection termination, and SSL handshake failed.
FCS_TLSS_EXT.1[1]-[4]	Failure to establish an TLS Session	Reason for failure	Section 9.2.8 describes audit records demonstrating failure to establish a TLS data plane session (BIG-IP as server) with a description of failure such as the protocol version is unsupported (note that the error code is from RFC 5246).
FIA_AFL.1	Unsuccessful login attempt limits is met or exceeded.	Origin of the attempt (e.g., IP address)	See FIA_UAU_EXT.2 for description of relevant audit records.
FIA_UIA_EXT.1	All use of identification and authentication mechanism.	Origin of the attempt (e.g., IP address).	See FIA_UAU_EXT.2 for description of relevant audit records.
FIA_UAU_EXT.2	All use of identification and authentication mechanism.	Origin of the attempt (e.g., IP address).	Section 9.2.11 describes audit records demonstrating both successful and failed password-based authentication for login via the GUI, SSH, iControl, and iControl REST. All audit records contain the IP address of the origin of the attempt, for example, `httpd(mod_auth_pam): user=admin(admin) partition=[All] level=Administrator tty=/sbin/nologin host=192.168.43.146`

SFR	Auditable Events	Additional Audit Record Contents	Relevant Audit Events
FIA_X509_EXT.1/Rev	Unsuccessful attempt to validate a certificate	Reason for failure	Section 9.2.12 describes audit records for unsuccessful attempt to validate a certificate with reason for failure as unable to validate certificate with an invalid x509 file.
FMT_MOF.1/Services	Starting and stopping of services.	None.	Sections 9.2.3.6, 9.2.3.7, and 9.2.14 describe audit records demonstrating starting and stopping of services.
FMT_MOF.1/ManualUpdate	Any attempt to initiate a manual update	None.	Section 9.2.15 describes audit records demonstrating the following: • successful installation of the TOE software on 2 different volumes by the administrator "admin". • failed installation of a TOE update/installation
FMT_MTD.1/CoreData	All management activities of TSF data.	None	Section 9.2.16 describes audit records demonstrating management of TSF data including creating a certificate file and resetting the administrator password via tmsh. Also, other management activities of TSF data are described by related audit records throughout section 9.2.
FMT_MTD.1/CryptoKeys	Management of cryptographic keys	None.	Section 9.2.17 describes the audit showing management of cryptographic keys is restricted to "admin" user. Also, section 9.2.3.4 describes audit records for generating cryptographic keys, importing a key, changing a key, and deleting a key.
FMT_SMF.1	All management activities of TSF data.	None	Management activities of TSF data are described by audit records throughout section 9.2. Also, other management activities of TSF data are described by audit records such as: • Section 9.2.16 describes audit records demonstrating management of TSF data including creating a certificate file and resetting the administrator password via tmsh. • Section 9.2.17 describes the audit showing management of cryptographic keys is restricted to "admin" user. Also, section 9.2.3.4 describes audit records for generating cryptographic keys, importing a key, changing a key, and deleting a key. • Sections 9.2.3.6, 9.2.3.7, and 9.2.14 describe audit records demonstrating starting and stopping of services.

SFR	Auditable Events	Additional Audit Record Contents	Relevant Audit Events
FPT_STM_EXT.1	Discontinuous changes to time – either Administrator actuated or changed via an automated process. (Note that no continuous changes to time need to be logged. See also application note on FPT_STM_EXT.1 in the NDcPP.	For discontinuous changes to time: The old and new values for the time. Origin of the attempt to change time for success and failure (e.g., IP address).	Section 9.2.19 describes audit records demonstrating successful and failed attempts of changing time. The audit records show the old and new time values, origin of the attempt, and successful and failure of the attempt.
FPT_TUD_EXT.1	Initiation of update; result of the update attempt (success or failure).	None.	Section 9.2.18 describes audit records demonstrating the following: • successful installation of the TOE software on 2 different volumes by the administrator "admin". • failed installation of a TOE update/installation
FTA_SSL_EXT.1 (if “terminate the session” is selected)	The termination of a local session by the session locking mechanism.	No additional information	Section 9.2.20 describes audit records for inactivity timeout demonstrating a user is logged out of an tmsh session when the inactivity timeout is reached.
FTA_SSL.3	The termination of a remote session by the session locking mechanism.	No additional information	Section 9.2.21 describe an audit record for inactivity timeout demonstrating a user is logged out of a SSH session when the inactivity timeout is reached.
FTA_SSL.4	The termination of an interactive session.	No additional information	See FTA_SSL.3 for description of relevant audit records.
FTP_ITC.1	Initiation of the trusted channel	Identification of the initiator and target of failed trusted channels establishment attempt.	Section 9.2.23 describes audit records for the TLS connections for the following scenarios: • Successful connection attempt • Failed connection attempt • Terminated connection For each audit record, IP addresses of the initiator and target are provided.
	Termination of the trusted channel		See description above as well as FCS_HTTPS_EXT.1, FCS_TLSC_EXT.1[1]-[2], and FCS_TLSS_EXT.1[1]-[4] for relevant audit records.

SFR	Auditable Events	Additional Audit Record Contents	Relevant Audit Events
	Failure of the trusted channel functions		See description above as well as FCS_HTTPS_EXT.1, FCS_TLSC_EXT.1[1]-[2], and FCS_TLSS_EXT.1[1]-[4] for relevant audit records.
FTP_TRP.1	Initiation of the trusted path	Identification of the claimed user identity.	Section 9.2.24 describes audit records demonstrating the following: For TLS connections (GUI, iControl SOAP, and iControl REST): • Successful login attempt • Successful logout attempt • Failed login attempt For SSH connection: • Successful connection attempt • Failed connection attempt • Terminated connection All of the audit records contain the user identity (e.g., admin, root).
	Termination of the trusted path.		See description of the relevant audit records above.
	Failure of the trusted path functions		See description of the relevant audit records above.

Table 3: Audit record description

Test Assurance Activities

Assurance Activity AA-FAU_GEN.1-ATE-01

The evaluator shall test the TOE's ability to correctly generate audit records by having the TOE generate audit records for the events listed in the table of audit events and administrative actions listed above. This should include all instances of an event: for instance, if there are several different I&A mechanisms for a system, the FIA_UIA_EXT.1 events must be generated for each mechanism. The evaluator shall test that audit records are generated for the establishment and termination of a channel for each of the cryptographic protocols contained in the ST. If HTTPS is implemented, the test demonstrating the establishment and termination of a TLS session can be combined with the test for an HTTPS session. When verifying the test results, the evaluator shall ensure the audit records generated during testing match the format specified in the guidance documentation, and that the fields in each audit record have the proper entries.

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

Summary

The evaluator verified the installation and the configuration of the TOE according to [ECG]. The evaluator used a computer running Ubuntu Linux as a SSH client and web browser in order to have remote access to the TOE. The evaluator used the built in Web Console in the VMware Hypervisor to access the virtual serial console.

- The evaluator checked whether an audit record is generated during the start-up and the shut-down of an audit event.

- The evaluator checked whether an audit record is generated during administrator user login and logout.
- The evaluator checked whether an audit record is generated during security related configuration changes.
- The evaluator checked whether an audit record is generated during generating/import of, changing, or deleting of cryptographic keys.
- The evaluator checked whether an audit record is generated for resetting passwords.
- The evaluator checked whether an audit record is generated for starting and stopping services.
- The evaluator checked whether an audit record is generated when the log size reach its limits.
- The evaluator checked whether connection establishments are failed through HTTPS, SSH and TLS.
- The evaluator checked whether an audit record is generated during authentication for all the listed services are available (SSH, GUI, IControl, IControl Rest) remotely and locally.
- The evaluator did not observe any audit log for trying to authenticate to the TOE with a non supported public key algorithm. The evaluator considers this as an expected result because the SSH-DSA public key algorithm is considered deprecated for OpenSSH.
- The evaluator checked whether an audit record is generated during failed certificate validation (as expected).
- The evaluator performed activation or modification of the welcome banner and checked for audit records.
- The evaluator performed manual update of the TOE and checked for audit records for successful and failed attempts (as expected).
- The evaluator performed modification, deletion, generation/import of cryptographic keys and checked for audit records.
- The evaluator performed modification on time zone through NTP server and checked for audit records.
- The evaluator performed modification on inactivity time period and checked for audit records for unlocking attempts of the interactive session.
- The evaluator performed modification on inactivity time period and checked for audit records for termination of the interactive session.
- The evaluator checked whether an audit record is generated when he terminates the interactive session remotely or locally.
- The evaluator checked whether an audit record is generated during initiation, termination or failure of a trusted channel.

Assurance Activity AA-FAU_GEN.1-ATE-02

For distributed TOEs the evaluator shall perform tests on all TOE components according to the mapping of auditable events to TOE components in the Security Target. For all events involving more than one TOE component when an audit event is triggered, the evaluator has to check that the event has been audited on both sides (e.g. failure of building up a secure communication channel between the two components). This is not limited to error cases but includes also events about successful actions like successful build up/tear down of a secure communication channel between TOE components.

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

Summary

The evaluator found that the TOE is not a distributed TOE, therefore there are no specific requirements for this assurance activity.

2.1.1.2 User Identity Association (FAU_GEN.2)

TSS Assurance Activities

Assurance Activity AA-FAU_GEN.2-ASE-01

The TSS and Guidance Documentation requirements for FAU_GEN.2 are already covered by the TSS and Guidance Documentation requirements for FAU_GEN.1.

Summary

This Evaluation Activity was performed in conjunction with FAU_GEN.1.

Guidance Assurance Activities

Assurance Activity AA-FAU_GEN.2-AGD-01

The TSS and Guidance Documentation requirements for FAU_GEN.2 are already covered by the TSS and Guidance Documentation requirements for FAU_GEN.1.

Summary

This Assurance Activity was performed in conjunction with the Assurance Activity for FAU_GEN.1.

Test Assurance Activities

Assurance Activity AA-FAU_GEN.2-ATE-01

This activity should be accomplished in conjunction with the testing of FAU_GEN.1.1.

Summary

This activity was accomplished in conjunction with the testing of FAU_GEN.1.1.

Assurance Activity AA-FAU_GEN.2-ATE-02

For distributed TOEs the evaluator shall verify that where auditable events are instigated by another component, the component that records the event associates the event with the identity of the instigator. The evaluator shall perform at least one test on one component where another component instigates an auditable event. The evaluator shall verify that the event is recorded by the component as expected and the event is associated with the instigating component. It is assumed that an event instigated by another component can at least be generated for building up a secure channel between two TOE components. If for some reason (could be e.g. TSS or Guidance Documentation) the evaluator would come to the conclusion that the overall TOE does not generate any events instigated by other components, then this requirement shall be omitted.

Summary

The evaluator found that the TOE is not a distributed TOE, therefore there are no specific requirements for this assurance activity.

2.1.1.3 Protected Audit Trail Storage (FAU_STG.1)

TSS Assurance Activities

Assurance Activity AA-FAU_STG.1-ASE-01

The evaluator shall examine the TSS to ensure it describes the amount of audit data that are stored locally and how these records are protected against unauthorized modification or deletion. The evaluator shall ensure that the TSS describes the conditions that must be met for authorized deletion of audit records.

Summary

Chapter 7 of [ST] contains the TSS. Section 7.1 *Security Audit* describes auditing.

The evaluator examined the TSS and found the following information:

- The TOE supports (and the evaluated configuration mandates) logging to external syslog hosts. Audit records in transit to the remote host are protected by TLS channels.
- For the case that the remote syslog host becomes unavailable, audit records are stored locally in syslog files managed, and protected against unauthorized access, by using file permission bits in the underlying Linux host. The TOE will attempt to periodically reestablish the connection with the remote syslog host indefinitely. The TOE retries within seconds of each connection failure. The TOE implements a buffer to store audit records collected during the period of time when the remote syslog host is unavailable. If the connection is reestablished before the buffers overflow, no audit records are lost. If the connection is reestablished after the buffers overflow, audit records are lost. Locally stored audit records are also available for review through the administrative interfaces of the TOE. Only users in the Administrator role can modify or delete those records. The TOE does not support deletion of audit records by authorized users.

The TOE logs a warning if the local space for syslog files on the box exceeds a configurable maximum size. The TOE implements a local syslog file rotation scheme that numbers the locally archived syslog files. The TOE will delete the oldest syslog file once the maximum size for local syslog file space is exceeded. A cron job runs every two minutes to check the audit trail storage partition in order to accomplish this. The evaluated configuration requires allocation of 7 GB of audit storage, and a warning to be logged when 90% of the storage space are exhausted. The administrator receives the warnings when reviewing the log files as instructed the CC guidance document.

Thus, the evaluator verified that the TSS provides the necessary details regarding protection of the audit data.

Assurance Activity AA-FAU_STG.1-ASE-02

For distributed TOEs the evaluator shall examine the TSS to ensure it describes to which TOE components this SFR applies and how local storage is implemented among the different TOE components (e.g. every TOE component does its own local storage or the data is sent to another TOE component for central local storage of all audit events).

Summary

According to the Security Target [ST] , the TOE is not distributed therefore this Evaluation Activity is not applicable.

Guidance Assurance Activities

Assurance Activity AA-FAU_STG.1-AGD-01

The evaluator shall examine the guidance documentation to determine that it describes any configuration required for protection of the locally stored audit data against unauthorized modification or deletion.

Summary

The evaluator examined section 2.3.8 "Event (audit) logging" and 10 Appendix "Sample Secure Remote Syslog Configuration" of [ECG] , where the guidance related to the SFR FAU_STG.1 is provided. The evaluator determined that:

- The TOE protects the local audit trail from unauthorized modification and deletion with no action required by design. In other words, no action is required on behalf of the administrator. Such protection is provided by the TOE by default.

Test Assurance Activities

Assurance Activity AA-FAU_STG.1-ATE-01

The evaluator shall perform the following tests:

a) Test 1: The evaluator shall access the audit trail without authentication as Security Administrator (either by authentication as a non-administrative user, if supported, or without authentication at all) and attempt to modify and delete the audit records. The evaluator shall verify that these attempts fail. According to the implementation no other users than the Security Administrator might be defined and without any user authentication the user might not be able to get to the point where the attempt to access the audit trail can be executed. In that case it shall be demonstrated that access control mechanisms prevent execution up to the step that can be reached without authentication as Security Administrator.

b) Test 2: The evaluator shall access the audit trail as an authorized administrator and attempt to delete the audit records. The evaluator shall verify that these attempts succeed. The evaluator shall verify that only the records authorized for deletion are deleted.

Summary

Test 1:
The evaluator checked if modification or deletion of the audit records is possible for a user with lower privileges i.e. to a non administrative user and was found it was not possible to alter or delete (as expected).

Test 2:
[ST] states that all command shells other than tmsh are disabled. For example, bash and other user-serviceable shells are excluded. As a result no authorized audit records are available for deletion.

Assurance Activity AA-FAU_STG.1-ATE-02

For distributed TOEs the evaluator shall perform test 1 and test 2 for each component that is defined by the TSS to be covered by this SFR.

Summary

The evaluator found that the TOE is not a distributed TOE, therefore there are no specific requirements for this assurance activity.

2.1.1.4 Protected Audit Event Storage (FAU_STG_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FAU_STG_EXT.1-ASE-01

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

The evaluator shall examine the TSS to ensure it describes the amount of audit data that are stored locally; what happens when the local audit data store is full; and how these records are protected against unauthorized access.

The evaluator shall examine the TSS to ensure it describes whether the TOE is a standalone TOE that stores audit data locally or a distributed TOE that stores audit data locally on each TOE component or a distributed TOE that contains TOE components that cannot store audit data locally on themselves but need to transfer audit data to other TOE components that can store audit data locally.

Summary

Chapter 7 of [ST] contains the TSS. Section 7.1 *Security Audit* describes auditing.

The evaluator examined the TSS and found the following details:

- The TOE is standalone and stores audit data locally and transmitting audit data to external syslog hosts.
- The syslog mechanism provided by the TOE's underlying operating system is used for the creation and forwarding of audit records. Audit records are sent to both local and remote storage.
- The TOE supports TLS channels for the protection of the audit records sent from the TOE to an external audit server.
- The TOE is configured to log a warning if the local space for syslog files on the TOE system exceeds the specified maximum storage space of 7 GB.
- When audit data exceeds the maximum storage space, the TOE will overwrite the oldest records. This is done by a cron job running every two minutes to check the audit trail storage partition.
- A warning is logged when 90% of the log storage space is filled.
- The TOE will overwrite the oldest records once the maximum log size is exceeded using a local syslog file rotation which numbers the locally archived syslog files and deletes the oldest syslog files.
- When a remote syslog host becomes unavailable, audit records are stored locally in syslog files managed and protected against unauthorized access via file permissions bits in TOE's the underlying operating system.
- The TOE will attempt to periodically reestablish the connection with the remote syslog host indefinitely and retries within seconds of each connection failure. The TOE implements a buffer to store audit records collected during the period of time when the remote syslog host is unavailable. If the connection is reestablished before the buffers overflow, no audit records are lost. If the connection is reestablished after the buffers overflow, audit records are lost.
- The audit records are sent to the remote storage immediately.

The evaluator noted that the ST does not claim FAU_STG_EXT.2.

Thus, the evaluator verified that the TSS provides the necessary details regarding protection of the audit data.

Assurance Activity AA-FAU_STG_EXT.1-ASE-02

The evaluator shall examine the TSS to ensure that for distributed TOEs it contains a list of TOE components that store audit data locally. The evaluator shall examine the TSS to ensure that for distributed TOEs that contain components which do not store audit data locally but transmit their generated audit data to other components it contains a mapping between the transmitting and storing TOE components.

Summary

According to the Security Target [ST] , the TOE is not distributed therefore this Evaluation Activity is not applicable.

Assurance Activity AA-FAU_STG_EXT.1-ASE-03

The evaluator shall examine the TSS to ensure that it details the behaviour of the TOE when the storage space for audit data is full. When the option "overwrite previous audit record" is selected this description should include an outline of the rule for overwriting audit data. If "other actions" are chosen such as sending the new audit data to an external IT entity, then the related behaviour of the TOE shall also be detailed in the TSS.

The evaluator shall examine the TSS to ensure that it details whether the transmission of audit information to an external IT entity can be done in real-time or periodically. In case the TOE does not perform transmission in real-time the evaluator needs to verify that the TSS provides details about what event stimulates the transmission to be made as well as the possible acceptable frequency for the transfer of audit data.

Summary

Chapter 7 of [ST] contains the TSS. Section 7.1 *Security Audit* describes auditing.

The evaluator examined the TSS and found the following details:

- When audit data exceeds the maximum storage space, the TOE will overwrite the oldest records. This is done by a cron job running every two minutes to check the audit trail storage partition.
- The TOE will overwrite the oldest records once the maximum log size is exceeded using a local syslog file rotation which numbers the locally archived syslog files and deletes the oldest syslog files.
- The audit records are sent to the remote storage immediately (i.e., in real-time).

The evaluator noted that the ST does not claim FAU_STG_EXT.2.

Thus, the evaluator verified that the TSS provides the necessary details regarding protection of the audit data.

Assurance Activity AA-FAU_STG_EXT.1-ASE-04

For distributed TOEs the evaluator shall examine the TSS to ensure it describes to which TOE components this SFR applies and how audit data transfer to the external audit server is implemented among the different TOE components (e.g. every TOE components does its own transfer or the data is sent to another TOE component for central transfer of all audit events to the external audit server).

For distributed TOEs the evaluator shall examine the TSS to ensure it describes which TOE components are storing audit information locally and which components are buffering audit information and forwarding the information to another TOE component for local storage. For every component the TSS shall describe the behaviour when local storage space or buffer space is exhausted.

Summary

According to the Security Target [ST] , the TOE is not distributed therefore this Evaluation Activity is not applicable.

Guidance Assurance Activities

Assurance Activity AA-FAU_STG_EXT.1-AGD-01

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

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

The evaluator shall also ensure that the guidance documentation describes all possible configuration options for FAU_STG_EXT.1.3 and the resulting behaviour of the TOE for each possible configuration. The description of possible configuration options and resulting behaviour shall correspond to those described in the TSS.

Summary

The evaluator examined section 2.3.8 "Event (audit) logging" and 10 Appendix "Sample Secure Remote Syslog Configuration" of [ECG] , where the guidance related to the SFR FAU_STG_EXT.1 is provided. The evaluator determined the guidance contains the following description:

- Logging must be configured to use a dedicated network interface. This ensures a limited attack surface for the administratively-controlled logging function.
- Secure remote logging of event records, and local logging as a backup in case the remote connection fails, are required. The logging framework will simultaneously send the event record to both of the subscribed recipients.

Section 2.3.8.1 "Configuring a dedicated network interface" of [ECG] provides the instructions to configure a dedicated network interface for logging as follows:

- Create a dedicated VLAN for logging
- Assign a data plane interface to the VLAN
- Assign one or more static self-IPs to the interface (several self-IPs help prevent source port exhaustion).
- Ensure that the remote syslog pool of servers created as described in section 2.3.8 of [ECG] is configured to be on the dedicated VLAN

FAU_STG_EXT.1.3 (section 6.2.1.4 of [ST]) specifies that the TSF shall overwrite previous audit records according to the following rule: log files are numbered and the oldest log file is deleted. The TSS (section 7.1 of [ST]) states that the TOE can be configured to log a warning if the local space for syslog files on the box exceeds a configurable maximum size; a local syslog file rotation scheme that numbers the locally archived syslog files. The TOE will delete the oldest syslog file once the maximum size for local syslog file space is exceeded. A cron job runs every two minutes to check the audit trail storage partition in order to accomplish this. The evaluated configuration requires allocation of 7 GB of audit storage, and a warning to be logged when 90% of the storage space is exhausted. The administrator receives the warnings when reviewing the log files as instructed the CC guidance document. In such event, the TOE automatically overwrites previous audit records.

The evaluator confirmed the guidance contains the corresponding information, in particular section 2.3.8 of [ECG] states the following:

- Secure remote logging of event records, and local logging as a backup in case the remote connection fails, are required.
- The logging framework will simultaneously send the event record to both of the remote and local storage.

The evaluator then examined 10 Appendix which provides information on how to configure the remote syslog. It specifically states that in order to configure secure logging to an external syslog server, a local SSL-to-server virtual server must be configured to encrypt the TCP syslog traffic generated by the BIG-IP's logging systems. Traffic from high-speed logging (HSL) system and standard syslog service are then sent to this virtual server. Additional configuration are described in detailed including:

- creating a pool for the high-speed logging system that contains the IP address and port of the local encrypting virtual server
- creating the SSL-to-server virtual server using the appropriate key and certificate
- modifying the local syslog server to send audit data to the encrypting virtual server as some of the older audit records do not use the high-speed logging system
- configuring the high-speed-logging destination targeting the pool
- in order to get syslog timestamp and other identifying information included with each log message, an HSL remote-syslog destination is created targeting the remote-high-speed-log
- creating an HSL publisher to send selected audit records to both the internal syslog server as well as the HSL destination
- creating an HSL filters to select audit records and send them to the remote system server

Test Assurance Activities

Assurance Activity AA-FAU_STG_EXT.1-ATE-01

Testing of the trusted channel mechanism for audit will be performed as specified in the associated assurance activities for the particular trusted channel mechanism. The evaluator shall perform the following additional tests for this requirement:

a) Test 1: The evaluator shall establish a session between the TOE and the audit server according to the configuration guidance provided. The evaluator shall then examine the traffic that passes between the audit server and the TOE during several activities of the evaluator's choice designed to generate audit data to be transferred to the audit server. The evaluator shall observe that these data are not able to be viewed in the clear during this transfer, and that they are successfully received by the audit server. The evaluator shall record the particular software (name, version) used on the audit server during testing. The evaluator shall verify that the TOE is capable of transferring audit data to an external audit server automatically without administrator intervention.

b) Test 2: The evaluator shall perform operations that generate audit data and verify that this data is stored locally. The evaluator shall perform operations that generate audit data until the local storage space is exceeded and verifies that the TOE complies with the behaviour defined in FAU_STG_EXT.1.3. Depending on the configuration this means that the evaluator has to check the content of the audit data when the audit data is just filled to the maximum and then verifies that

1) The audit data remains unchanged with every new auditable event that should be tracked but that the audit data is recorded again after the local storage for audit data is cleared (for the option "drop new audit data" in FAU_STG_EXT.1.3).

2) The existing audit data is overwritten with every new auditable event that should be tracked according to the specified rule (for the option "overwrite previous audit records" in FAU_STG_EXT.1.3)

3) The TOE behaves as specified (for the option "other action" in FAU_STG_EXT.1.3).

c) Test 3: If the TOE complies with FAU_STG_EXT.2/LocSpace the evaluator shall verify that the numbers provided by the TOE according to the selection for FAU_STG_EXT.2/LocSpace are correct when performing the tests for FAU_STG_EXT.1.3

d) Test 4: For distributed TOEs, Test 1 defined above should be applicable to all TOE components that forward audit data to an external audit server. For the local storage according to FAU_STG_EXT.1.2 and FAU_STG_EXT.1.3 the Test 2 specified above shall be applied to all TOE components that store audit data locally. For all TOE components that store audit data locally and comply with FAU_STG_EXT.2/LocSpace Test 3 specified above shall be applied. The evaluator shall verify that the transfer of audit data to an external audit server is implemented.

Summary

Test 1:

The evaluator established a connection between the TOE and an external audit server, started Wireshark to sniff the traffic between them and performed various operations to generate audit data. The data was encrypted and no data could be seen in clear text.

Test 2:
The ST states that the TOE overwrites previous audit records and therefore option 2 was tested. The evaluator checked whether the oldest audit record is deleted when the local storage space for audit data is full. The evaluator generated logs until the maximum audit file size limit was reached and verified that the oldest logs are being deleted. The other audit storage options are not applicable.

Test 3:
This test is not applicable since the TOE does not comply with FAU_STG_EXT.2/LocSpace.

Test 4:
This test is not applicable since the TOE is not distributed.

2.1.1.5 Action In Case of Possible Audit Data Loss (FAU_STG.3/LocSpace)

TSS Assurance Activities

Assurance Activity AA-FAU_STG_EXT.3-LS-ASE-01

> *The evaluator shall examine the TSS to ensure that it details how the user is warned before the local storage for audit data is full.*

Summary

Chapter 7 of [ST] contains the TSS. Section 7.1 *Security Audit* describes auditing which states the following:

The TOE logs a warning if the local space for syslog files on the box exceeds a configurable maximum size. The TOE implements a local syslog file rotation scheme that numbers the locally archived syslog files. The TOE will delete the oldest syslog file once the maximum size for local syslog file space is exceeded. A cron job runs every two minutes to check the audit trail storage partition in order to accomplish this. The evaluated configuration requires allocation of 7 GB of audit storage, and a warning to be logged when 90% of the storage space are exhausted. The administrator receives the warnings when reviewing the log files as instructed the CC (administrative) guidance document.

Assurance Activity AA-FAU_STG_EXT.3-LS-ASE-02

> *For distributed TOEs the evaluator shall examine the TSS to ensure it describes to which TOE components this SFR applies and how each TOE component realises this SFR. Since this SFR is optional, it might only apply to some TOE components but not all. This might lead to the situation where all TOE components store their audit information themselves but FAU_STG_EXT.3/LocSpace is supported only by one of the components. In particular, the evaluator has to verify, that the TSS describes for every component supporting this functionality, whether the warning is generated by the component itself or through another component and name the corresponding component in the latter case. The evaluator has to verify that the TSS makes clear any situations in which audit records might be "invisibly lost".*

Summary

According to the Security Target [ST] , the TOE is not distributed therefore this Evaluation Activity is not applicable.

Guidance Assurance Activities

Assurance Activity AA-FAU_STG_EXT.3-LS-AGD-01

The evaluator shall also ensure that the guidance documentation describes how the user is warned before the local storage for audit data is full and how this warning is displayed or stored (since there is no guarantee that an administrator session is running at the time the warning is issued, it is probably stored in the log files). The description in the guidance documentation shall correspond to the description in the TSS.

Summary

The evaluator examined section 2.3.8 "Event (audit) logging" of [ECG] , where guidance related to the SFR FAU_STG_EXT.3 is provided. The evaluator determined the following:

- A warning is issued when 90% of local log storage is full; this warning is logged in the log files.

Additionally, the evaluator examined section 3.3 "Audit Review" of [ECG] which instructs the administrator to review the audit data at least weekly.

FAU_STG_EXT.1.3 ([ST] section 6.2.1.4) defines that the TSF shall overwrite previous audit records according to by numbering log files and delete oldest log files. The TSS ([ST] section 7.1) defines that the TOE logs a warning if the local space for syslog files on the box exceeds a configurable maximum size, and will overwrite the oldest records once the maximum log size is exceeded by numbering log files and delete the oldest log file.

The evaluator confirmed the description in [ECG] corresponds to the description in the TSS. The evaluator also confirmed that the TOE automatically overwrites the audit logs as described above should the audit storage reaches the maximum size.

Test Assurance Activities

Assurance Activity AA-FAU_STG_EXT.3-LS-ATE-01

The evaluator shall verify that a warning is issued by the TOE before the local storage space for audit data is full.

Summary

The evaluator checked whether a warning message is issued before the local storage space for audit data is full. The evaluator generated audit logs until the maximum audit file size limit was reached and verified that a warning message is issued by the TOE before the local storage space for audit data is full.

Assurance Activity AA-FAU_STG_EXT.3-LS-ATE-02

For distributed TOEs the evaluator shall verify the correct implementation of display warning for local storage space for all TOE components that are supporting this feature according to the description in the TSS. The evaluator shall verify that each component that supports this feature according to the description in the TSS is capable of generating a warning itself or through another component.

Summary

The evaluator found that the TOE is not a distributed TOE, therefore there are no specific requirements for this assurance activity.

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

Summary

Chapter 7 of [ST] contains the TSS. Section 7.2.1 *Key Generation and Establishment* describes key generation and establishment. This section provides Table 6 "Key generation for the TOE" outlining the key generation scheme, key establishment scheme, key sizes / NIST curves, and their usage. The table is reproduced below:

Table 4: Key Generation and Establishment

Key Generation Scheme	Key Establishment Scheme	Key sizes / NIST curves	Usage
RSA	RSA RSA NIST SP 800-56B	Key sizes: 2048, 3072	TLS certificate TLS ephemeral session keys SSH key pair The TLS static keys are created once, imported to the TOE, and stored on disk until the Administrator creates a new key. The SSH key pair is created on first boot. The TOE can act as a receiver or both sender and receiver depending upon the deployment. When acting as a receiver, decryption errors are handled in a side channel resistant method and reported as MAC errors.
ECC	ECC NIST SP 800-56A	NIST curves: P-256, P-384	For ECDHE and ECDSA in TLS. The TOE can act as a receiver or both sender and receiver depending upon the deployment.

The evaluator verified the table above and determined that it identifies key sizes supported by the TOE and it also identifies the usage for each scheme. This description is found to be consistent with the definition of FCS_CKM.1.

Guidance Assurance Activities

Assurance Activity AA-FCS_CKM.1-AGD-01

The evaluator shall verify that the AGD guidance instructs the administrator how to configure the TOE to use the selected key generation scheme(s) and key size(s) for all cryptographic protocols defined in the Security Target.

Summary

[ST] specifies following key generation schemes in FCS_CKM.1:

- RSA with 2048-bit or greater key;
- ECC with NIST curves p-256 and p-384.

For TLS, the evaluator examined section 2.3.10.1 "SSL Profiles" of [ECG] , which states that the ccmode command sets the allowable ciphersuites. The evaluator found in section 4 "Appendix: ccmode command" a description of the ccmode command including the setting of the allowable key generation schemes.

For SSH, the evaluator examined section 2.3.10.2 "SSH" of [ECG] , which states that the ccmode command sets the SSH server profile to use only allowable ciphersuites and unsupported ciphersuites such as diffie-hellman-group14-sha1 must be removed from the profile per instructions provided in section 2.2.3.2 "Updating the SSH cipher configuration.

Thus, the evaluator determined that the administrator does not need to configure the TOE to use the key generation scheme and key size. The cryptographic keys are generated along with the TLS client certificates. The use of the key establishment schemes is done automatically during the negotiation of TLS/SSH session establishment.

Test Assurance Activities

Assurance Activity AA-FCS_CKM.1-ATE-01

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. Generation of long-term cryptographic keys (i.e. keys that are not ephemeral keys/session keys) might be performed automatically (e.g. during initial start-up). Testing of key generation must cover not only administrator invoked key generation but also automated key generation (if supported).

Key Generation for FIPS PUB 186-4 RSA Schemes

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

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

a) Random Primes:

- *Provable primes*
- *Probable primes*

b) 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 shall verify the correctness of the TSF's implementation by comparing values generated by the TSF with those generated from a known good implementation.

Key Generation for Elliptic Curve Cryptography (ECC)

FIPS 186-4 ECC Key Generation Test

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

FIPS 186-4 Public Key Verification (PKV) Test

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

Key Generation for Finite-Field Cryptography (FFC)

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

The Parameter generation specifies 2 ways (or methods) to generate the cryptographic prime q and the field prime p:

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

and two ways to generate the cryptographic group generator g:

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

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

- *len(q) bit output of RBG where 1 <=x <= q-1*
- *len(q) + 64 bit output of RBG, followed by a mod q-1 operation and a +1 operation, where 1<= x<=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 shall have the TSF generate 25 parameter sets and key pairs. The evaluator shall verify the correctness of the TSF's implementation by comparing values generated by the TSF with those generated from a known good implementation. Verification must also confirm

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

for each FFC parameter set and key pair.

FFC Schemes using “safe-prime” groups

[TD0580] Testing for FFC Schemes using safe-prime groups is done as part of testing in CKM.2.1.

Summary

This test is covered by CAVS-test. Please see Table 2 for a mapping to the CAVP certificates.

2.1.2.2 Cryptographic Key Dstablishment (FCS_CKM.2)

TSS Assurance Activities

Assurance Activity AA-FCS_CKM.2-ASE-01

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

The intent of this activity is to be able to identify the scheme being used by each service.

Summary

Section 7 of [ST] contains the TSS. Section 7.2.1 *Key Generation and Establishment* describes key generation and establishment. This section provides table 6 outlining the key generation scheme, key establishment scheme, key sizes / NIST curves, and their usage. The table is reproduced in the previous Evaluation Activity.

The evaluator verified table 6 of the TSS and determined that it identifies the supported key establishment schemes consistent to the key generation schemes identified in FCS_CKM.1.1. Also, for each scheme, its usage is identified including whether the TOE acts as a sender, a recipient, or both. For example, the second table entry identifies ECC as being used for ECDHE and ECDSA in TLS and the TOE acts as a receiver or sender depending on the deployment.

The evaluator noted that Diffie-Hellman group 14 is not selected in FCS_CKM.2.1.

Guidance Assurance Activities

Assurance Activity AA-FCS_CKM.2-AGD-01

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

Summary

According to [ST] section 6.2.2.2, the TOE supports following key establishment schemes:

- RSAES-PKCS1- v1_5 as specified in Section 7.2 of RFC 3447, “Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1"
- NIST Special Publication 800-56A Revision 2, “Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography”

The evaluator examined sections 2.3.10.1 "SSL Profiles" and 2.3.10.2 "SSH" of [ECG] and determined that the ccmode command sets the allowable key establishment schemes and no additional user actions are required to configure the TOE to use the key establishment schemes as this is done automatically during the negotiation of TLS/SSH session establishment.

Test Assurance Activities

Assurance Activity AA-FCS_CKM.2-ATE-01

Key Establishment Schemes

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

SP800-56A Key Establishment Schemes

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

Function Test

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

The evaluator shall obtain the DKM, the corresponding TOE's public keys (static and/or ephemeral), the MAC tag(s), and any inputs used in the KDF, such as the Other Information field OI and TOE id fields.

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

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

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

Validity Test

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

The evaluator shall inject an error in some of the test vectors to test that the TOE recognizes invalid key agreement results caused by the following fields being incorrect: the shared secret value Z, the DKM, the other information field OI, the data to be MACed, or the generated MACTag. If the TOE contains the full or partial (only ECC) public key validation, the evaluator will also individually inject errors in both parties' static public keys, both parties' ephemeral public keys and the TOE's static private key to assure the TOE detects errors in the public key validation function and/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 TOE shall use these modified test vectors to emulate the key agreement scheme using the corresponding parameters. The evaluator shall compare the TOE's results with the results using a known good implementation verifying that the TOE detects these errors.

RSA-based key establishment

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_TRP.1/Admin, FTP_TRP.1/Join, FTP_ITC.1 and FPT_ITT.1 that uses RSAES-PKCS1-v1_5.

FFC Schemes using "safe-prime" groups

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_TRP.1/Admin, FTP_TRP.1/Join, FTP_ITC.1 and FPT_ITT.1 that uses safe-prime groups. This test must be performed for each safe-prime group that each protocol uses.

Summary

This test is covered by CAVS-test. Please see Table 2 for a mapping to the CAVP certificates.

2.1.2.3 Cryptographic Key Destruction (FCS_CKM.4)

TSS Assurance Activities

Assurance Activity AA-FCS_CKM.4-ASE-01

The evaluator examines the TSS to ensure it lists all relevant keys (describing the origin and storage location of each), all relevant key destruction situations (e.g. factory reset or device wipe function, disconnection of trusted channels, key change as part of a secure channel protocol), and the destruction method used in each case. For the purpose of this Evaluation Activity the relevant keys are those keys that are relied upon to support any of the SFRs in the Security Target. The evaluator confirms that the description of keys and storage locations is consistent with the functions carried out by the TOE (e.g. that all keys for the TOE-specific secure channels and protocols, or that support FPT_APW.EXT.1 and FPT_SKP_EXT.1, are accounted for). In particular, if a TOE claims not to store plaintext keys in non-volatile memory then the evaluator checks that this is consistent with the operation of the TOE.

The evaluator shall check to ensure the TSS identifies how the TOE destroys keys stored as plaintext in non-volatile memory, and that the description includes identification and description of the interfaces that the TOE uses to destroy keys (e.g., file system APIs, key store APIs).

Note that where selections involve "destruction of reference" (for volatile memory) or "invocation of an interface" (for non-volatile memory) then the relevant interface definition is examined by the evaluator to ensure that the interface supports the selection(s) and description in the TSS. In the case of non-volatile memory, the evaluator includes in their examination the relevant interface description for each media type on which plaintext keys are stored. The presence of OS-level and storage device-level swap and cache files is not examined in the current version of the Evaluation Activity.

Where the TSS identifies keys that are stored in a non-plaintext form, the evaluator shall check that the TSS identifies the encryption method and the key-encrypting-key used, and that the key-encrypting-key is either itself stored in an encrypted form or that it is destroyed by a method included under FCS_CKM.4.

The evaluator shall check that the TSS identifies any configurations or circumstances that may not conform to the key destruction requirement. Note that reference may be made to the Guidance Documentation for description of the detail of such cases where destruction may be prevented or delayed.

Where the ST specifies the use of "a value that does not contain any CSP" to overwrite keys, the evaluator examines the TSS to ensure that it describes how that pattern is obtained and used, and that this justifies the claim that the pattern does not contain any CSPs.

Summary

Chapter 7 of [ST] contains the TSS. Section 7.2.2 *Zeroization of Critical Security Parameters* describes zeroization of critical security parameters. This section also provides table 7 " Zeroization of Critical Security Parameters" outlining how critical security parameters used for TOE-specific secure channels and protocols are zeroized by the TOE's OpenSSL when they are no longer in use. The table is reproduced below.

Table 5: Zeroization of Critical Security Parameters

Application	Key type	Storage location	Volatile/ Non-volatile	Zeroized when?	Description
Key generation	seeds, prime numbers	Stack/heap	Volatile	After each key has been generated.	These are zeroized in OpenSSL by calling OPENSSL_cleanse(), which overwrites the memory upon release
TLS	Session keys	Stack/heap	Volatile	After session has ended	The TLS session keys are created within OpenSSL during session initiation. These are zeroized in OpenSSL by calling OPENSSL_cleanse(), which overwrites the memory upon release
TLS	private keys in TLS certificates	On the disk	Non-volatile	Upon deletion by administrator.	Private keys are zeroized when they are deleted by the administrator. Zeroization is done by overwriting the file once with zeroes and deleting

Application	Key type	Storage location	Volatile/ Non-volatile	Zeroized when?	Description
					the file. The API used for zeroization is the write(2) system call which is called with buffer filled with zeros as input.
SSH	Session keys	Stack/heap	Volatile	After session has ended	The SSH session keys are created within OpenSSL during session initiation These are zeroized in OpenSSL by calling OPENSSL_cleanse(), which overwrites the memory upon release.
SSH	SSH keys	On the disk	Non-volatile	Upon deletion by administrator.	SSH keys are zeroized when using the key-swap utility. Zeroization is done by overwriting the file once with zeroes and deleting the file. The API used for zeroization is the shred(1) Linux command which uses the write(2) system call which is called with buffer filled with zeros as input.

This section of the TSS also states the following in:

- Only session keys for TLS and SSH are stored in plaintext form, other keys are stored encrypted.
- Encrypted keys are stored via F5 Secure Vault which uses a Master Key and a Unit Key to protect sensitive configuration attributes such as passwords and passphrases.
- Master Key is a 128-bit AES symmetric key that is stored with the data it protects.
- The Unit Key which is a key-encrypting-key is a symmetric key stored in the EEPROM associated with the TOE device and is used to protect the Master Key.
- The Unit Key (a key-encrypting-key) is a symmetric key stored in a hidden file in the file system that is associated with the device and is used to protect the Master Key.
- If the Unit Key is replaced, the old Unit Key is cleared by overwriting it with random data and then the new Unit Key is written.

The evaluator noted that the ST does not select ‘destruction of reference’ (for volatile memory) thus no corresponding TSS description is required.

The evaluator noted that the ST does specify ‘invocation of an interface’ (for non-volatile memory) which requires the relevant interface definition to be examined by the evaluator to ensure that the interface supports the selection(s) and description in the TSS. In the case of non-volatile memory the evaluator includes in their examination the relevant interface description for each media type on which plaintext keys are stored. This is to be done as part of guidance evaluation.

The evaluator note that the TSS does not identify any configurations or circumstances that may not conform to the key destruction requirement.

The evaluator noted that the ST does not specify the use of 'a value that does not contain any CSP' to overwrite keys, thus no corresponding TSS description is required.

The evaluator verified that the TSS (table 7) describes how each key is cleared.

Guidance Assurance Activities

Assurance Activity AA-FCS_CKM.4-AGD-01

A TOE may be subject to situations that could prevent or delay key destruction in some cases. The evaluator shall 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 supporting information used). The evaluator shall check that the guidance documentation provides guidance on situations where key destruction may be delayed at the physical layer.

For example, when the TOE does not have full access to the physical memory, it is possible that the storage may be implementing wear-levelling and garbage collection. This may result in additional copies of the key that are logically inaccessible but persist physically. Where available, the TOE might then describe use of the TRIM command and garbage collection to destroy these persistent copies upon their deletion (this would be explained in TSS and Operational Guidance).

Summary

The TSS of [ST] in section 7.2.2 describes zeroization critical security parameters including cryptographic keys when they are no longer in use by the TOE. It states the following:

- Seeds and prime numbers used for key generation are destroyed after each key has been generated. The destruction involves zeroizing in OpenSSL via the OPENSSL_cleanse() which overwrites the memory upon release.
- TLS session keys are destroyed after the session has ended. The destruction involves zeroizing in OpenSSL via the OPENSSL_cleanse() which overwrites the memory upon release.
- Private keys in TLS certificates are deleted by administrator. Zeroization is done by overwriting the file once with zeroes and deleting the file. The API used for zeroization is the write(2) system call which is called with buffer filled with zeros as input.
- SSH session keys are destroyed after the session has ended. The destruction involves zeroizing in OpenSSL via the OPENSSL_cleanse() which overwrites the memory upon release.
- SSH keys are deleted by administrator using the key-swap utility. SSH keys are zeroized when using the key-swap utility. Zeroization is done by overwriting the file once with zeroes and deleting the file. The API used for zeroization is the shred(1) Linux command which uses the write(2) system call which is called with buffer filled with zeros as input.

While examining the TSS description summarized above, the evaluator determined that the TSS does not explicitly identify any configurations or circumstances where the TOE may not strictly conform to the key destruction requirement or any situations where key destruction may be delayed at the physical layer. Thus, the evaluator concluded that no guidance was necessary.

Test Assurance Activities

No assurance activities defined.

2.1.2.4 Cryptographic operation (AES Data Encryption/Decryption) (FCS_COP.1/DataEncryption)

TSS Assurance Activities

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

The evaluator shall examine the TSS to ensure it identifies the key size(s) and mode(s) supported by the TOE for data encryption/decryption.

Summary

Chapter 7 of [ST] contains the TSS. Section 7.2.3 *Cryptographic operations in the TOE* provides table 7 "Cryptographic primitives in the TOE" which identifies the following key sizes and modes supported by the TOE for data encryption/decrypton:

- Algorithm: AES
- Modes: CBC, GCM
- Key length (bits): 128, 256

The evaluator determined that the TSS identifies the key size(s) and mode(s) supported by the TOE for data encryption/decryption.

Guidance Assurance Activities

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

The evaluator shall verify that the AGD guidance instructs the administrator how to configure the TOE to use the selected mode(s) and key size(s) defined in the Security Target supported by the TOE for data encryption/decryption.

Summary

[ST] specifies the following selected mode(s) and key size(s) in FCS_COP.1/DataEncryption:

- AES used in [CBC, GCM] mode
- cryptographic key sizes [128 bits, 256 bits]

For SSH, the evaluator examined section 2.3.10.2 "SSH" of [ECG] , which states that the ccmode command sets the SSH server profile to use only allowable ciphersuites and unsupported ciphersuites such as aes128-gmc, aes256-gmc , aes128-ctr, and aes 256-ctr must be removed from the profile per instructions provided in section 2.2.3.2 "Updating the SSH cipher configuration". The evaluator found in section 4 "Appendix: ccmode command" a description of the ccmode command including the updating of the sshd configuration file to only allow aes128-cbc and aes256-cbc as the allowed ciphers. Thus, the evaluator determined that the administrator does not need to configure the TOE for data encryption/decryption. For TLS, the evaluator examined section 2.3.10.3 "Configuration Utility" of the [ECG], which states that the ccmode command sets the configuration utility SSL ciphers and protocols. However, in order to restrict the allowed curves, a further restriction on these ciphers must be applied. Run the following command to properly configure the ciphers for the configuration utility:

```
tmsh modify /sys httpd {ssl-ciphersuite RSA+AES:@STRENGTH ssl-protocol “all -SSLv2
 -SSLv3 -TLSv1”} RSA
```

Test Assurance Activities

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

AES-CBC Known Answer Tests

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

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

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

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

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

KAT-3. *To test the encrypt functionality of AES-CBC, the evaluator shall supply the two sets of key values described below and obtain the ciphertext value that results from AES encryption of an all-zeros plaintext using the given key value and an IV of all zeros. The first set of keys shall have 128 128-bit keys, and the second set shall have 256 256-bit keys. Key i in each set shall have the leftmost i bits be ones and the rightmost N-i bits be zeros, for i in [1,N].*

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

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

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

AES-CBC Multi-Block Message Test

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

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

AES-CBC Monte Carlo Tests

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

```
# Input: PT, IV, Key
for i = 1 to 1000:
if i == 1:
CT[1] = AES-CBC-Encrypt(Key, IV, PT)
PT = IV
else:
CT[i] = AES-CBC-Encrypt(Key, PT)
PT = CT[i-1]
```

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

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

AES-GCM Test

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

128 bit and 256 bit keys

a) 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.

b) 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.

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

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

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

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

AES-CTR Known Answer Tests

*The Counter (CTR) mode is a confidentiality mode that features the application of the forward cipher to a set of input blocks, called counters, to produce a sequence of output blocks that are exclusive-ORed with the plaintext to produce the ciphertext, and vice versa. Since the Counter Mode does not specify the counter that is used, it is not possible to implement an automated test for this mode. The generation and management of the counter is tested through FCS_SSH*_EXT.1.4. If CBC and/or GCM are selected in FCS_COP.1/DataEncryption, the test activities for those modes sufficiently demonstrate the correctness of the AES algorithm. If CTR is the only selection in FCS_COP.1/DataEncryption, the AES-CBC Known Answer Test, AES-GCM Known Answer Test, or the following test shall be performed (all of these tests demonstrate the correctness of the AES algorithm):*

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

KAT-1 *To test the encrypt functionality, the evaluator shall supply a set of 5 plaintext values for each selected keysize and obtain the ciphertext value that results from encryption of the given plaintext using a key value of all zeros.*

KAT-2*To test the encrypt functionality, the evaluator shall supply a set of 5 key values for each selected keysize and obtain the ciphertext value that results from encryption of an all zeros plaintext using the given key value.*

KAT-3*To test the encrypt functionality, the evaluator shall supply a set of key values for each selected keysize as described below and obtain the ciphertext values that result from AES encryption of an all zeros plaintext using the given key values. A set of 128 128-bit keys, a set of 192 192-bit keys, and/or a set of 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].*

KAT-4*To test the encrypt functionality, the evaluator shall supply the set of 128 plaintext values described below and obtain the ciphertext values that result from encryption of the given plaintext using each selected keysize with a key value of all zeros (e.g. 256 ciphertext values will be generated if 128 bits and 256 bits are selected and 384 ciphertext values will be generated if all keysizes are selected). Plaintext value i in each set shall have the leftmost bits be ones and the rightmost 128-i bits be zeros, for i in [1, 128].*

AES-CTR Multi-Block Message Test

The evaluator shall test the encrypt functionality by encrypting an i-block message where 1 less-than i less-than-or-equal to 10 (test shall be performed using AES-ECB mode). For each i the evaluator shall choose a key and plaintext message of length i blocks and encrypt the message, using the mode to be tested, with the chosen key. The ciphertext shall be compared to the result of encrypting the same plaintext message with the same key using a known good implementation. The evaluator shall perform this test using each selected keysize.

AES-CTR Monte-Carlo Test

The evaluator shall test the encrypt functionality using 100 plaintext/key pairs. The plaintext values shall be 128-bit blocks. For each pair, 1000 iterations shall be run as follows:

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

The ciphertext computed in the 1000th iteration is the result for that trial. This result shall be compared to the result of running 1000 iterations with the same values using a known good implementation. The evaluator shall perform this test using each selected keysize.

There is no need to test the decryption engine.

Summary

This test is covered by CAVS-test. Please see Table 2 for a mapping to the CAVP certificates.

2.1.2.5 Cryptographic operation (Hash Algorithm) (FCS_COP.1/Hash)

TSS Assurance Activities

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

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

Summary

Chapter 7 of [ST] contains the TSS. Section 7.2.3 *Cryptographic operations in the TOE* describes the cryptographic operations in the TOE. This section provides Table 8 "Cryptographic primitives in the TOE" listing for each cryptographic operation the associated algorithm, key lengths, standard, and SFRs. The table is reproduced below:

Table 6: Cryptographic operations in the TOE

Algorithm	Keylength (bits)	Purpose	Reference	SFR
AES (CBC, GCM modes)	128 256	payload encryption	AES as specified by ISO 18033-3 CBC as specified in ISO 10116 GCM as specified in ISO 19772	FCS_COP.1/DataEncryption
RSA	Modulus of 2048, 3072	certificate-based authentication, key exchange	FIPS PUB 186-4 Section 5.5 using RSASSA-PKCS1v1_5, ISO/IEC 9796-2	FCS_COP.1/SigGen
ECDSA	256, 384 bits NIST curves: P-256, P-384, and no other	certificate-based authentication, key exchange	FIPS PUB 186-4 Section 6 and Appendix D ISO/IEC 14888-3 Section 6.4	FCS_COP.1/SigGen
SHA-1 SHA-256 SHA-384	none	certificate-based authentication / digital signature verification	ISO/IEC 10118-3:2004	FCS_COP.1/Hash
HMAC-SHA-1	Key sizes: ≥ 160 bits Hash Function: SHA-1 Message digest sizes: 160 bits Block size: 512 bits Output MAC length: 160 bits	message integrity	ISO/IEC 9797-2:2011, Section 7	FCS_COP.1/KeyedHash
HMAC-SHA-256	Key sizes: ≥ 256 bits Hash Function: SHA-256 Message digest sizes: 256 bits Block size: 512 bits Output MAC length: 256 bits	message integrity	ISO/IEC 9797-2:2011, Section 7	FCS_COP.1/KeyedHash

Algorithm	Keylength (bits)	Purpose	Reference	SFR
HMAC-SHA-384	Key sizes: ≥ 384 bits Hash Function: SHA-384 Message digest sizes: 388 bits Block size: 1024 bits Output MAC length: 384 bits	message integrity	ISO/IEC 9797-2:2011, Section 7	FCS_COP.1/KeyedHash
Random Bit Generation	none	key generation	ISO/IEC 18031:2011 using CTR DRBG (AES)	FCS_RBG_EXT.1

As listed in the fourth table entry, SHA-1, SHA-256, and SHA-384 are the hash functions supported by the TOE which are consistent with the definition of FCS_COP.1/Hash. The column "Purpose" also points out that these hash functions are used for certificate-based authentication and digital signature verification.

Guidance Assurance Activities

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

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

Summary

According to [ST] section 6.2.2.6, the TOE supports SHA-1, SHA-256, and SHA-384.

The evaluator examined sections 2.3.10.1 "SSL Profiles", 2.3.10.2 "SSH", 5.2 "SSH Server Protocol", and section 4 "Appendix: ccmode command" and determined that there is no user specific configuration exist for all the cryptographic algorithms and the selection is based on negotiation during SSH/TLS session establishment.

Test Assurance Activities

Assurance Activity AA-FCS_COP.1-HASH-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 shall perform all of the following tests for each hash algorithm implemented by the TSF and used to satisfy the requirements of this PP.

Short Messages Test - Bit-oriented Mode

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

Short Messages Test - Byte-oriented Mode

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

Selected Long Messages Test - Bit-oriented Mode

*The evaluators devise an input set consisting of m messages, where m is the block length of the hash algorithm (e.g. 512 bits for SHA-256). The length of the i^{th} message is m + 99*i, where 1 i m. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF.*

Selected Long Messages Test - Byte-oriented Mode

*The evaluators devise an input set consisting of m/8 messages, where m is the block length of the hash algorithm (e.g. 512 bits for SHA-256). The length of the i^{th} message is m + 8*99*i, where 1 i m/8. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF.*

Pseudorandomly Generated Messages Test

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

Summary

This test is covered by CAVS-test. Please see Table 2 for a mapping to the CAVP certificates.

2.1.2.6 Cryptographic operation (Keyed Hash Algorithm) (FCS_COP.1/KeyedHash)

TSS Assurance Activities

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

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

Summary

Chapter 7 of [ST] contains the TSS. Section 7.2.3 *Cryptographic operations in the TOE* describes the cryptographic operations in the TOE. This section provides Table 8 "Cryptographic primitives in the TOE" listing for each cryptographic operation the associated algorithm, key lengths, standard, and SFRs. The table is reproduced the previous Evaluation Activity AA-FCS_COP.1-HASH-ASE-01 .

As listed Table 8, the HMAC functions supported for message integrity are HMAC-SHA-1, HMAC-SHA-256, and HMAC-SHA-384. The corresponding key length, message digest sizes, and output MAC length are listed in the column "Key length (bits)". The evaluator verified the information in this table is consistent with the definition of FCS_COP.1/KeyHashed.

Guidance Assurance Activities

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

The evaluator shall verify that the AGD guidance instructs the administrator how to configure the TOE to use the values used by the HMAC function: key length, hash function used, block size, and output MAC length used defined in the Security Target supported by the TOE for keyed hash function.

Summary

[ST] section 6.2.2.10 defines FCS_COP.1/KeyedHash which states that the keyed-hash message authentication uses HMAC-SHA-1, HMAC-SHA-256, HMAC- SHA-384 and cryptographic key sizes for SHA-1 the key size is ≥ 160 bits, for SHA- 256 the key size is ≥ 256 bits, for SHA-384 the key size is ≥ 384 bits used in HMAC and message digest sizes [160, 256, 384] bits. The TSS (section 7.2.3 - "Cryptographic operations in the TOE") of the [ST] list the following:

- HMAC-SHA-1: Key sizes >= 160 bits, Hash function: SHA-1, Message digest sizes: 160 bits, Block size: 512 bits, Output MAC length: 160 bits
- HMAC-SHA-256: Key sizes >= 256 bits, Hash function: SHA-256, Message digest sizes: 256 bits, Block size: 512 bits, Output MAC length: 256 bits
- HMAC-SHA-384: Key sizes >= 384 bits, Hash function: SHA-384, Message digest sizes: 384 bits, Block size: 1024 bits, Output MAC length: 384 bits

For related guidance, within the [ECG] , the evaluator identified section 2.2.3.2 "Updating the SSH cipher configuration" for instructions on removing unsupported algorithms from the default configuration which refers the user manual [K80425458] "K80425458: Modifying the list of ciphers and MAC algorithms used. The evaluator examined [K80425458] along with specific instructions provided in section 2.2.3.2 "Updating the SSH cipher configuration" in [ECG] and verified that it contains sufficient instructions to disable the data integrity algorithms not allowed in the evaluated configuration e.g., hmac-sha2-512. Please note that the set of algorithms are restricted when the TOE is configured to the CC mode (ccmode command), which is an initial configuration step for using the TOE (section 2.3.1 "ccmode command" of [ECG]).

Test Assurance Activities

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

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

Summary

This test is covered by CAVS-test. Please see Table 2 for a mapping to the CAVP certificates.

2.1.2.7 Cryptographic operation (Signature Generation and Verification) (FCS_COP.1/SigGen)

TSS Assurance Activities

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

The evaluator shall examine the TSS to determine that it specifies the cryptographic algorithm and key size supported by the TOE for signature services.

Summary

Chapter 7 of [ST] contains the TSS. Section 7.5.3 *Update Verification* states the following:

- A signature file exists for each software change update provided by F5. The content of the signature file is a digital signature of a SHA-256 digest of the image file.

Also, section 7.2.3 *Cryptographic operations in the TOE* describes the cryptographic operations in the TOE. This section provides Table 7 "Cryptographic primitives in the TOE" listing for each cryptographic operation the associated algorithm, key lengths, standard, and SFRs. This table which is reproduced in the next Evaluation Activity, in the fourth table entry, SHA-1, SHA-256, and SHA-384 are the hash functions supported by the TOE which are consistent with the definition of FCS_COP.1/Hash. The column "Purpose" also points out that these hash functions are used for certificate-based authentication and digital signature verification.

Guidance Assurance Activities

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

The evaluator shall verify that the AGD guidance instructs the administrator how to configure the TOE to use the selected cryptographic algorithm and key size defined in the Security Target supported by the TOE for signature services.

Summary

[ST] specifies following cryptographic signature services (generation and verification) in FCS_COP.1/SigGen:

- RSA Digital Signature Algorithm and cryptographic key sizes (modulus) [2048 bits or greater]
- Elliptic Curve Digital Signature Algorithm and cryptographic key sizes [256 bits or greater]

The evaluator examined section 2.3.10.1 "SSL Profiles" of [ECG] , which states that the ccmode command sets the allowable ciphersuites for the default client and server SSL profiles: clientssl and serverssl. It also indicates to create and use SSL profiles based only off those default profiles, and do not modify the configured ciphersuites, in order to ensure that your TLS connections are Common-Criteria-compliant. When configuring SSL profiles, only use 2048-bit or higher RSA key sizes, or ECDSA curves p-256 or p-384. The admin is refer to *ltm profile server-ssl* in the [TMSH-REFv17] . The evaluator found in section 4 "Appendix: ccmode command" a description of the ccmode command including the command to ensure that SSL profiles only use the restrictive set of ciphers. Thus, the evaluator determined that the administrator does not need to perform any additional steps to configure the TOE to use the cryptographic signature services.

Test Assurance Activities

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

ECDSA Algorithm Tests

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 shall generate 10 1024-bit long messages and obtain for each message a public key and the resulting signature values R and S. To determine correctness, the evaluator shall use the signature verification function of a known good implementation.

ECDSA FIPS 186-4 Signature Verification Test

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

RSA Signature Algorithm Tests

Signature Generation Test

The evaluator generates or obtains 10 messages for each modulus size/SHA combination supported by the TOE. The TOE generates and returns the corresponding signatures.

The evaluator shall verify the correctness of the TOE's signature using a trusted reference implementation of the signature verification algorithm and the associated public keys to verify the signatures.

Signature Verification Test

For each modulus size/hash algorithm selected, the evaluator generates a modulus and three associated key pairs, (d, e). Each private key d is used to sign six pseudorandom messages each of 1024 bits using a trusted reference implementation of the signature generation algorithm. Some of the public keys, e, messages, or signatures are altered so that signature verification should fail. For both the set of original messages and the set of altered messages: the modulus, hash algorithm, public key e values, messages, and signatures are forwarded to the TOE, which then attempts to verify the signatures and returns the verification results.

The evaluator verifies that the TOE confirms correct signatures on the original messages and detects the errors introduced in the altered messages.

Summary

This test is covered by CAVS-test. Please see Table 2 for a mapping to the CAVP certificates.

2.1.2.8 HTTPS Protocol (FCS_HTTPS_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FCS_HTTPS_EXT.1-ASE-01

The evaluator shall examine the TSS and determine that enough detail is provided to explain how the implementation complies with RFC 2818.

Summary

Chapter 7 of [ST] contains the TSS. Section 7.2.7 *HTTPS Protocol* describes the HTTPS protocol. It states the following:

- The TOE implements HTTPS per RFC 2818, HTTP over TLS.
- The HTTPS implementation complies with all mandatory portions of RFC 2818 (as denoted in the RFC by keywords “MUST”, “MUST NOT”, and “REQUIRED”) including Connection Initiation, Connection Closure, Client Behavior, Server Behavior, and Server Identity.
- For Connection Closure, the TOE includes a configuration setting in the SSL profile that controls alert protocols and the session close behavior.
- By default, the TOE is configured to close the underlying TCP connections without exchanging the required TLS shutdown close notify.
- The TOE can be configured to perform a clean shutdown of all TLS connections by sending a close notify.

The evaluator verified that the TSS provides sufficient detail explaining how the TOE HTTPS implementation complies with RCF 2818.

Guidance Assurance Activities

Assurance Activity AA-FCS_HTTPS_EXT.1-AGD-01

The evaluator shall examine the guidance documentation to verify it instructs the Administrator how to configure TOE for use as an HTTPS client or HTTPS server.

Summary

[ST] specifies the TSF shall implement the HTTPS protocol over TLS that complies with RFC 2818. Section 7.2.7 "HTTPS Protocol" of the [ST] states that the HTTPS implementation is designed to comply with all mandatory portions of RFC 2808. The evaluator examined section 2.3.10.3 "Configuration Utility" of [ECG] , which states that ccmode script applies the following command for configuring utility SSL ciphers and protocols:

```
tmsh modify /sys httpd ( ssl-ciphersuite EECDH+AES:RSA+AES:@STRENGTH ssl- protocol
 all -SSLv2 -SSLv3 -TLSv1)
```

The same section states that in order to restrict the allowed curves, a further restriction on these ciphers must be applied. The section then provides the following command to properly configure the ciphers:

```
tmsh modify /sys httpd {ssl-ciphersuite RSA+AES:@STRENGTH ssl-protocol “all -SSLv2
 -SSLv3 -TLSv1”} RSA
```

The evaluator determined that the guidance provides the necessary documentation regarding how to configure TOE for use as an HTTPS client or HTTPS server.

Test Assurance Activities

Assurance Activity AA-FCS_HTTPS_EXT.1-ATE-01

This test is now performed as part of FIA_X509_EXT.1/Rev testing.

Tests are performed in conjunction with the TLS evaluation activities.

If the TOE is an HTTPS client or an HTTPS server utilizing X.509 client authentication, then the certificate validity shall be tested in accordance with testing performed for FIA_X509_EXT.1.

Summary

The evaluator used a computer running Ubuntu Linux as a HTTPS client and established a connection between the TOE and a client via configuration utility, started Wireshark to capture the traffic between them and checked if HTTPS connection was established. The evaluator used a computer running Ubuntu Linux as a HTTPS client and established a connection between the TOE and a client via IControl Soap, started Wireshark to capture the traffic between them and checked if HTTPS connection was established. The evaluator used a computer running Ubuntu Linux as a HTTPS client and established a connection between the TOE and a client via IControl Rest, started Wireshark to capture the traffic between them and checked if HTTPS connection was established.

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

TSS Assurance Activities

Assurance Activity AA-FCS_RBG_EXT.1-ASE-01

Documentation shall be produced (and the evaluator shall perform the activities) in accordance with Appendix D of [NDcPPv2.2e].

The evaluator shall examine the TSS to determine that it specifies the DRBG type, identifies the entropy source(s) seeding the DRBG, and state the assumed or calculated min-entropy supplied either separately by each source or the min-entropy contained in the combined seed value.

Summary

Chapter 7 *TOE Summary Specification* of [ST] contains the TSS. Section 7.2.4 *Random Number Generation* describes random number generation supported by the TOE. Also, section 7.2.3 "Cryptographic operations in the TOE" provides Table 8 "Cryptographic primitives in the TOE" which the last table entry identifies the random bit generation for key generation uses the CTR_DRBG (AES).

Section 7.2.4 states the following:

- The TOE transfers one or more random bit-streams from the defined entropy sources to entropy pool of the TOE's Linux OS. The bit stream will be transferred as needed during system operation.
- On F5 devices and vCMP, the defined sources will be specific to the hardware available on each platform but will include one or more of the following: the jitterentropy-engine, Cavium Nitrox III hardware, Intel QAT hardware, and the Intel rdrand instruction. On hypervisors, the jitterentropy-engine is the second entropy source.
- The entropy pool is used as a seed source for DRNG via the /dev/random and /dev/urandom.
- The random bit stream from the entropy source will be fed to the Linux DRNG on demand and fresh entropy is inserted and the entropy estimate is increased when it is low, thus, ensuring sufficient entropy is always available to prevent blocking applications that read from /dev/random, or will release any blocked applications.
- The increase in the entropy estimate caused by the transfer of the random bit stream is not equal to the number of bits transferred, rather it scaled by a factor which is dependent on the entropy source.

The evaluator also examined the Entropy Documentation as required by Appendix D of [NDcPPv2.2e] and considered the documentation sufficiently addresses all the requirements in Appendix D.

Guidance Assurance Activities

Assurance Activity AA-FCS_RBG_EXT.1-AGD-01

The evaluator shall confirm that the guidance documentation contains appropriate instructions for configuring the RNG functionality.

Summary

In the evaluated configuration, the TOE uses the default RNG functionality the Linux DRNG that comes with the TOE [ST] . No configuration is required by the administrator as stated in [ECG] section 2.1:

> *The random number generator implemented in BIG-IP does not require configuration because the entropy sources are securely configured by default.*

Test Assurance Activities

Assurance Activity AA-FCS_RBG_EXT.1-ATE-01

The evaluator shall perform 15 trials for the RNG implementation. If the RNG is configurable, the evaluator shall perform 15 trials for each configuration.

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 shall generate eight input values for each trial. The first is a count (0 14). The next three are entropy input, nonce, and personalization string for the instantiate operation. The next two are additional input and entropy input for the first call to generate. The final two are additional input and entropy input for the second call to generate. These values are randomly generated. "generate one block of random bits" means to generate random bits with number of returned bits equal to the Output Block Length (as defined in NIST SP800-90A).

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 shall generate eight input values for each trial. The first is a count (0 14). The next three are entropy input, nonce, and personalization string for the instantiate operation. The fifth value is additional input to the first call to generate. The sixth and seventh are additional input and entropy input to the call to reseed. The final value is additional input to the second generate call.

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

Entropy input: the length of the entropy input value must equal the seed length. Nonce: If a nonce is supported (CTR_DRBG with no 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 <= 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 shall use personalization strings of two different lengths. If the implementation does not use a personalization string, no value needs to be supplied. Additional input: the additional input bit lengths have the same defaults and restrictions as the personalization string lengths.

Summary

This test is covered by CAVS-test. Please see Table 2 for a mapping to the CAVP certificates.

2.1.2.10 SSH Server Protocol (FCS_SSHS_EXT.1)

FCS_SSHS_EXT.1.2

TSS Assurance Activities

Assurance Activity AA-FCS_SSHS_EXT.1.2-ASE-01

[TD0631] The evaluator shall check to ensure that the TSS contains a list of supported public key algorithms that are accepted for client authentication and that this list is consistent with signature verification algorithms selected in FCS_COP.1/SigGen (e.g., accepting EC keys requires corresponding Elliptic Curve Digital Signature algorithm claims).

[TD0631] The evaluator shall confirm that the TSS includes the description of how the TOE establishes a user identity when an SSH client presents a public key or X.509v3 certificate. For example, the TOE could verify that the SSH client's presented public key matches one that is stored within the SSH server's authorized_keys file.

[TD0631] If password-based authentication method has been selected in the FCS_SSHS_EXT.1.2, then the evaluator shall confirm its role in the authentication process is described in the TSS.

Summary

Chapter 7 of [ST] contains the TSS. Section 7.2.5, *SSH* describes the SSH protocol.

The evaluator checked the TSS which states that SSH connections to the TOE's command line interface (administrative interface) are protected using SSH version 2, using ssh-rsa for public key authentication.

This list of public key algorithms is found to be consistent with the specification of FCS_SSHS_EXT.1.5 which also lists rsa-sha2-256, rsa-sha2-512.

Additionally, this section of the TSS states that administrators are authenticated locally by user name and password thus indicating that password-based authentication method is also supported by the TOE.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-FCS_SSHS_EXT.1.2-ATE-01

[TD0631] Test objective: The purpose of these tests is to verify server supports each claimed client authentication method.

Test 1: For each supported client public-key authentication algorithm, the evaluator shall configure a remote client to present a public key corresponding to that authentication method (e.g., 2048-bit RSA key when using ssh-rsa public key). The evaluator shall establish sufficient separate SSH connections with an appropriately configured remote non-TOE SSH client to demonstrate the use of all applicable public key algorithms. It is sufficient to observe the successful completion of the SSH Authentication Protocol to satisfy the intent of this test.

Test 2: The evaluator shall choose one client public key authentication algorithm supported by the TOE. The evaluator shall generate a new client key pair for that supported algorithm without configuring the TOE to recognize the associated public key for authentication. The evaluator shall use an SSH client to attempt to connect to the TOE with the new key pair and demonstrate that authentication fails.

Test 3: [Conditional] If password-based authentication method has been selected in the FCS_SSHS_EXT.1.2, the evaluator shall configure the TOE to accept password-based authentication and demonstrate that user authentication succeeds when the correct password is provided by the connecting SSH client.

Test 4: [Conditional] If password-based authentication method has been selected in the FCS_SSHS_EXT.1.2, the evaluator shall configure the TOE to accept password-based authentication and demonstrate that user authentication fails when the incorrect password is provided by the connecting SSH client.

Summary

Test 1:
The evaluator configured the TOE according to provided guidance for password based authentication. The evaluator tried to establish a connection with the TOE using password based authentication via SSH, GUI, IControl and IControl Rest and correct credentials.

Test 2:
The evaluator configured the TOE according to provided guidance for password based authentication. The evaluator tried to establish a connection with the TOE using password based authentication via SSH, GUI, IControl and IControl Rest and with incorrect credentials but failed (as expected).

FCS_SSHS_EXT.1.3

TSS Assurance Activities

Assurance Activity AA-FCS_SSHS_EXT.1.3-ASE-01

The evaluator shall check that the TSS describes how "large packets" in terms of RFC 4253 are detected and handled.

Summary

Chapter 7 of [ST] contains the TSS. Section 7.2.5 *SSH* describes the SSH protocol.

The evaluator examined the TSS which states that the SSH implementation monitors packet size on all channels and limits packet size as suggested in RFC 4253 Section 6.1; the maximum packet size is (256*1024) bytes with larger packets being silently dropped.

This information is consistent with FCS_SSHS_EXT.1.3 which specifies that packets greater than 256*1024 bytes in an SSH transport connection are dropped.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-FCS_SSHS_EXT.1.3-ATE-01

The evaluator shall demonstrate that if the TOE receives a packet larger than that specified in this component, that packet is dropped.

Summary

The evaluator used a computer running Ubuntu Linux. The evaluator established a SSH connection and issued some commands in order to generate traffic. The evaluator sent a packet from the TOE to the SSH server and the connection was successful.

The evaluator used a computer running Ubuntu Linux and installed a modified SSH server. The evaluator established a SSH connection using the modified SSH server in order to generate packets larger than that specified in this component. The evaluator sent a packet from the TOE to the SSH server and the reply packet is dropped (as expected).

FCS_SSHS_EXT.1.4

TSS Assurance Activities

Assurance Activity AA-FCS_SSHS_EXT.1.4-ASE-01

The evaluator shall check the description of the implementation of this protocol in the TSS to ensure that optional characteristics are specified, and the encryption algorithms supported are specified as well. The evaluator shall check the TSS to ensure that the encryption algorithms specified are identical to those listed for this component.

Summary

Chapter 7 of [ST] contains the TSS. Section 7.2.5 *SSH* describes the SSH protocol.

The evaluator checked the TSS which states that SSH connections to the TOE's command line interface are protected using SSH version 2, using AES CBC and CTR modes with 128-bit or 256-bit key for data encryption. The evaluator verified that the encryption algorithms are identical to those specified in FCS_SSHS_EXT.1.4 which are aes128-cbc, aes256-cbc, aes128, aes256-ctr.

Guidance Assurance Activities

Assurance Activity AA-FCS_SSHS_EXT.1.4-AGD-01

The evaluator shall also check the guidance documentation to ensure that it contains instructions on configuring the TOE so that SSH conforms to the description in the TSS (for instance, the set of algorithms advertised by the TOE may have to be restricted to meet the requirements).

Summary

[ST] section 6.2.2.10 defines FCS_SSHS_EXT.1.4 which states that the SSH transport implementation only uses aes128-cbc and aes256-cbc and rejects all other encryption algorithm. The TSS (section 7.2.5 "SSH" of [ST]) provides consistent information. The evaluator examined section 2.3.10.2 "SSL" of [ECG] and determined that it describes the following:

- The ccmode command and the default SSH server profile set the allowable ciphersuites for SSH. There is one additional changed required and the section reference section 2.2.3.2 "Updating the SSH cipher configuration" for that change. The section states that the following algorithms must be removed from the default configuration,
  - diffie-hellman-group14-sha1
  - ecdh-sha2-nistp521
  - hmac-sha2-512
  - aes128-ctr, aes 256-ctr
  - aes128-gmc, aes256-gmc

  and refers to [K80425458] on how to remove the ciphers as well as providing an example command.

Additionally, the evaluator examined section 5.2 "SSH Server Protocol" of [ECG] which specifies the encryption algorithms as AES128-CBC and AES256-CBC.

The evaluator determined that the guidance indicates that no additional action is necessary for selecting the encryption algorithms as they are determined during protocol handshake at the time of establishment.

The evaluator determined that the guidance description conforms to the description in the TSS.

Please note that the set of algorithms are restricted when the TOE is configured to the CC mode (ccmode command), which is an initial configuration step for using the TOE (section 2.3.1 "ccmode command" of [ECG]).

Test Assurance Activities

Assurance Activity AA-FCS_SSHS_EXT.1.4-ATE-01

The evaluator must ensure that only claimed ciphers and cryptographic primitives are used to establish an SSH connection. To verify this, the evaluator shall start session establishment for an SSH connection from a remote client (referred to as "remote endpoint" below). The evaluator shall capture the traffic exchanged between the TOE and the remote endpoint during protocol negotiation (e.g. using a packet capture tool or information provided by the endpoint, respectively). The evaluator shall verify from the captured traffic that the TOE offers all the ciphers defined in the TSS for the TOE for SSH sessions, but no additional ones compared to the definition in the TSS. The evaluator shall perform one successful negotiation of an SSH session to verify that the TOE behaves as expected. It is sufficient to observe the successful negotiation of the session to satisfy the intent of the test. If the evaluator detects that not all ciphers defined in the TSS for SSH are supported by the TOE and/or the TOE supports one or more additional ciphers not defined in the TSS for SSH, the test shall be regarded as failed.

Summary

Test 1:
The evaluator used a computer running Ubuntu Linux as a SSH client and configured it with all the supported encryption algorithms see [ST] 7.2.5 "SSH". The evaluator used Wireshark to record and analyze the traffic between the SSH client and the TOE. The analysis showed the corresponding encryption algorithm each time.

The evaluator used a computer running Ubuntu Linux as a SSH client and configured it with AES128-CTR encryption algorithm. The evaluator checked if the connection will get established with the TOE but failed (as expected).

FCS_SSHS_EXT.1.5

TSS Assurance Activities

Assurance Activity AA-FCS_SSHS_EXT.1.5-ASE-01

[TD0631] The evaluator shall check the description of the implementation of this protocol in the TSS to ensure that the SSH server's host public key algorithms supported are specified and that they are identical to those listed for this component.

Summary

Chapter 7 *TOE Summary Specification* of [ST] contains the TSS. Section 7.2.5 *SSH* describes the SSH protocol.

The evaluator checked the TSS which states that SSH connections to the TOE's command line interface are protected using SSH version 2, using ssh-rsa for public key authentication.

This list of public key algorithms is found to be consistent with FCS_SSHS_EXT.1.5.

The evaluator notes that the TSS description of SSH does not explicitly specify any optional characteristics.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

Assurance Activity AA-FCS_SSHS_EXT.1.5-ATE-01

[TD0631] Test objective: This test case is meant to validate that the TOE server will support host public keys of the claimed algorithm types.

Test 1: The evaluator shall configure (only if required by the TOE) the TOE to use each of the claimed host public key algorithms. The evaluator will then use an SSH client to confirm that the client can authenticate the TOE server public key using the claimed algorithm. It is sufficient to observe (on the wire) the successful negotiation of the algorithm to satisfy the intent of the test.

Has effectively been moved to FCS_SSHS_EXT.1.2.

Test objective: This negative test case is meant to validate that the TOE server does not support host public key algorithms that are not claimed.

Test 2: The evaluator shall configure a non-TOE SSH client to only allow it to authenticate an SSH server host public key algorithm that is not included in the ST selection. The evaluator shall attempt to establish an SSH connection from the non-TOE SSH client to the TOE SSH server and observe that the connection is rejected.

Summary

Test 1:
The evaluator used a computer running Ubuntu Linux as a SSH client and configured it with all the supported public key algorithms see [ST] 7.2.5 "SSH". The evaluator generated a new SSH key pair for that algorithm and configured the TOE for that key pair. The evaluator used Wireshark to record and analyze the traffic between the SSH client and the TOE. The analysis showed the corresponding public key algorithm each time.

Test 2:
The evaluator used a computer running Ubuntu Linux as a SSH client and configured it with all the supported public key algorithms see [ST] 7.2.5 "SSH". The evaluator generated a new SSH key pair for that algorithm. The evaluator did not configure the TOE for that key pair. The evaluator tried to establish key based authentication with the TOE for that SSH key pair but the connection failed.

Test 3:
The evaluator used a computer running Ubuntu Linux as a SSH client and configured it with SSH-DSA public key algorithm. The evaluator checked if the connection will established between the SSH client and the TOE but failed (as expected).

FCS_SSHS_EXT.1.6

TSS Assurance Activities

Assurance Activity AA-FCS_SSHS_EXT.1.6-ASE-01

The evaluator shall check the TSS to ensure that it lists the supported data integrity algorithms, and that the list corresponds to the list in this component.

Summary

Chapter 7of [ST] contains the TSS. Section 7.2.5 *SSH* describes the SSH protocol.

The evaluator checked the TSS which states that SSH connections to the TOE's command line interface are protected using SSH version 2, using transport data integrity protection hash algorithm HMAC-SHA1 and HMAC-SHA2-256. This list is found to be consistent with FCS_SSHS_EXT.1.6.

Guidance Assurance Activities

Assurance Activity AA-FCS_SSHS_EXT.1.6-AGD-01

The evaluator shall also check the guidance documentation to ensure that it contains instructions to the Security Administrator on how to ensure that only the allowed data integrity algorithms are used in SSH connections with the TOE (specifically, that the "none" MAC algorithm is not allowed).

Summary

[ST] section 6.2.2.10 defines FCS_SSHS_EXT.1.6 which states that the SSH transport implementation uses hmac-sha1 and hmac-sha2-256, for data integrity. The TSS (section 7.2.5 "SSH" of [ST]) provides consistent information. For related guidance, the evaluator identified section 2.3.4.1 "SSH" of [ECG] which refers to the user manual "K13454: Configuring SSH public key authentication on BIG-IP systems (11.x - 16.x)", [K13454] . The evaluator also identified section 2.2.3.2 "Updating the SSH cipher configuration" for instructions on removing unsupported algorithms from the default configuration which refers the user manual "K80425458: Modifying the list of ciphers and MAC algorithms used by the SSH service on the BIG-IP system or BIG-IQ system", [K80425458] . The evaluator examined [K13454] and verified that it contains sufficient information to configure SSH to conform with the TSS. Additionally, the evaluator examined [K80425458] along with specific instructions provided in section 2.2.3.2 "Updating the SSH cipher configuration" in [ECG] and verified that it contains sufficient instructions to disable the data integrity algorithms not allowed in the evaluated configuration e.g., hmac-sha2-512). Please note that the set of algorithms are restricted when the TOE is configured to the CC mode (ccmode command), which is an initial configuration step for using the TOE (section 2.3.1 "ccmode command" of [ECG]).

Test Assurance Activities

Assurance Activity AA-FCS_SSHS_EXT.1.6-ATE-01

Test 1: [conditional, if an HMAC or AEAD_AES_*_GCM algorithm is selected in the ST] The evaluator shall establish an SSH connection using each of the algorithms, except "implicit", specified by the requirement. It is sufficient to observe (on the wire) the successful negotiation of the algorithm to satisfy the intent of the test.

Note: To ensure the observed algorithm is used, the evaluator shall ensure a non-aes-gcm@openssh.com encryption algorithm is negotiated while performing this test.*

Test 2: [conditional, if an HMAC or AEAD_AES_*_GCM algorithm is selected in the ST] The evaluator shall configure an SSH client to only allow a MAC algorithm that is not included in the ST selection. The evaluator shall attempt to connect from the SSH client to the TOE and observe that the attempt fails.

Note: To ensure the proposed MAC algorithm is used, the evaluator shall ensure a non-aes-gcm@openssh.com encryption algorithm is negotiated while performing this test.*

Summary

Test 1:
The evaluator used a computer running Ubuntu Linux as a SSH client and configured it with all the supported integrity algorithms see [ST] 7.2.5 "SSH". The evaluator used Wireshark to record and analyze the traffic between the SSH client and the TOE. The analysis showed the corresponding integrity algorithm each time.

Test 2:
The evaluator used a computer running Ubuntu Linux as a SSH client and configured it with hmac-md5 MAC algorithm. The evaluator checked if the connection will get established between the SSH client and the TOE but it failed (as expected).

FCS_SSHS_EXT.1.7

TSS Assurance Activities

Assurance Activity AA-FCS_SSHS_EXT.1.7-ASE-01

The evaluator shall check the TSS to ensure that it lists the supported key exchange algorithms, and that the list corresponds to the list in this component.

Summary

Chapter 7 of [ST] contains the TSS. Section 7.2.5 *SSH* describes the SSH protocol and section 7.7 *Trusted Path/Channels* describes trusted path/channels.

The evaluator checked both sections of the TSS which state that the SSH implementation has hard-coded ecdh-sha2-nistp256 and ecdh-sha2-nistp384 for key exchange. This list is found to be consistent with FCS_SSHS_EXT.1.7.

Guidance Assurance Activities

Assurance Activity AA-FCS_SSHS_EXT.1.7-AGD-01

The evaluator shall also check the guidance documentation to ensure that it contains instructions to the Security Administrator on how to ensure that only the allowed key exchange algorithms are used in SSH connections with the TOE.

Summary

[ST] section 6.2.2.10 defines FCS_SSHS_EXT.1.7 which states that the SSH transport implementation only uses ecdh-sha2-nistp256 and ecdh-sha2-nistp384 as key exchange methods. The TSS (section 7.2.5 "SSH" of [ST]) provides consistent information. For related guidance, the evaluator identified section 2.3.4.1 "SSH" of [ECG] which refers to the user manual "K13454: Configuring SSH public key authentication on BIG-IP systems (11.x - 16.x)", [K13454] for setting up SSH public key authentication. The evaluator also identified section 2.2.3.2 "Updating the SSH cipher configuration" which refers to the user manual "K80425458: Modifying the list of ciphers and MAC algorithms used by the SSH service on the BIG-IP system or BIG-IQ system", [K80425458] for guidance on how to remove unsupported algorithms for key exchange from default configuration. The evaluator examined [K13454] and verified that it contains sufficient information to configure SSH to conform with the TSS. Additionally, the evaluator examined [K80425458] along with specific instructions provided in section 2.2.3.2 "Updating the SSH cipher configuration" in [ECG] and verified that it contains sufficient instructions to disable the key exchange algorithms not allowed in the evaluated configuration (i.e.., diffie-hellman-group14-sha1 and ecdh-sha2-nistp521). Please note that the set of algorithms are restricted when the TOE is configured to the CC mode (ccmode command), which is an initial configuration step for using the TOE (section 2.3.1 "ccmode command" of [ECG]).

Test Assurance Activities

Assurance Activity AA-FCS_SSHS_EXT.1.7-ATE-01

Test 1: The evaluator shall configure an SSH client to only allow the diffie-hellman-group1-sha1 key exchange. The evaluator shall attempt to connect from the SSH client to the TOE and observe that the attempt fails.

Test 2: For each allowed key exchange method, the evaluator shall configure an SSH client to only allow that method for key exchange, attempt to connect from the client to the TOE, and observe that the attempt succeeds.

Summary

Test 1:
The evaluator used a computer running Ubuntu Linux as a SSH client and configured it with a diffie-hellman-group1-sha1 Key Exchange method. The evaluator used Wireshark to record and analyze the traffic between the SSH client and the TOE. The analysis showed the connection rejection (as expected).

Test 2 :
The evaluator used a computer running Ubuntu Linux as a SSH client and configured it with all the supported Key Exchange methods. The evaluator used Wireshark to record and analyze the traffic between the SSH client and the TOE. The analysis showed the corresponding public key algorithm each time.

FCS_SSHS_EXT.1.8

TSS Assurance Activities

Assurance Activity AA-FCS_SSHS_EXT.1.8-ASE-01

The evaluator shall check that the TSS specifies the following:

a) Both thresholds are checked by the TOE.

b) Rekeying is performed upon reaching the threshold that is hit first.

Summary

Chapter 7 of [ST] contains the TSS. Section 7.2.5 *SSH* describes the SSH protocol.

The evaluator checked the TSS which states the following:

> *The SSH connection session key will be renegotiated after either of two thresholds has been reached. SSH connection session keys will be renegotiated after one hour of use. In addition, the SSH connection session key will be renegotiated after an administrator-configured maximum amount of data, the RekeyLimit, is transmitted over the connection. The administrative guidance will instruct the user to not set the RekeyLimit to a value greater that 1 GB.*

Guidance Assurance Activities

Assurance Activity AA-FCS_SSHS_EXT.1.8-AGD-01

If one or more thresholds that are checked by the TOE to fulfil the SFR are configurable, then the evaluator shall check that the guidance documentation describes how to configure those thresholds. Either the allowed values are specified in the guidance documentation and must not exceed the limits specified in the SFR (one hour of session time, one gigabyte of transmitted traffic) or the TOE must not accept values beyond the limits specified in the SFR. The evaluator shall check that the guidance documentation describes that the TOE reacts to the first threshold reached.

Summary

[ST] section 6.2.2.10 defines FCS_SSHS_EXT.1.8 which states the following:

- The TSF shall ensure that within SSH connections the same keys are used for a threshold of no longer than one hour, and no more than one gigabyte of transmitted data. After either of the thresholds are reached a rekey needs to be performed.

The evaluator examined section 2.3.10.2 "SSH" of [ECG] which states the following:

- The default rekey limit set in the SSH configuration file provided with the BIG-IP ensures that not more than 2^28 packets are transmitted or 1 hour passes before the session keys are rekeyed.
- Session key rekeying will occur when the first of these thresholds is reached.

The evaluator determined that the guidance provides the necessary documentation regarding how the TOE reacts to the first threshold reached.

Test Assurance Activities

Assurance Activity AA-FCS_SSHS_EXT.1.8-ATE-01

The evaluator needs to perform testing that rekeying is performed according to the description in the TSS. The evaluator shall test both, the time-based threshold and the traffic-based threshold.

For testing of the time-based threshold, the evaluator shall use an SSH client to connect to the TOE and keep the session open until the threshold is reached. The evaluator shall verify that the SSH session has been active longer than the threshold value and shall verify that the TOE initiated a rekey (the method of verification shall be reported by the evaluator).

Testing does not necessarily have to be performed with the threshold configured at the maximum allowed value of one hour of session time, but the value used for testing shall not exceed one hour. The evaluator needs to ensure that the rekeying has been initiated by the TOE and not by the SSH client that is connected to the TOE.

For testing of the traffic-based threshold the evaluator shall use the TOE to connect to an SSH client and shall transmit data to and/or receive data from the TOE within the active SSH session until the threshold for data protected by either encryption key is reached. It is acceptable if the rekey occurs before the threshold is reached (e.g. because the traffic is counted according to one of the alternatives given in the Application Note for FCS_SSHS_EXT.1.8).

The evaluator shall verify that more data has been transmitted within the SSH session than the threshold allows and shall verify that the TOE initiated a rekey (the method of verification shall be reported by the evaluator).

Testing does not necessarily have to be performed with the threshold configured at the maximum allowed value of one gigabyte of transferred traffic, but the value used for testing shall not exceed one gigabyte. The evaluator needs to ensure that the rekeying has been initiated by the TOE and not by the SSH client that is connected to the TOE.

If one or more thresholds that are checked by the TOE to fulfil the SFR are configurable, the evaluator needs to verify that the threshold(s) can be configured as described in the guidance documentation and the evaluator needs to test that modification of the thresholds is restricted to Security Administrators (as required by FMT_MOF.1/Functions).

In cases where data transfer threshold could not be reached due to hardware limitations it is acceptable to omit testing of this (SSH rekeying based on data transfer threshold) threshold if both the following conditions are met:

- *An argument is present in the TSS section describing this hardware-based limitation and*
- *All hardware components that are the basis of such argument are definitively identified in the ST. For example, if specific Ethernet Controller or WiFi radio chip is the root cause of such limitation, these chips must be identified.*

Summary

The evaluator used a computer running Ubuntu Linux as a SSH client, configured it to transmit to the TOE via sftp more than the allowed file size until the connection reached the transmission limit. The evaluator used a computer running Ubuntu Linux as a SSH client, configured it to establish a connection with the TOE until the connection reached the time limit.

2.1.2.11 Extended: TLS Client Protocol without mutual authentication (FCS_TLSC_EXT.1)

FCS_TLSC_EXT.1.1

TSS Assurance Activities

Assurance Activity AA-FCS_TLSC_EXT.1.1-ASE-01

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

Summary

Please note this analysis covers the following (iterated) SFRs defined in [ST] :

- FCS_TLSC_EXT.1[1]
- FCS_TLSC_EXT.1[2]

Chapter 7 of [ST] contains the TSS. Section 7.2.6 *TLS Protocol* describes the TLS protocol.

Table 9 "Cipher Suites" of this section lists all the supported ciphersuites for TLS v1.1 and v1.2 connections. The ciphersuites are reproduced below:

- TLS_RSA_WITH_AES_128_CBC_SHA
- TLS_RSA_WITH_AES_256_CBC_SHA
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA
- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA
- TLS_RSA_WITH_AES_128_CBC_SHA256
- TLS_RSA_WITH_AES_256_CBC_SHA256

- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256
- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384
- TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384
- TLS_RSA_WITH_AES_128_GCM_SHA256
- TLS_RSA_WITH_AES_256_GCM_SHA384

The evaluator compared the above ciphersuites list with the list specified in FCS_TLSC_EXT.1.1[1]-[2] of [ST] and found them to be consistent.

Guidance Assurance Activities

Assurance Activity AA-FCS_TLSC_EXT.1.1-AGD-01

> *The evaluator shall check the guidance documentation to ensure that it contains instructions on configuring the TOE so that TLS conforms to the description in the TSS.*

Summary

Section 6.2.2.12 and 6.2.2.13 of [ST] define FCS_TLSC_EXT.1.1[1] and FCS_TLSC_EXT.1.1[2], respectively, which specify the TLS ciphersuites supported by the (data plane) part of the TOE. The TSS (section 7.2.6 "TLS Protocol") of [ST] provides consistent information. The evaluator examined section 5.1 "TLS" of [ECG] which lists identical sets of allowable ciphersuites for TLS v1.1 and TLS v1.2. The evaluator examined section 2.3.10.1 "SSL Profiles" of [ECG] which provides the following guidance:

- The ccmode command sets the allowable ciphersuites for the default client and server SSL profiles: clientssl and serverssl.
- Create and use SSL profiles based only off those default profiles, and not modify the configured ciphersuites, in order to ensure that your TLS connections are Common-Criteria-compliant.
- Do not use the clientssl-insecure-compatible and serverssl-insecure-compatible default profiles, as these include weak TLS ciphers which are not Common-Criteria-compliant.
- Use only 2048-bit or higher RSA key sizes, or ECDSA curves p-256 or p-384 for SSL profiles.

The evaluator determined that [ECG] indicates no specific configuration required for TLS clients and that the TLS version and the ciphersuite selection are determined during protocol negotiation handshake at the time of session establishment. Please note that the set of algorithms are restricted when the TOE is configured to the CC mode (ccmode command), which is an initial configuration step for using the TOE as per section 2.3.1 "ccmode command" of [ECG] .

Test Assurance Activities

Assurance Activity AA-FCS_TLSC_EXT.1.1-ATE-01

Test 1: The evaluator shall establish a TLS connection using each of the ciphersuites 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 HTTPS session. It is sufficient to observe the successful negotiation of a ciphersuite to satisfy the intent of the test; it is not necessary to examine the characteristics of the encrypted traffic to discern the ciphersuite being used (for example, that the cryptographic algorithm is 128-bit AES and not 256-bit AES).

Test 2: The evaluator shall 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 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 shall send a server certificate in the TLS connection that does not match the server-selected ciphersuite (for example, send an ECDSA certificate while using the TLS_RSA_WITH_AES_128_CBC_SHA ciphersuite). The evaluator shall verify that the TOE disconnects after receiving the server's Certificate handshake message.

Test 4: The evaluator shall perform the following 'negative tests':

a) The evaluator shall configure the server to select the TLS_NULL_WITH_NULL_NULL ciphersuite and verify that the client denies the connection.

b) Modify the server's selected ciphersuite in the Server Hello handshake message to be a ciphersuite not presented in the Client Hello handshake message. The evaluator shall verify that the client rejects the connection after receiving the Server Hello.

c) [conditional]: If the TOE presents the Supported Elliptic Curves/Supported Groups Extension the evaluator shall configure the server to perform an ECDHE or DHE key exchange in the TLS connection using a non-supported curve/group (for example P-192) and shall verify that the TOE disconnects after receiving the server's Key Exchange handshake message.

Test 5: The evaluator performs the following modifications to the traffic:

a) Change the TLS version selected by the server in the Server Hello to a non-supported TLS version and verify that the client rejects the connection.

b) [conditional]: If using DHE or ECDH, modify the signature block in the Server's Key Exchange handshake message, and verify that the handshake does not finished successfully, and no application data flows. This test does not apply to cipher suites using RSA key exchange. If a TOE only supports RSA key exchange in conjunction with TLS, then this test shall be omitted.

Test 6: The evaluator performs the following 'scrambled message tests':

a) Modify a byte in the Server Finished handshake message and verify that the handshake does not finish successfully and no application data flows.

b) Send a garbled message from the server after the server has issued the ChangeCipherSpec message and verify that the handshake does not finish successfully and no application data flows.

c) 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 ciphersuite) or that the server denies the client's Finished handshake message.

Summary

Test 1:
The evaluator used a computer running Kali Linux. The evaluator created a CA certificate and server certificate for s_server and a client certificate for the TOE and installed the certificates accordingly. The evaluator configured the TOE to act as a TLS client using a Virtual Server and Client SSL profile, as per [K14783]. The evaluator also used tcpdump to capture and analyse the traffic. For each supported cipher, the evaluator configured s_server and initiated a connection from the TOE. The evaluator verified that the captured traffic showed the corresponding cipher suite and that the connection was successful.

Test 2:
The evaluator used a computer running Kali Linux. The evaluator created a CA certificate and server certificate for s_server that contains the Server Authentication purpose in the extendedKeyUsage field and a client certificate for the TOE and installed the certificates accordingly. The evaluator configured the TOE to act as a TLS client using a Virtual Server and Client SSL profile, as per [K14783]. The evaluator also used tcpdump to capture and analyse the traffic. The evaluator

verified that a connection from the TOE was successful. Then the evaluator created a server certificate without server authentication purpose for s_server and installed the certificate accordingly. The evaluator checked if the connection would be established, but it failed (as expected).

Test 3:
The evaluator used a computer running Kali Linux. The evaluator created a CA certificate and server certificate for s_server and a client certificate for the TOE and installed the certificates accordingly. The evaluator configured the TOE to act as a TLS client using a Virtual Server and Client SSL profile, as per [K14783]. The evaluator also used tcpdump to capture and analyse the traffic. The TLS server was set up with a RSA cipher suite and an ECDSA certificate. The evaluator checked if the connection would be established, but it failed (as expected).

Test 4a):
The evaluator used a computer running Kali Linux. The evaluator created a CA certificate and server certificate for s_server and a client certificate for the TOE and installed the certificates accordingly. The evaluator configured the TOE to act as a TLS client using a Virtual Server and Client SSL profile, as per [K14783]. The evaluator also used tcpdump to capture and analyse the traffic. The TLS server was configured to only allow the TLS_NULL_WITH_NULL_NULL cipher suite. The evaluator checked if the connection would be established, but it failed (as expected).

Test 4b):
The evaluator used a computer running Kali Linux. The evaluator created a CA certificate and server certificate for s_server and a client certificate for the TOE and installed the certificates accordingly. The evaluator configured the TOE to act as a TLS client using a Virtual Server and Client SSL profile, as per [K14783]. The evaluator also used tcpdump to capture and analyse the traffic. The evaluator used a proxy that intercepts the communication and replies back to the TOE with server's selected ciphersuite in the Server Hello handshake message to be a ciphersuite not presented in the Client Hello handshake message. The evaluator checked if the connection would be established, but it failed (as expected).

Test 4c):
The evaluator used a computer running Linux Kali Linux. The evaluator created a CA certificate, a server certificate for openssl s_server and a client certificate for the TOE and installed the certificates accordingly. The evaluator configured the openssl s_server with an ECDHE certificate with non supported curve. The evaluator configured the TOE to act as a TLS client using a Virtual Server and Client SSL profile, as per [K14783]. The evaluator also used tcpdump to capture and analyse the traffic. The analysis showed that the connection could not be established (as expected).

Test 5a):
The evaluator used a computer running Kali Linux. The evaluator created a CA certificate and server certificate for s_server and a client certificate for the TOE and installed the certificates accordingly. The evaluator configured the TOE to act as a TLS client using a Virtual Server and Client SSL profile, as per [K14783]. The evaluator also used tcpdump to capture and analyse the traffic. The evaluator used a proxy that intercepts the communication and replies back to the TOE with a non supported TLS version. The evaluator checked if the connection would be established, but it failed (as expected).

Test 5b):
The evaluator used a computer running Kali Linux. The evaluator created a CA certificate and server certificate for s_server and a client certificate for the TOE and installed the certificates accordingly. The evaluator configured the TOE to act as a TLS client using a Virtual Server and Client SSL profile, as per [K14783]. The evaluator also used tcpdump to capture and analyse the traffic. The evaluator used a proxy that intercepts the communication and replies back to the TOE with modified signature block in the Server's Key Exchange handshake message. The evaluator checked if the connection would be established, but it failed (as expected).

Test 6a):

The evaluator used a computer running Kali Linux. The evaluator created a CA certificate and server certificate for s_server and a client certificate for the TOE and installed the certificates accordingly. The evaluator configured the TOE to act as a TLS client using a Virtual Server and Client SSL profile, as per [K14783]. The evaluator also used tcpdump to capture and analyse the traffic. The evaluator used a proxy that intercepts the communication and replies back to the TOE with modified Finished handshake message. The evaluator checked if the connection would be established, but it failed (as expected).

Test 6b):
The evaluator used a computer running Kali Linux. The evaluator created a CA certificate and server certificate for s_server and a client certificate for the TOE and installed the certificates accordingly. The evaluator configured the TOE to act as a TLS client using a Virtual Server and Client SSL profile, as per [K14783]. The evaluator also used tcpdump to capture and analyse the traffic. The evaluator used a proxy that intercepts the communication and replies back to the TOE with a garbled message after issuing the ChangeCipherSpec message. The evaluator checked if the connection would be established, but it failed (as expected).

Test 6c):
The evaluator used a computer running Kali Linux. The evaluator created a CA certificate and server certificate for s_server and a client certificate for the TOE and installed the certificates accordingly. The evaluator configured the TOE to act as a TLS client using a Virtual Server and Client SSL profile, as per [K14783]. The evaluator also used tcpdump to capture and analyse the traffic.The evaluator used a proxy that intercepts the communication and replies back to the TOE with a modified server's nonce in the Server Hello handshake message. The evaluator checked if the connection would be established, but it failed (as expected).

FCS_TLSC_EXT.1.2

TSS Assurance Activities

Assurance Activity AA-FCS_TLSC_EXT.1.2-ASE-01

The evaluator shall ensure that the TSS describes the client's method of establishing all reference identifiers from the administrator/application-configured reference identifier, including which types of reference identifiers are supported (e.g. application-specific Subject Alternative Names) and whether IP addresses and wildcards are supported.

Note that where a TLS channel is being used between components of a distributed TOE for FPT_ITT.1, the requirements to have the reference identifier established by the user are relaxed and the identifier may also be established through a "Gatekeeper" discovery process. The TSS should describe the discovery process and highlight how the reference identifier is supplied to the "joining" component. Where the secure channel is being used between components of a distributed TOE for FPT_ITT.1 and the ST author selected attributes from RFC 5280, the evaluator shall ensure the TSS describes which attribute type, or combination of attributes types, are used by the client to match the presented identifier with the configured identifier. The evaluator shall ensure the TSS presents an argument how the attribute type, or combination of attribute types, uniquely identify the remote TOE component; and the evaluator shall verify the attribute type, or combination of attribute types, is sufficient to support unique identification of the maximum supported number of TOE components.

If IP addresses are supported in the CN as reference identifiers, the evaluator shall ensure that the TSS describes the TOE's conversion of the text representation of the IP address in the CN to a binary representation of the IP address in network byte order. The evaluator shall also ensure that the TSS describes whether canonical format (RFC 5952 for IPv6, RFC 3986 for IPv4) is enforced.

Summary

Please note this analysis covers the following (iterated) SFRs defined in [ST] :

- FCS_TLSC_EXT.1[1]
- FCS_TLSC_EXT.1[2]

Chapter 7 of [ST] contains the TSS. Section 7.2.6 *TLS Protocol* describes the TLS protocol. It states the following:

- For the TOE acting as TLS client, the TOE checks Common Name (CN) and DNS name. The CN or SAN in the certificate is compared by requiring an exact string match of the authenticate name against the IPv4 address in the certificate. The reference identifiers do not need to be converted by the TOE to perform this comparison.
- The BIG-IP TLS client supports ECDH in the Client Hello by default for the dataplane portion of the TOE. This can optionally be disabled by removing the corresponding cipher suites, although individual curves cannot be configured.
- Use of wildcards for reference identifiers constructed by the TOE and certificate pinning for TLS client connections are not supported by the TOE.

The evaluator noted that the ST does not claim FPT_ITT.1.

Guidance Assurance Activities

Assurance Activity AA-FCS_TLSC_EXT.1.2-AGD-01

The evaluator shall ensure that the operational guidance describes all supported identifiers, explicitly states whether the TOE supports the SAN extension or not and includes detailed instructions on how to configure the reference identifier(s) used to check the identity of peer(s). If the identifier scheme implemented by the TOE includes support for IP addresses, the evaluator shall ensure that the operational guidance provides a set of warnings and/or CA policy recommendations that would result in secure TOE use.

Where the secure channel is being used between components of a distributed TOE for FPT_ITT.1, the SFR selects attributes from RFC 5280, and FCO_CPC_EXT.1.2 selects "no channel"; the evaluator shall verify the guidance provides instructions for establishing unique reference identifiers based on RFC5280 attributes.

Summary

The TSS (section 7.2.6 "TLS Protocol" in [ST]) describes that the TOE uses the DNS names included in the Subject Alternative Name (SAN) field as the reference identifier for the server certificate. The evaluator examined the guidance for the control plane provided section 2.3.10.1 "SSL Profiles" of [ECG] and determined that there is no specific configuration required for TLS clients with regard to setting the reference identifier to be used for the purposes of certificate validation in TLS. Please note that the set of algorithms are restricted when the TOE is configured to the CC mode (ccmode command), which is an initial configuration step for using the TOE as per section 2.3.1 "ccmode command" of [ECG] .

Section 2.3.8 "Event (audit) logging" states the following:

> *Note that when configuring the Server SSL profile for the connection path to the syslog server, the Authenticate Name can be configured as an IPv4 address. The TOE supports both CN and the SAN extension. Refer to the description of "Authenticate Name" in [K14806] K14806: Overview of the Server SSL profile (11.x – 17.x) for details on configuring that option*

Test Assurance Activities

Assurance Activity AA-FCS_TLSC_EXT.1.2-ATE-01

Note that the following tests are marked conditional and are applicable under the following conditions:

a) For TLS-based trusted channel communications according to FTP_ITC.1 where RFC 6125 is selected, tests 1-6 are applicable.

or

b) For TLS-based trusted path communications according to FTP_TRP where RFC 6125 is selected, tests 1-6 are applicable

or

c) For TLS-based trusted path communications according to FPT_ITT.1 where RFC 6125 is selected, tests 1-6 are applicable. Where RFC 5280 is selected, only test 7 is applicable.

Note that for some tests additional conditions apply.

IP addresses are binary values that must be converted to a textual representation when presented in the CN of a certificate. When testing IP addresses in the CN, the evaluator shall follow the following formatting rules:

- *IPv4: The CN contains a single address that is represented a 32-bit numeric address (IPv4) is written in decimal as four numbers that range from 0-255 separated by periods as specified in RFC 3986.*
- *IPv6: The CN contains a single IPv6 address that is represented as eight colon separated groups of four lowercase hexadecimal digits, each group representing 16 bits as specified in RFC 4291. Note: Shortened addresses, suppressed zeros, and embedded IPv4 addresses are not tested.*

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

a) Test 1 [conditional]: The evaluator shall present a server certificate that contains a CN that does not match the reference identifier and does not contain the SAN extension. The evaluator shall verify that the connection fails. The evaluator shall repeat this test for each identifier type (e.g. IPv4, IPv6, FQDN) supported in the CN. When testing IPv4 or IPv6 addresses, the evaluator shall modify a single decimal or hexadecimal digit in the CN. Remark: Some systems might require the presence of the SAN extension. In this case the connection would still fail but for the reason of the missing SAN extension instead of the mismatch of CN and reference identifier. Both reasons are acceptable to pass Test 1.

b) Test 2 [conditional]: The evaluator shall present a server certificate that contains a CN that matches the reference identifier, contains the SAN extension, but does not contain an identifier in the SAN that matches the reference identifier. The evaluator shall verify that the connection fails. The evaluator shall repeat this test for each supported SAN type (e.g. IPv4, IPv6, FQDN, URI). When testing IPv4 or IPv6 addresses, the evaluator shall modify a single decimal or hexadecimal digit in the SAN.

c) Test 3 [conditional]: If the TOE does not mandate the presence of the SAN extension, the evaluator shall present a server certificate that contains a CN that matches the reference identifier and does not contain the SAN extension. The evaluator shall verify that the connection succeeds. The evaluator shall repeat this test for each identifier type (e.g. IPv4, IPv6, FQDN) supported in the CN. If the TOE does mandate the presence of the SAN extension, this Test shall be omitted.

d) Test 4 [conditional]: The evaluator shall present a server certificate that contains a CN that does not match the reference identifier but does contain an identifier in the SAN that matches. The evaluator shall verify that the connection succeeds. The evaluator shall repeat this test for each supported SAN type (e.g. IPv4, IPv6, FQDN, SRV).

e) Test 5 [conditional]: The evaluator shall perform the following wildcard tests with each supported type of reference identifier that includes a DNS name (i.e. CN-ID with DNS, DNS-ID, SRV-ID, URIID):

1) [conditional]: The evaluator shall present a server certificate containing a wildcard that is not in the left-most label of the presented identifier (e.g. foo..example.com) and verify that the connection fails.*

*2) [conditional]: The evaluator shall present a server certificate containing a wildcard in the left-most label (e.g. *.example.com). The evaluator shall configure the reference identifier with a single left-most label (e.g. foo.example.com) and verify that the connection succeeds, if wildcards are supported, or fails if wildcards are not supported. The evaluator shall configure the reference identifier without a left-most label as in the certificate (e.g. example.com) and verify that the connection fails. The evaluator shall configure the reference identifier with two left-most labels (e.g. bar.foo.example.com) and verify that the connection fails. (Remark: Support for wildcards was always intended to be optional. It is sufficient to state that the TOE does not support wildcards and observe rejected connection attempts to satisfy corresponding assurance activities.)*

f) [TD0634] Objective: The objective of this test is to ensure the TOE is able to differentiate between IP address identifiers that are not allowed to contain wildcards and other types of identifiers that may contain wildcards. Test 6: [conditional] If IP address identifiers supported in the SAN or CN, the evaluator shall present a server certificate that contains a CN that matches the reference identifier, except one of the groups has been replaced with a wildcard asterisk (*) (e.g. CN=.168.0.1 when connecting to 192.168.0.1, CN=2001:0DB8:0000:0000:0008:0800:200C:* when connecting to 2001:0DB8:0000:0000:0008:0800:200C:417A). The certificate shall not contain the SAN extension. The evaluator shall verify that the connection fails. The evaluator shall repeat this test for each supported IP address version (e.g. IPv4, IPv6).*

Remark: Some systems might require the presence of the SAN extension. In this case the connection would still fail but for the reason of the missing SAN extension instead of the mismatch of CN and reference identifier. Both reasons are acceptable to pass Test 6.

g) Test 7 [conditional]: If the secure channel is used for FPT_ITT, and RFC 5280 is selected, the evaluator shall perform the following tests. Note, when multiple attribute types are selected in the SFR (e.g. when multiple attribute types are combined to form the unique identifier), the evaluator modifies each attribute type in accordance with the matching criteria described in the TSS (e.g. creating a mismatch of one attribute type at a time while other attribute types contain values that will match a portion of the reference identifier):

1) The evaluator shall present a server certificate that does not contain an identifier in the Subject (DN) attribute type(s) that matches the reference identifier. The evaluator shall verify that the connection fails.

2) The evaluator shall present a server certificate that contains a valid identifier as an attribute type other than the expected attribute type (e.g. if the TOE is configured to expect id-atserialNumber=correct_identifier, the certificate could instead include id-at-name=correct_identifier), and does not contain the SAN extension. The evaluator shall verify that the connection fails. Remark: Some systems might require the presence of the SAN extension. In this case the connection would still fail but for the reason of the missing SAN extension instead of the mismatch of CN and reference identifier. Both reasons are acceptable to pass this test.

3) The evaluator shall present a server certificate that contains a Subject attribute type that matches the reference identifier and does not contain the SAN extension. The evaluator shall verify that the connection succeeds.

4) The evaluator shall confirm that all use of wildcards results in connection failure regardless of whether the wildcards are used in the left or right side of the presented identifier. (Remark: Use of wildcards is not addressed within RFC 5280.)

Summary

Test 1:
The evaluator used a computer running Kali Linux. The evaluator created a CA certificate, a server certificate that contains a CN that does not match the reference identifier and without Subject Alternative Name (SAN) extension for s_server server and a client certificate for the TOE and installed the certificates accordingly. The evaluator configured the TOE to act as a TLS client using a Virtual Server and Client SSL profile, as per [K14783]. OpenSSL s_server was used as the TLS server. The evaluator also used tcpdump to capture and analyse the traffic. The evaluator checked if the connection would be established, but it failed (as expected).

Test 2:
The evaluator used a computer running Kali Linux. The evaluator created a CA certificate, a server certificate that contains a CN and matches the reference identifier, contains an SAN extension, but does not match the r eference identifier in the SAN and a client certificate for the TOE and installed the certificates accordingly. The evaluator configured the TOE to act as a TLS client using a Virtual Server and Client SSL profile, as per [K14783]. OpenSSL s_server was used as the TLS server. The evaluator also used tcpdump to capture and analyse the traffic. The evaluator checked if the connection would be established, but it failed (as expected).

Test 3:
The evaluator used a computer running Kali Linux. The evaluator created a CA certificate, a server certificate that contains a CN that matches the reference identifier and does not contain the SAN extension for s_server server and a client certificate for the TOE and installed the certificates accordingly. The evaluator configured the TOE to act as a TLS client using a Virtual Server and Client SSL profile, as per [K14783]. OpenSSL s_server was used as the TLS server. The evaluator also used tcpdump to capture and analyse the traffic. The evaluator checked if the connection would be established and it succeeded (as expected).

Test 4:
The evaluator used a computer running Kali Linux. The evaluator created a CA certificate, a server certificate that contains a CN that matches the reference identifier and a SAN extension that matches the reference identifier for s_server server and a client certificate for the TOE and installed the

certificates accordingly. The evaluator configured the TOE to act as a TLS client using a Virtual Server and Client SSL profile, as per [K14783]. OpenSSL s_server was used as the TLS server. The evaluator also used tcpdump to capture and analyse the traffic. The evaluator checked if the connection would be established and it succeeded (as expected).

Test 5:
[ST] 7.2.6 "TLS Protocol" states that the use of wildcards for reference identifiers constructed by the TOE and certificate pinning for TLS client connections are not supported by the TOE. The evaluator therefore determines that this requirement not is applicable.

Test 6:
The evaluator used a computer running Kali Linux. The evaluator created a CA certificate, a server certificate that contains a CN that matches the reference identifier, but exchanged one group with the wildcard asterisk. The evaluator configured the TOE to act as a TLS client using a Virtual Server and Client SSL profile which validated the common name, as per [K14783]. OpenSSL s_server was used as the TLS server. The evaluator also used tcpdump to capture and analyse the traffic. The evaluator checked if the connection would fail (as expected).

Test 7:
[ST] 7.2.6 "TLS Protocol" states that the use of wildcards for reference identifiers constructed by the TOE and certificate pinning for TLS client connections are not supported by the TOE. The evaluator therefore determines that this requirement is not applicable.

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 shall demonstrate that using an invalid certificate results in the function failing as follows:

Test 1: Using the administrative guidance, the evaluator shall load a CA certificate or certificates needed to validate the presented certificate used to authenticate an external entity and demonstrate that the function succeeds, and a trusted channel can be established.

Test 2: The evaluator shall then change the presented certificate(s) so that validation fails and show that the certificate is not automatically accepted. The evaluator shall repeat this test to cover the selected types of failure defined in the SFR (i.e. the selected ones from failed matching of the reference identifier, failed validation of the certificate path, failed validation of the expiration date, failed determination of the revocation status). The evaluator performs the action indicated in the SFR selection observing the TSF resulting in the expected state for the trusted channel (e.g. trusted channel was established) covering the types of failure for which an override mechanism is defined.

Test 3 [conditional]: The purpose of this test to verify that only selected certificate validation failures could be administratively overridden. If any override mechanism is defined for failed certificate validation, the evaluator shall configure a new presented certificate that does not contain a valid entry in one of the mandatory fields or parameters (e.g. inappropriate value in extendedKeyUsage field) but is otherwise valid and signed by a trusted CA. The evaluator shall confirm that the certificate validation fails (i.e. certificate is rejected), and there is no administrative override available to accept such certificate.

Summary

Test 1:

The evaluator used a computer running Kali Linux. The evaluator created a root CA certificate, two intermediate CAs, a server certificate for openssl s_server server and a client certificate for the TOE and installed the certificates accordingly. The evaluator configured the TOE to act as a TLS client using a Virtual Server and Client SSL profile, as per [K14783]. The evaluator used Wireshark to record and analyze the traffic between the s_server and the TOE. The analysis showed that the connection could be established (as expected).

Test 2:
This test is not applicable. The TOE does not have any failure conditions defined for administrator override in the ST. Please refer to the tests for FIA_X509_EXT.1/Rev for testing of invalid certificates.

Test 3:
This test is not applicable since the TOE does not implement any administrator override mechanism.

FCS_TLSC_EXT.1.4

TSS Assurance Activities

Assurance Activity AA-FCS_TLSC_EXT.1.4-ASE-01

The evaluator shall verify that TSS describes the Supported Elliptic Curves/Supported Groups Extension and whether the required behaviour is performed by default or may be configured.

Summary

Please note this analysis covers the following (iterated) SFRs defined in [ST] :

- FCS_TLSC_EXT.1[1]
- FCS_TLSC_EXT.1[2]

Chapter 7 of [ST] contains the TSS. Section 7.2.6 *TLS Protocol* describes the TLS protocol. It states the following:

- The BIG-IP TLS client supports ECDH in the Client Hello by default. This can optionally be disabled by removing the corresponding cipher suites, although individual curves cannot be configured.

Guidance Assurance Activities

Assurance Activity AA-FCS_TLSC_EXT.1.4-AGD-01

If the TSS indicates that the Supported Elliptic Curves/Supported Groups Extension must be configured to meet the requirement, the evaluator shall verify that AGD guidance includes configuration of the Supported Elliptic Curves/Supported Groups Extension.

Summary

Sections 6.2.2.12 and 6.2.2.13 of [ST] define FCS_TLSC_EXT.1.4[1] and FCS_TLSC_EXT.1.4[2], respectively, which state that the TSF supports the Elliptic Curves Extensions secp256r1 and secp384r1. The TSS (section 7.2.6 of [ST]) provides consistent information. The TSS also states that the support for Elliptic Curve Extensions is provided by the TOE (data plane only) by default. The evaluator examined the guidance for the data plane provided in 2.3.10.1 "SSL Profiles" of [ECG] , and determined that there is no specific configuration required for TLS clients and that the TLS version and the ciphersuite selection is determined during protocol negotiation handshake at the time of session establishment. Please note that the set of algorithms are restricted when the TOE is configured to the CC mode (ccmode command), which is an initial configuration step for using the TOE as per section 2.3.1 "ccmode command" of [ECG] .

The evaluator determined that the guidance description conforms to the description in the TSS.

Test Assurance Activities

Assurance Activity AA-FCS_TLSC_EXT.1.4-ATE-01

> *Test 1 [conditional]: If the TOE presents the Supported Elliptic Curves/Supported Groups Extension, the evaluator shall configure the server to perform ECDHE or DHE (as applicable) key exchange using each of the TOE's supported curves and/or groups. The evaluator shall verify that the TOE successfully connects to the server.*

Summary

The evaluator used a computer running Linux Kali Linux. The evaluator created a CA certificate, a server certificate for openssl s_server and a client certificate for the TOE and installed the certificates accordingly. The evaluator configured the openssl s_server with an ECDHE certificate with non supported curve. The evaluator configured the TOE to act as a TLS client using a Virtual Server and Client SSL profile, as per [K14783]. The evaluator also used tcpdump to capture and analyse the traffic. The analysis showed that the connection could not be established (as expected).

2.1.2.12 Extended: TLS Client Protocol with authentication (FCS_TLSC_EXT.2)

TSS Assurance Activities

Assurance Activity AA-FCS_TLSC_EXT.2-ASE-01

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

Summary

Chapter 7 of [ST] contains the TSS. Section 7.2.6 "TLS Protocol" describes the TLS protocol.

The TSS states that the TOE implements both the TLS server and TLS client protocols. The TOE implementation of TLS client is capable of presenting to a TLS server for TLS mutual authentication. The evaluator concluded that the TSS does mention the use of client-side certificate for TLS mutual authentication.

Guidance Assurance Activities

Assurance Activity AA-FCS_TLSC_EXT.2-AGD-01

> *If the TSS indicates that mutual authentication using X.509v3 certificates is used, the evaluator shall verify that the AGD guidance includes instructions for configuring the client-side certificates for TLS mutual authentication.*

Summary

The evaluator examined section 2.3.9 "Certificate Management" of [ECG] which refers to [SSLADM] "BIG-IP System: SSL Administration" for instructions for configuring the client-side certificates for TLS mutual authentication. The evaluator then examined [SSLADM] and verified that it sufficiently explains how to perform such configuration.

Test Assurance Activities

Assurance Activity AA-FCS_TLSC_EXT.2-ATE-01

For all tests in this chapter the TLS server used for testing of the TOE shall be configured to require mutual authentication

[TD0670] Test 1: The evaluator shall establish a connection to a peer server that is configured for mutual authentication (i.e. sends a server Certificate Request (type 13) message). The evaluator observes that the TOE TLS client sends both client Certificate (type 11) and client Certificate Verify (type 15) messages during its negotiation of a TLS channel and that Application Data is sent.

[TD0670] In addition, all other testing in FCS_TLSC_EXT.1 and FIA_X509_EXT. must be performed as per the requirements.*

Summary

Test 1:
The evaluator used a computer running Linux Kali Linux. The evaluator created a CA certificate, a server certificate for openssl s_server and a client certificate for the TOE and installed the certificates accordingly. The evaluator configured the openssl s_server to support client authentication. The evaluator configured the TOE to act as a TLS client using a Virtual Server and Client SSL profile, as per [K14783]. The evaluator also used tcpdump to capture and analyse the traffic. The analysis showed that the connection was established and contained the expected messages as part of the TLS handshake.

2.1.2.13 Extended: TLS Server Protocol (FCS_TLSS_EXT.1)

FCS_TLSS_EXT.1.1

TSS Assurance Activities

Assurance Activity AA-FCS_TLSS_EXT.1.1-ASE-01

The evaluator shall check the description of the implementation of this protocol in the TSS to ensure that the ciphersuites supported are specified. The evaluator shall check the TSS to ensure that the ciphersuites specified are identical to those listed for this component.

Summary

Please note this analysis covers the following (iterated) SFRs defined in [ST] :

- FCS_TLSS_EXT.1[1]
- FCS_TLSS_EXT.1[2]
- FCS_TLSS_EXT.1[3]
- FCS_TLSS_EXT.1[4]

Chapter 7 of [ST] contains the TSS. Section 7.2.6 *TLS Protocol* describes the TLS protocol.

Table 9 "Cipher Suites" of this section lists all the supported ciphersuites for TLS v1.1 and v1.2 connections. The ciphersuites are reproduced below:

- TLS_RSA_WITH_AES_128_CBC_SHA
- TLS_RSA_WITH_AES_256_CBC_SHA
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA
- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA
- TLS_RSA_WITH_AES_128_CBC_SHA256
- TLS_RSA_WITH_AES_256_CBC_SHA256

- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256
- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384
- TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384
- TLS_RSA_WITH_AES_128_GCM_SHA256
- TLS_RSA_WITH_AES_256_GCM_SHA384

The evaluator compared the above ciphersuites list with the list specified in FCS_TLSS_EXT.1.1[1]-[4] of [ST] and found them to be consistent. The evaluator notes that although the TOE can act as a server on both the data plane and control plane, only the data plane supports elliptic curve-based TLS ciphersuites (e.g., TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA). The evaluator also noted that only the data plan supports TLS_RSA_WITH_AES_128_CBC_SHA and TLS_RSA_WITH_AES_256_CBC_SHA. These exceptions are shown in table 9 as well as the paragraphs below table 9 and found to be consistent with the claims for the control plane defined in FCS_TLSS_EXT.1[3] and FCS_TLSS_EXT.1[4].

Guidance Assurance Activities

Assurance Activity AA-FCS_TLSS_EXT.1.1-AGD-01

The evaluator shall check the guidance documentation to ensure that it contains instructions on configuring the TOE so that TLS conforms to the description in the TSS (for instance, the set of ciphersuites advertised by the TOE may have to be restricted to meet the requirements).

Summary

Sections 6.2.2.14, 6.2.2.15, 6.2.2.16, 6.2.2.17 of [ST] define FCS_TLSS_EXT.1.1[1], FCS_TLSS_EXT.1.1[2], FCS_TLSS_EXT.1.1[3], FCS_TLSS_EXT.1.1[4] which specify the TLS ciphersuites supported by the data plane and control plane parts of the TOE. The TSS (section 7.2.6 "TLS Protocol" of [ST] provides consistent information. The evaluator examined section 5.1 "TLS" of [ECG] which lists the sets of allowable ciphersuites for TLS v1.1 and TLSv1.2 for the data plane and control plane parts of the TOE. For the data plane, the evaluator examined section 2.3.10.1 "SSL Profiles" of [ECG] which provides guidance to restrict ciphersuites/algorithms for the data plane of the TOE that is performed as part of the ccmode command. It states the following:

- The ccmode command sets the allowable ciphersuites for the default client and server SSL profiles: clientssl and serverssl.
- Create and use SSL profiles based only off those default profiles, and not modify the configured ciphersuites, in order to ensure that your TLS connections are Common-Criteria-compliant.
- Do not use the clientssl-insecure-compatible and serverssl-insecure-compatible default profiles, as these include weak TLS ciphers which are not Common-Criteria-compliant.
- Specify SSL profiles to use only 2048-bit or higher RSA key sizes, or ECDSA curves p-256 or p-384

For the control plane of the TOE, the evaluator examined section 2.3.10.3 "Configuration Utility" of [ECG] , which states that ccmode script applies the following command for configuring utility SSL ciphers and protocols:

```
tmsh modify /sys httpd ( ssl-ciphersuite EECDH+AES:RSA+AES:@STRENGTH ssl- protocol
 all -SSLv2 -SSLv3 -TLSv1)
```

The same section states that in order to restrict the allowed curves, a further restriction on these ciphers must be applied. The section then provides the following command to properly configure the ciphers:

```
tmsh modify /sys httpd {ssl-ciphersuite RSA+AES:@STRENGTH ssl-protocol “all -SSLv2
 -SSLv3 -TLSv1”} RSA
```

Elliptic curves are not supported on the TLS control plane. The evaluator determined that the guidance provides the necessary documentation regarding how to configure TOE for ciphersuites.

Test Assurance Activities

Assurance Activity AA-FCS_TLSS_EXT.1.1-ATE-01

Test 1: The evaluator shall establish a TLS connection using each of the ciphersuites 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 HTTPS session. It is sufficient to observe the successful negotiation of a ciphersuite to satisfy the intent of the test; it is not necessary to examine the characteristics of the encrypted traffic to discern the ciphersuite being used (for example, that the cryptographic algorithm is 128-bit AES and not 256-bit AES).

Test 2: The evaluator shall send a Client Hello to the server with a list of ciphersuites that does not contain any of the ciphersuites in the server's ST and verify that the server denies the connection. Additionally, the evaluator shall send a Client Hello to the server containing only the TLS_NULL_WITH_NULL_NULL ciphersuite and verify that the server denies the connection.

Test 3: The evaluator shall perform the following modifications to the traffic:

a) Modify a byte in the Client Finished handshake message, and verify that the server rejects the connection and does not send any application data.

b) (Test Intent: The intent of this test is to ensure that the server's TLS implementation immediately makes use of the key exchange and authentication algorithms to: a) Correctly encrypt (D)TLS Finished message and b) Encrypt every (D)TLS message after session keys are negotiated.)
The evaluator shall use one of the claimed ciphersuites to complete a successful handshake and observe transmission of properly encrypted application data. The evaluator shall verify that no Alert with alert level Fatal (2) messages were sent.
The evaluator shall verify that the Finished message (Content type hexadecimal 16 and handshake message type hexadecimal 14) is sent immediately after the server's ChangeCipherSpec (Content type hexadecimal 14) message. The evaluator shall examine the Finished message (encrypted example in hexadecimal of a TLS record containing a Finished message, 16 03 03 00 40 11 22 33 44 55...) and confirm that it does not contain unencrypted data (unencrypted example in hexadecimal of a TLS record containing a Finished message, 16 03 03 00 40 14 00 00 0c...), by verifying that the first byte of the encrypted Finished message does not equal hexadecimal 14 for at least one of three test messages. There is a chance that an encrypted Finished message contains a hexadecimal value of '14' at the position where a plaintext Finished message would contain the message type code '14'. If the observed Finished message contains a hexadecimal value of '14' at the position where the plaintext Finished message would contain the message type code, the test shall be repeated three times in total. In case the value of '14' can be observed in all three tests it can be assumed that the Finished message has indeed been sent in plaintext and the test has to be regarded as 'failed'. Otherwise it has to be assumed that the observation of the value '14' has been due to chance and that the Finished message has indeed been sent encrypted. In that latter case the test shall be regarded as 'passed'.

Summary

Test 1:
The evaluator used a computer running Kali Linux. The evaluator created a CA certificate and client certificate for s_client and a server certificate for the TOE and installed the certificates accordingly. The evaluator configured the s_client with all the supported cipher suites see [ST] 7.2.6 "TLS Protocol". The evaluator configured the s_client for TLS communication with the TOE and configured

the TOE for TLS communication with the s_client see [ECG] 2.3.9 "Certificate Management". The evaluator used Wireshark to record and analyze the traffic between the s_server and the TOE. The analysis showed the corresponding cipher suite each time.

Test 2:
The evaluator used a computer running Kali Linux. The evaluator created a CA certificate and client certificate for s_client and a server certificate for the TOE and installed the certificates accordingly. The evaluator configured the s_client with a list of cipher suites which are not supported cipher suites see [ST] 7.2.6 "TLS Protocol". The evaluator configured the s_client for TLS communication with the TOE and configured the TOE for TLS communication with the s_client see [ECG] 2.3.9 "Certificate Management". The evaluator used Wireshark to record and analyze the traffic between the s_server and the TOE. The analysis showed that the server did not accept the provided list with the cipher suites from s_client.
The evaluator configured the s_client for TLS communication with the TOE with TLS_NULL_WITH_NULL_NULL ciphersuite and configured the TOE for TLS communication with the s_client see [ECG] 2.3.9 "Certificate Management". The evaluator used Wireshark to record and analyze the traffic between the s_server and the TOE. The analysis showed that the connection was not established (as expected).

Test 3:
The evaluator used a computer running Kali Linux. The evaluator created a CA certificate, a client certificate for s_client and a server certificate for the TOE and installed the certificates accordingly. The evaluator configured the TOE for TLS communication with the s_client see [ECG] 2.3.9 "Certificate Management" and the s_client with the TOE. The evaluator used a proxy that intercepts the communication and replies with a modified byte in the Client Finished handshake message. The evaluator checked if the connection would be established, but it failed (as expected).
After generating a fatal alert by sending a Finished message, the evaluator used a proxy that intercepts the communication and replies with a ChangeCipherSpec message, send a Client Hello with the session identifier from the previous test. The evaluator checked if the connection would be established, but it failed (as expected).
The evaluator configured the s_client with one of the supported cipher suites see [ST] 7.2.6 "TLS Protocol" for TLS communication with the TOE and configured the TOE for TLS communication with the s_client see [ECG] 2.3.9 "Certificate Management". The evaluator used Wireshark to record and analyze the traffic between the s_server and the TOE. The analysis showed that no Alert with alert level Fatal (2) messages were sent, the Finished message (handshake type hexadecimal 16) is sent immediately after the server's ChangeCipherSpec (handshake type hexadecimal 14) message and the Finished message does not contain unencrypted data.

FCS_TLSS_EXT.1.2

TSS Assurance Activities

Assurance Activity AA-FCS_TLSS_EXT.1.2-ASE-01

The evaluator shall verify that the TSS contains a description of how the TOE technically prevents the use of old SSL and TLS versions.

Summary

Please note this analysis covers the following (iterated) SFRs defined in [ST] :

- FCS_TLSS_EXT.1[1]
- FCS_TLSS_EXT.1[2]
- FCS_TLSS_EXT.1[3]

- FCS_TLSS_EXT.1[4]

Chapter 7 of [ST] contains the TSS. Section 7.2.6 *TLS Protocol* describes the TLS protocol.

The TSS states the TLS server implementation in the TOE will deny SSL 1.0, SSL 2.0, SSL 3.0, and TLS 1.0 session requests. This list of unsupported TLS versions is found to be consistent with FCS_TLSS_EXT.1.2[1]-[4].

The evaluator noted that the both FCS_TLSS_EXT.1.2 and TSS includes SSL 1.0 even though the PP does not. The evaluator finds this addition acceptable as the Application Note in [NDcPPv2.2e] states that all SSL and TLS versions must be denied by the TOE.

Guidance Assurance Activities

Assurance Activity AA-FCS_TLSS_EXT.1.2-AGD-01

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

Summary

Sections 6.2.2.14, 6.2.2.15, 6.2.2.16, 6.2.2.17 of [ST] defines FCS_TLSS_EXT.1.2[1], FCS_TLSS_EXT.1.2[2], FCS_TLSS_EXT.1.2[3], FCS_TLSS_EXT.1.2[4] which state that the TSF shall deny connections from clients requesting SSL 2.0, SSL 3.0, and TLS 1.0. The evaluator examined section 2.3.10.1 "SSL Profiles" of [ECG] and determined that there are no specific configuration required for TLS servers. The TLS version and the ciphersuite selection is determined during protocol negotiation handshake during session establishment. Please note that the set of algorithms are restricted when the TOE is configured to the CC mode (ccmode command), which is an initial configuration step for using the TOE as per section 2.3.1 "ccmode command" of [ECG] . The evaluator also examined the description in section 4 "Appendix: ccmode command" (table 2 "ccmode command") of [ECG] , which outlines the ccmode command script that is used for configuring the evaluated configuration including restricting the use of SSL v2.0, SSL v3.0, and TLS v1.0.

Test Assurance Activities

Assurance Activity AA-FCS_TLSS_EXT.1.2-ATE-01

The evaluator shall send a Client Hello requesting a connection for all mandatory and selected protocol versions in the SFR (e.g. by enumeration of protocol versions in a test client) and verify that the server denies the connection for each attempt.

Summary

The evaluator used a computer running Kali Linux. The evaluator created a CA certificate and client certificate for s_client and a server certificate for the TOE and installed the certificates accordingly. The evaluator configured the s_client for TLSv1.0 communication with the TOE and configured the TOE for TLS communication with the s_client see [ECG] 2.3.9 "Certificate Management". The evaluator used Wireshark to record and analyze the traffic between the s_server and the TOE. The analysis showed that the connection was not established (as expected).
The evaluator configured the s_client for SSL 2.0 communication with the TOE and configured the TOE for TLS communication with the s_client see [ECG] 2.3.9 "Certificate Management". The evaluator used Wireshark to record and analyze the traffic between the s_server and the TOE. The analysis showed that the connection was not established (as expected).

The evaluator configured the s_client for SSL 3.0 communication with the TOE and configured the TOE for TLS communication with the s_client see [ECG] 2.3.9 "Certificate Management". The evaluator used Wireshark to record and analyze the traffic between the s_server and the TOE. The analysis showed that the connection was not established (as expected).

FCS_TLSS_EXT.1.3

TSS Assurance Activities

Assurance Activity AA-FCS_TLSS_EXT.1.3-ASE-01

[TD0635] If using ECDHE and/or DHE ciphers, the evaluator shall verify that the TSS lists all EC Diffie-Hellman curves and/or Diffie-Hellman groups used in the key establishment by the TOE when acting as a TLS Server. For example, if the TOE supports TLS_DHE_RSA_WITH_AES_128_CBC_SHA cipher and Diffie-Hellman parameters with size 2048 bits, then list Diffie-Hellman Group 14.

Summary

Please note this analysis covers the following (iterated) SFRs defined in [ST] :

- FCS_TLSS_EXT.1[1]
- FCS_TLSS_EXT.1[2]
- FCS_TLSS_EXT.1[3]
- FCS_TLSS_EXT.1[4]

Section of 7 [ST] contains the TSS. Section 7.2.6 *TLS Protocol* describes the TLS protocol.

This section contains the following description:

> *When acting as a TLS server on the data plane, BIG-IP also generates ECDH parameters over NIST curves secp256r1 and secp384r1. The TLS server key exchange message parameters (ECDH) are as defined / required by RFC 5246 Section 7.4.3 for TLS 1.2, RFC 4346 Section 7.4.3 for TLS 1.1, and RFC 4492. For example, its classic ECDH using named curves with predefined parameters. The TOE does not support DHE_RSA cipher suites, so server key exchange messages are not sent.*

Guidance Assurance Activities

Assurance Activity AA-FCS_TLSS_EXT.1.3-AGD-01

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

Summary

Sections 6.2.2.14, 6.2.2.15 of [ST] defines FCS_TLSS_EXT.1.3[1] and FCS_TLSS_EXT.1.3[2], which states that the TSF shall generate key establishment parameters using RSA and ECDHE. Sections 6.2.2.16, 6.2.2.17 of [ST] defines FCS_TLSS_EXT.1.3[3] and FCS_TLSS_EXT.1.3[4], which states that the TSF shall generate key establishment parameters using RSA. For the data plane of the TOE, the evaluator examined section 2.3.10.1 "SSL Profiles" which states the ccmode command sets the allowable ciphersuits for the default client and server SSL profiles. When creating and using SSL profiles based off those default profiles, the administrator must not modify the configured ciphersuits in order to ensure that the TLS connections are Common-Criteria-compliant. When configuring SSL profiles, the admin should only use 2048-bit or higher RSA key sizes or ECDSA curves p-256 or p-384. The section indicates the administrator should refer to *ltm profile server-ssl* in the [TMSH-REFv12] . For the control plane of the TOE the evaluator examined section 2.3.10.3

of [ECG] and determined that there are no specific configuration required for TLS servers. The TLS version and the ciphersuite selection is determined during protocol negotiation handshake during session establishment. Please note that support for allowed algorithms, ciphersuites, and elliptic curves are restricted when the TOE is configured to the CC mode (ccmode command), which is an initial configuration step for using the TOE as per section 2.3.1 "ccmode command" of [ECG] .

Test Assurance Activities

Assurance Activity AA-FCS_TLSS_EXT.1.3-ATE-01

Test 1: [conditional] If ECDHE ciphersuites are supported:

a) The evaluator shall repeat this test for each supported elliptic curve. The evaluator shall attempt a connection using a supported ECDHE ciphersuite and a single supported elliptic curve specified in the Elliptic Curves Extension. The Evaluator shall verify (though a packet capture or instrumented client) that the TOE selects the same curve in the Server Key Exchange message and successfully establishes the connection.

b) The evaluator shall attempt a connection using a supported ECDHE ciphersuite and a single unsupported elliptic curve (e.g. secp192r1 (0x13)) specified in RFC4492, chap. 5.1.1. The evaluator shall verify that the TOE does not send a Server Hello message and the connection is not successfully established.

Test 2: [conditional] If DHE ciphersuites are supported, the evaluator shall repeat the following test for each supported parameter size. If any configuration is necessary, the evaluator shall configure the TOE to use a supported Diffie-Hellman parameter size. The evaluator shall attempt a connection using a supported DHE ciphersuite. The evaluator shall verify (through a packet capture or instrumented client) that the TOE sends a Server Key Exchange Message where p Length is consistent with the message are the ones configured Diffie-Hellman parameter size(s).

Test 3: [conditional] If RSA key establishment ciphersuites are supported, the evaluator shall repeat this test for each RSA key establishment key size. If any configuration is necessary, the evaluator shall configure the TOE to perform RSA key establishment using a supported key size (e.g. by loading a certificate with the appropriate key size). The evaluator shall attempt a connection using a supported RSA key establishment ciphersuite. The evaluator shall verify (through a packet capture or instrumented client) that the TOE sends a certificate whose modulus is consistent with the configured RSA key size.

Summary

The evaluator used a computer running Kali Linux. The evaluator created CA certificate, client certificate for s_client and a server certificates for the TOE and installed the certificates accordingly. The evaluator configured the s_client with all the supported Diffie Hellman curves see [ST] 7.2.6 "TLS Protocol". The evaluator configured the s_client for TLS communication with the TOE and configured the TOE for TLS communication with the s_client see [ECG] 2.3.9 "Certificate Management". The evaluator used Wireshark to record and analyze the traffic between the s_server and the TOE. The analysis showed that the connection could be established (as expected) for each parameter combination.
The evaluator used a computer running Kali Linux. The evaluator created CA certificate and server certificates with different key sizes for the TOE and installed the certificates accordingly. The evaluator configured the s_client with appropriate curve each time for TLS communication with the TOE and configured the TOE with appropriate key size for TLS communication with the s_client see [ECG] 2.3.9 "Certificate Management". The evaluator used Wireshark to record and analyze the traffic between the s_client and the TOE. The analysis showed that the connection was established (as expected) for each parameter combination.

FCS_TLSS_EXT.1.4

TSS Assurance Activities

Assurance Activity AA-FCS_TLSS_EXT.1.4-ASE-01

The evaluator shall verify that the TSS describes if session resumption based on session IDs is supported (RFC 4346 and/or RFC 5246) and/or if session resumption based on session tickets is supported (RFC 5077).

If session tickets are supported, the evaluator shall verify that the TSS describes that the session tickets are encrypted using symmetric algorithms consistent with FCS_COP.1/DataEncryption. The evaluator shall verify that the TSS identifies the key lengths and algorithms used to protect session tickets.

If session tickets are supported, the evaluator shall verify that the TSS describes that session tickets adhere to the structural format provided in section 4 of RFC 5077 and if not, a justification shall be given of the actual session ticket format.

Summary

Please note this analysis covers the following (iterated) SFRs defined in [ST] :

- FCS_TLSS_EXT.1[1]
- FCS_TLSS_EXT.1[2]
- FCS_TLSS_EXT.1[3]
- FCS_TLSS_EXT.1[4]

For FCS_TLSS_EXT.1.4, the ST selects "session resumption based on session tickets according to RFC 5077".

Section 7.2.6 *TLS Protocol* in the TSS of [ST] states the following:

- The TLS server supports session resumption based on session tickets according to RFC 5077. These session tickets adhere to the structural format described in Section 4 of RFC 5077. These session tickets are encrypted using the AES with CBC mode symmetric algorithm with 128 bit key length as defined in FCS_COP.1/DataEncryption.
- Session establishment creates a session ID. When a new context is started and a session ID is offered, the session ID is verified to be acceptable to allow session resumption by checking the validity of the session ID in the session ID table, the age of the session ID, the cipher suite offered in the session ID, configuration settings of the session ID, and the Server Name Indication (SNI). Any failure in these validation steps listed below would trigger a full handshake.
- Multiple contexts are supported for session resumption. A session can be constructed in one context and resumed in another context. The context which constructs the session ID during full handshake is the owner of that session ID and also validates the session ID and session state. Contexts which resume a session request that the originating context session owner validate the session ID and session state. If the originating context session validation response does not validate the session, a full handshake is triggered. Contexts validate sessions by requesting that the originating owner of a session validate a session before resumption can continue. If a session is not validated, a full handshake is triggered.

Guidance Assurance Activities

Assurance Activity AA-FCS_TLSS_EXT.1.4-AGD-01

[TD0569] The evaluator shall verify that any configuration necessary to meet the requirement must be contained in the AGD guidance.

Summary

The evaluator exmined section 3.10 "Session Resumption" of [ECG] which states that the TLS server supports session resumption based on session tickets according to RFC 5077. These session tickets adhere to the structural format described in Section 4 of RFC 5077. These session tickets are encrypted using the AES with CBC mode symmetric algorithm with 128 key length as defined in FCS_COP.1/DataEncryption.

It also states the following:

> "*Session establishment creates a session ID. When a new context is started and a session ID is offered, the session ID is verified to be acceptable to allow session resumption by checking the validity of the session ID in the session ID table, the age of the session ID, the cipher suite offered in the session ID, configuration settings of the session ID, and the Server Name Indication (SNI). Any failure in these validation steps listed below would trigger a full handshake.*"
>
> "*Multiple contexts are supported for session resumption. A session can be constructed in one context and resumed in another context. The context which constructs the session ID during full handshake is the owner of that session ID and also validates the session ID and session state. Contexts which resume a session request that the originating context session owner validate the session ID and session state. If the originating context session validation response does not validate the session, a full handshake is triggered. Contexts validate sessions by requesting that the originating owner of a session validate a session before resumption can continue. If a session is not validated, a full handshake is triggered.*"

Test Assurance Activities

Assurance Activity AA-FCS_TLSS_EXT.1.4-ATE-01

Test Objective: To demonstrate that the TOE will not resume a session for which the client failed to complete the handshake (independent of TOE support for session resumption).

Test 1 [conditional]: If the TOE does not support session resumption based on session IDs according to RFC4346 (TLS1.1) or RFC5246 (TLS1.2) or session tickets according to RFC5077, the evaluator shall perform the following test:

a) The client sends a Client Hello with a zero-length session identifier and with a SessionTicket extension containing a zero-length ticket.

b) The client verifies the server does not send a NewSessionTicket handshake message (at any point in the handshake).

c) The client verifies the Server Hello message contains a zero-length session identifier or passes the following steps: Note: The following steps are only performed if the ServerHello message contains a non-zero length SessionID.

d) The client completes the TLS handshake and captures the SessionID from the ServerHello.

e) The client sends a ClientHello containing the SessionID captured in step d). This can be done by keeping the TLS session in step d) open or start a new TLS session using the SessionID captured in step d).

f) The client verifies the TOE (1) implicitly rejects the SessionID by sending a ServerHello containing a different SessionID and by performing a full handshake (as shown in Figure 1 of RFC 4346 or RFC 5246), or (2) terminates the connection in some way that prevents the flow of application data.

[TD0569] Remark: If multiple contexts are supported for session resumption, the session ID or session ticket may be obtained in one context for resumption in another context. It is possible that one or more contexts may only permit the construction of sessions to be reused in other contexts but not actually permit resumption themselves. For contexts which do not permit resumption, the evaluator is required to verify this behaviour subject to the description provided in the TSS. It is not mandated that the session establishment and session resumption share context. For example, it is acceptable for a control channel to establish and application channel to resume the session.

Test 2 [conditional]: If the TOE supports session resumption using session IDs according to RFC4346 (TLS1.1) or RFC5246 (TLS1.2), the evaluator shall carry out the following steps (note that for each of these tests, it is not necessary to perform the test case for each supported version of TLS):

a) The evaluator shall conduct a successful handshake and capture the TOE-generated session ID in the Server Hello message. The evaluator shall then initiate a new TLS connection and send the previously captured session ID to show that the TOE resumed the previous session by responding with ServerHello containing the same SessionID immediately followed by ChangeCipherSpec and Finished messages (as shown in Figure 2 of RFC 4346 or RFC 5246).

b) The evaluator shall initiate a handshake and capture the TOE-generated session ID in the Server Hello message. The evaluator shall then, within the same handshake, generate or force an unencrypted fatal Alert message immediately before the client would otherwise send its ChangeCipherSpec message thereby disrupting the handshake. The evaluator shall then initiate a new Client Hello using the previously captured session ID, and verify that the server (1) implicitly rejects the session ID by sending a ServerHello containing a different SessionID and performing a full handshake (as shown in figure 1 of RFC 4346 or RFC 5246), or (2) terminates the connection in some way that prevents the flow of application data.

[TD0569] Remark: If multiple contexts are supported for session resumption, for each of the above test cases, the session ID may be obtained in one context for resumption in another context. There is no requirement that the session ID be obtained and replayed within the same context subject to the description provided in the TSS. All contexts that can reuse a session ID constructed in another context must be tested. It is not mandated that the session establishment and session resumption share context. For example, it is acceptable for a control channel to establish and application channel to resume the session.

Test 3 [conditional]: If the TOE supports session tickets according to RFC5077, the evaluator shall carry out the following steps (note that for each of these tests, it is not necessary to perform the test case for each supported version of TLS):

a) [TD0556] The evaluator shall permit a successful TLS handshake to occur in which a session ticket is exchanged with the non-TOE client. The evaluator shall then attempt to correctly reuse the previous session by sending the session ticket in the ClientHello. The evaluator shall confirm that the TOE responds with an abbreviated handshake described in section 3.1 of RFC 5077 and illustrated with an example in figure 2. Of particular note: if the server successfully verifies the client's ticket, then it may renew the ticket by including a NewSessionTicket handshake message after the ServerHello in the abbreviated handshake (which is shown in figure 2). This is not required, however as further clarified in section 3.3 of RFC 5077.

b) The evaluator shall permit a successful TLS handshake to occur in which a session ticket is exchanged with the non-TOE client. The evaluator will then modify the session ticket and send it as part of a new Client Hello message. The evaluator shall confirm that the TOE either (1) implicitly rejects the session ticket by performing a full handshake (as shown in figure 3 or 4 of RFC 5077), or (2) terminates the connection in some way that prevents the flow of application data.

[TD0569] Remark: If multiple contexts are supported for session resumption, for each of the above test cases, the session ticket may be obtained in one context for resumption in another context. There is no requirement that the session ticket be obtained and replayed within the same context subject to the description provided in the TSS. All contexts that can reuse a session ticket constructed in another context must be tested. It is not mandated that the session establishment and session resumption share context. For example, it is acceptable for a control channel to establish and application channel to resume the session

Summary

Test 1:
[ST] 7.2.6 "TLS Protocol" states that the TOE supports session resumption based on session tickets. The evaluator therefore determines that this requirement is not applicable.

Test 2:
[ST] 7.2.6 "TLS Protocol" states that the TOE supports session resumption based on session tickets. The evaluator therefore determines that this requirement is not applicable.

Test 3:
The evaluator used a computer running Kali Linux. The evaluator created a CA certificate, a client certificate for s_client and a server certificate for the TOE and installed the certificates accordingly. The evaluator configured the s_client for TLS communication with the TOE and configured the TOE for TLS communication with the s_client see [ECG] 2.3.9 "Certificate Management". The evaluator configured the s_client for TLS session resumption by sending the session ticket of the previous successfully established TLS connection in the ClientHello. The evaluator used Wireshark to record and analyze the traffic between the s_server and the TOE. The analysis showed that the connection was established (as expected). The evaluator configured the s_client for TLS session resumption by sending a modified session ticket of the previous successfully established TLS connection in the ClientHello. The evaluator used Wireshark to record and analyze the traffic between the s_server and the TOE. The analysis showed that the connection was not established (as expected).

2.1.3 Identification and authentication (FIA)

2.1.3.1 Authentication Failure Management (FIA_AFL.1)

TSS Assurance Activities

Assurance Activity AA-FIA_AFL.1-ASE-01

The evaluator shall examine the TSS to determine that it contains a description, for each supported method for remote administrative actions, of how successive unsuccessful authentication attempts are detected and tracked. The TSS shall also describe the method by which the remote administrator is prevented from successfully logging on to the TOE, and the actions necessary to restore this ability.

The evaluator shall examine the TSS to confirm that the TOE ensures that authentication failures by remote administrators cannot lead to a situation where no administrator access is available, either permanently or temporarily (e.g. by providing local logon which is not subject to blocking).

Summary

Chapter 7 of [ST] contains the TSS. Section 7.3 *Identification and Authentication* describes user authentication and identification. The evaluator examined the TSS which states the following relevant description:

- Except for the admin user account, both local and remote access to the TOE for individual users can be locked after reaching a configured number of consecutive, failed authentication attempts.
- For each administrative interface (both local and remote interfaces), a single centralized module in the TOE verifies user identification and authentication. This module returns authentication success or failure decisions and maintains the user lockout feature. A counter of failed authentication attempts is maintained for each user. If the failed authentication attempts exceeds the allowed number the user account is disabled.
- A counter is kept for each user to track authentication failures and is reset to zero when a successful authentication occurs.
- Users with the Administrator or User Manager role have to manually unlock accounts that have been locked by the TOE.
- It is not possible for all administrative users to be locked out of the TOE, because the primary administrative user account is permitted to login to the local console even if it is locked out when attempting to login through any remote interface.

Guidance Assurance Activities

Assurance Activity AA-FIA_AFL.1-AGD-01

The evaluator shall examine the guidance documentation to ensure that instructions for configuring the number of successive unsuccessful authentication attempts and time period (if implemented) are provided, and that the process of allowing the remote administrator to once again successfully log on is described for each "action" specified (if that option is chosen). If different actions or mechanisms are implemented depending on the secure protocol employed (e.g., TLS vs. SSH), all must be described.

The evaluator shall examine the guidance documentation to confirm that it describes, and identifies the importance of, any actions that are required in order to ensure that administrator access will always be maintained, even if remote administration is made permanently or temporarily unavailable due to blocking of accounts as a result of FIA_AFL.1.

Summary

Section 3.2 "Maximum Failed Login Attempts" of [ECG] provides relevant guidance for configuration failed login attempts. It states the following which applies to all administrative interfaces:

- The administrator can set a parameter that specifies the maximum number of consecutive failed login attempts that can occur before a given user account will be locked out. The default setting is 3. It is highly recommended that the default setting be retained (i.e., not changed).
- If a user becomes locked out, the user account will be unlocked after an administrator-specified duration. The ccmode script sets the default to 600 seconds (10 minutes).
- The ccmode script also configures the evaluated configuration to disable the manual unlock (in favor of the timed unlock), and to allow the primary administrative user (generally “admin”) to log on from the local serial console even if the account is locked. This ensures that at least one user account is available at all times. If the primary administrative user does log in locally, its lockout counter will be reset and it will be able to log in remotely as well.

Additionally, sections 2.2.7 "Create an Administrative User with tmsh access" and 2.3.3.1 "Administrative users" of [ECG] contain a the following relevant recommendations, respectively:

> *NOTE: It is strongly recommended that, in addition to the administrative-user created above, you configure the primary administrative user (generally “admin”) with tmsh access as well, as this user is the only administrative-user able to login locally if otherwise locked out.*

and

> *It is strongly recommended that the primary administrative user (generally “admin”) have tmsh access, as this user is the only administrative-user able to login locally if otherwise locked out.*

Test Assurance Activities

Assurance Activity AA-FIA_AFL.1-ATE-01

The evaluator shall perform the following tests for each method by which remote administrators access the TOE (e.g. any passwords entered as part of establishing the connection protocol or the remote administrator application):

a) Test 1: The evaluator shall use the operational guidance to configure the number of successive unsuccessful authentication attempts allowed by the TOE (and, if the time period selection in FIA_AFL.1.2 is included in the ST, then the evaluator shall also use the operational guidance to configure the time period after which access is re-enabled). The evaluator shall test that once the authentication attempts limit is reached, authentication attempts with valid credentials are no longer successful.

b) Test 2: After reaching the limit for unsuccessful authentication attempts as in Test 1 above, the evaluator shall proceed as follows.

If the administrator action selection in FIA_AFL.1.2 is included in the ST, then the evaluator shall confirm by testing that following the operational guidance and performing each action specified in the ST to re-enable the remote administrator's access results in successful access (when using valid credentials for that administrator).

If the time period selection in FIA_AFL.1.2 is included in the ST, then the evaluator shall wait for just less than the time period configured in Test 1 and show that an authorisation attempt using valid credentials does not result in successful access. The evaluator shall then wait until just after the time period configured in Test 1 and show that an authorisation attempt using valid credentials results in successful access.

Summary

Test 1:
The evaluator used a computer running Ubuntu Linux as a remote HTTPS client. The evaluator logged in successfully to the TOE as an administrator user through SSH. The evaluator configured the TOE with different maximum number of failed login attempts and lockout duration of the locked

out users, see [ECG] 3.2 "Maximum Failed Login Attempts". The evaluator locked out a user after attempting to login with invalid credentials via GUI. Then the evaluator tried to login with valid credentials via GUI for the same user but failed (as expected).

The evaluator used a computer running Ubuntu Linux as a remote HTTPS client. The evaluator logged in successfully to the TOE as an administrator user through SSH. The evaluator configured the TOE with different maximum number of failed login attempts and lockout duration of the locked out users, see [ECG] 3.2 "Maximum Failed Login Attempts". The evaluator locked out a user after attempting to login with invalid credentials via SSH. Then the evaluator tried to login with valid credentials via SSH for the same user but failed (as expected).

The evaluator used a computer running Ubuntu Linux as a remote HTTPS client. The evaluator logged in successfully to the TOE as an administrator user through SSH. The evaluator configured the TOE with different maximum number of failed login attempts and lockout duration of the locked out users, see [ECG] 3.2 "Maximum Failed Login Attempts". The evaluator locked out a user after attempting to login with invalid credentials via IControl Rest. Then the evaluator tried to login with valid credentials via IControl Rest for the same user but failed (as expected).

The evaluator used a computer running Ubuntu Linux as a remote HTTPS client. The evaluator logged in successfully to the TOE as an administrator user through SSH. The evaluator configured the TOE with different maximum number of failed login attempts and lockout duration of the locked out users, see [ECG] 3.2 "Maximum Failed Login Attempts". The evaluator locked out a user after attempting to login with invalid credentials via IControl. Then the evaluator tried to login with valid credentials via IControl for the same user but failed (as expected).

Test 2:
According to [ST] "7.3.1 Password policy and user lockout" the evaluator waited until the expiration of the lockout duration of the locked out users and tried to login via GUI again with valid credentials for the same user and it was successful.

According to [ST] "7.3.1 Password policy and user lockout" the evaluator waited until the expiration of the lockout duration of the locked out users and tried to login via SSH again with valid credentials for the same user and it was successful.

According to [ST] "7.3.1 Password policy and user lockout" the evaluator waited until the expiration of the lockout duration of the locked out users and tried to login via IControl Rest again with valid credentials for the same user and it was successful.

According to [ST] "7.3.1 Password policy and user lockout" the evaluator waited until the expiration of the lockout duration of the locked out users and tried to login via IControl again with valid credentials for the same user and it was successful.

2.1.3.2 Password Management (FIA_PMG_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FIA_PMG_EXT.1-ASE-01

The evaluator shall examine the TSS to determine that it contains the lists of the supported special character(s) and minimum and maximum number of characters supported for administrator passwords.

Summary

Section 7.4.1 *Password policy and user lockout* contains the lists of supported special character(s) and minimum and maximum number of characters supported for administrator passwords as follows:

- Special characters: “!”, “@”, “#”, “$”, “%”, “^”, “&”, “*”, “(“, “)”, “`”, “~”, “-“, “+”, “=”, “[“, “]”, “{“, “}”, “;”, “’”, “:”, “””, “,”, “.”, “/”, “>”, “<”, “”, “|”, “\”.
- The minimum password length default value is 15; the valid range is from 15 to 255.

Guidance Assurance Activities

Assurance Activity AA-FIA_PMG_EXT.1-AGD-01

The evaluator shall examine the guidance documentation to determine that it:

a) identifies the characters that may be used in passwords and provides guidance to security administrators on the composition of strong passwords, and

b) provides instructions on setting the minimum password length and describes the valid minimum password lengths supported.

Summary

The evaluator examined section 3.1 "Password Selection Requirements" of [ECG] and determined that it provides guidance to security administrators on the composition of strong passwords. This section states that the "ccmode" command includes password policy configuration to the Common Criteria requirements (as described in section 4 of [ECG] , however, the administrator may manually configure more restrictive password policy by following the instructions provided in this section. This section also refers to [K15497] (K15497: Configuring a secure password policy for the BIG-IP system (11.x - 16.x)) which the evaluator examined and found guidance on setting a number of strong password related parameters, e.g., minimum password length, maximum login failures.

Test Assurance Activities

Assurance Activity AA-FIA_PMG_EXT.1-ATE-01

The evaluator shall perform the following tests.

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

b) Test 2: The evaluator shall compose passwords that do not meet the requirements in some way. For each password, the evaluator shall verify that the TOE does not support the password. While the evaluator is not required (nor is it feasible) to test all possible compositions of passwords, the evaluator shall ensure that the TOE enforces the allowed characters and the minimum length listed in the requirement and justify the subset of those characters chosen for testing.

Summary

The evaluator used a computer running Ubuntu Linux as a HTTPS client. The evaluator logged in the TOE via GUI and created users with passwords that either meet the requirements, or fail to meet the requirements according to [ST] "6.2.3 Identification and authentication (FIA)" for the supported password rules. For example the evaluator created user accounts with:

- No uppercase letters in password and hence failed (as expected).
- No lowercase letters in password and hence failed (as expected).

- No numeric characters in password and hence failed (as expected).
- No special characters in password and hence failed (as expected).
- "P{a%*(12{3c45{67`~-" as password and succeeded.
- "Pa+=[123>456$7]{}" as password and succeeded.
- "P{a;':12{3$4 5{67",.'" as password and succeeded.
- "P{a)|\1{23$4”5{67$%^" as password and succeeded.

2.1.3.3 Protected Authentication Feedback (FIA_UAU.7)

TSS Assurance Activities

No assurance activities defined.

Guidance Assurance Activities

Assurance Activity AA-FIA_UAU.7-AGD-01

The evaluator shall examine the guidance documentation to determine that any necessary preparatory steps to ensure authentication data is not revealed while entering for each local login allowed.

Summary

The evaluator examined section 3.1.1 "Configuring a password policy for administrative users" of [ECG] which states that "Password entry is obfuscated by default."

Test Assurance Activities

Assurance Activity AA-FIA_UAU.7-ATE-01

The evaluator shall perform the following test for each method of local login allowed:

a) Test 1: The evaluator shall locally authenticate to the TOE. While making this attempt, the evaluator shall verify that at most obscured feedback is provided while entering the authentication information.

Summary

This test was covered under the test FTA_SSL_EXT.1

2.1.3.4 Password-based Authentication Mechanism (FIA_UAU_EXT.2)

TSS Assurance Activities

Assurance Activity AA-FIA_UAU_EXT.2-ASE-01

Evaluation Activities for this requirement are covered under those for FIA_UIA_EXT.1. If other authentication mechanisms are specified, the evaluator shall include those methods in the activities for FIA_UIA_EXT.1.

Summary

This work unit was performed in conjunction with AA-FIA_UIA_EXT.1-ASE-01 .

Guidance Assurance Activities

Assurance Activity AA-FIA_UAU_EXT.2-AGD-01

Evaluation Activities for this requirement are covered under those for FIA_UIA_EXT.1. If other authentication mechanisms are specified, the evaluator shall include those methods in the activities for FIA_UIA_EXT.1.

Summary

The evaluator performed these evaluation activities in conjunction with the evaluation activities for FIA_UIA_EXT.1. Thus, the evaluator determined the evaluation activities for FIA_UAU_EXT.2 are met.

Test Assurance Activities

Assurance Activity AA-FIA_UAU_EXT.2-ATE-01

Evaluation Activities for this requirement are covered under those for FIA_UIA_EXT.1. If other authentication mechanisms are specified, the evaluator shall include those methods in the activities for FIA_UIA_EXT.1.

Summary

This test was covered under the test FIA_UIA_EXT.1.

2.1.3.5 User Identification and Authentication (FIA_UIA_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FIA_UIA_EXT.1-ASE-01

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

The evaluator shall examine the TSS to determine that it describes which actions are allowed before user identification and authentication. The description shall cover authentication and identification for local and remote TOE administration.

Summary

Chapter 7 of [ST] contains the TSS. Section 7.4 *Security Function Management* identifies the user interfaces to the TOE:

- Configuration Utility (Web-based GUI) - browser-based GUI interface with normal GUI panels and selections. The client browser talks to the Apache HTTP server over HTTPS; then the request passes through tomcat and to the BIG-IP.
- tmsh shell commands – provide a command line interface, accessible through an SSH client or a serial port console.
- iControl API – SOAP-based programming interface over HTTPS.
- iControl REST API – REST-based programming interface over HTTPS.

The TOE supports local and remote login methods. Local login is described in section 7.3 *Identification and Authentication* and remote login is described in section 7.2 *Cryptographic Support* and section 7.8 *Trusted Path/Channels* of the TSS. Local login is via password-based authentication. Administrative users are identified by a user name and authenticated by an individual password associated with that user's account.

The TOE provides three interfaces for administrators to login remotely: Configuration Utility, iControl API, and iControl Rest API. These interfaces communicate with the TOE via TLS (HTTPS). TLS sessions are restricted to TLS versions 1.2 and 1.1, using the ciphersuites identified in table 9 "Cipher suites" of [ST] . Connection to tmsh is via SSH version 2 or serial port console.

For local password-based authentication, section 7.3 states:

> *Administrative users (i.e., all users authorized to access the TOE's administrative interfaces) are identified by a user name and authenticated by an individual password associated with that user's account. This is true regardless of how the administrative user interfaces with the TOE. If the supplied user name and password match the user name and password pair maintained by the TOE, the administrative session is successfully established. Otherwise, the user receives an error and the session is not established. [...].*

The evaluator determined that the TSS contains the necessary information.

Assurance Activity AA-FIA_UIA_EXT.1-ASE-02

For distributed TOEs the evaluator shall examine that the TSS details how Security Administrators are authenticated and identified by all TOE components. If not, all TOE components support authentication of Security Administrators according to FIA_UIA_EXT.1 and FIA_UAU_EXT.2, the TSS shall describe how the overall TOE functionality is split between TOE components including how it is ensured that no unauthorized access to any TOE component can occur.

For distributed TOEs, the evaluator shall examine the TSS to determine that it describes for each TOE component which actions are allowed before user identification and authentication. The description shall cover authentication and identification for local and remote TOE administration. For each TOE component that does not support authentication of Security Administrators according to FIA_UIA_EXT.1 and FIA_UAU_EXT.2 the TSS shall describe any unauthenticated services/services that are supported by the component.

Summary

According to the Security Target [ST] , the TOE is not distributed therefore this Evaluation Activity is not applicable.

Guidance Assurance Activities

Assurance Activity AA-FIA_UIA_EXT.1-AGD-01

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

Summary

The evaluator examined section 2 "Installation and Configuration Procedures" of [ECG] , and determined it describes the following necessary preparatory steps for logging in:

- Password-based login:
  - Execute the Configuration Setup utility to configure basic information such as admin password, management port IP address(es), basic network information, and high availability configuration.
  - Create an administrative-user with the Administrator role and tmsh access. This user account will be used to perform the rest of configuration steps. There is no password policy enforcement in effect at this time but a password must be created according to the password policy.

- Run the ccmode command to apply the Common Criteria requirements which performs functions such as setting the required password policy, the allowed ciphersuites for TLS, logging options, etc..
- The TOE supports identification/authentication of each user through local user database
- Configure administrative accounts, their associated roles, and password-policy-compliant passwords.
- Ensure that at least one Administrative-user account has tmsh access.
- Configure advisory notice and consent warming to be displayed before establishment of administrative user sessions for the GUI and tmsh sessions.
- Configure security settings for administrative login using the procedure described in section 2.3.4 "Login to the BIG-IP" of [ECG] .

- Key-based login:
  - Per section 2.3.10.2 "SSH" of [ECG] , the ccmode command and the default SSH server profile set the allowable ciphersuites for SSH. No additional action is necessary to use these restricted ciphersuites.

The evaluator determined the guidance provides clear instructions for successful logging. For example, both the GUI and tmsh interfaces would display a warning label once the user logs on.

FIA_UIA_EXT.1.1 (section 6.2.3.3 of [ST]) requires that the TSF shall only allow "display the warning banner in accordance with FTA_TAB.1" prior to requiring the non-TOE entity to initiate the identification and authentication process. Displaying the warning banner is an action taken by default by the TSF (for both GUI and tmsh administrative interfaces), no user configuration is required. The content of the banner can be configured by users according to the guidance described section 2.3.5 "Login welcome banners" of [ECG] .

Test Assurance Activities

Assurance Activity AA-FIA_UIA_EXT.1-ATE-01

The evaluator shall perform the following tests for each method by which administrators access the TOE (local and remote), as well as for each type of credential supported by the login method:

a) Test 1: The evaluator shall use the guidance documentation to configure the appropriate credential supported for the login method. For that credential/login method, the evaluator shall show that providing correct I&A information results in the ability to access the system, while providing incorrect information results in denial of access.

b) Test 2: The evaluator shall configure the services allowed (if any) according to the guidance documentation, and then determine the services available to an external remote entity. The evaluator shall determine that the list of services available is limited to those specified in the requirement.

c) Test 3: For local access, the evaluator shall determine what services are available to a local administrator prior to logging in, and make sure this list is consistent with the requirement.

d) Test 4: For distributed TOEs where not all TOE components support the authentication of Security Administrators according to FIA_UIA_EXT.1 and FIA_UAU_EXT.2, the evaluator shall test that the components authenticate Security Administrators as described in the TSS.

Summary

Test 1:
The evaluator used a computer running Ubuntu Linux as a remote SSH client. The evaluator logged in successfully to the TOE as a user through SSH. The evaluator attempted to login to the TOE with incorrect password but failed (as expected).

The evaluator used a computer running Ubuntu Linux as a remote HTTPS client. The evaluator logged in successfully to the TOE as a user through GUI. The evaluator attempted to login to the TOE with incorrect password but failed (as expected).

The evaluator used a computer running Ubuntu Linux as a remote HTTPS client. The evaluator could retrieve user management list successfully from the TOE as a user through IControl by providing the correct username and password. The evaluator attempted to retrieve data from TOE using incorrect password but failed (as expected).

The evaluator used a computer running Ubuntu Linux as a remote HTTPS client. The evaluator could retrieve user list successfully from the TOE as a user through IControl REST by providing the correct username and password. The evaluator attempted to retrieve data from TOE using incorrect password but failed (as expected).

The evaluator used a computer running Ubuntu Linux as a remote SSH client. The evaluator created a SSH key pair and installed the public key to the TOE. The evaluator logged in successfully to the TOE as a user through SSH. The evaluator attempted to login to the TOE with incorrect SSH key pair but failed (as expected).

Test 2:
The evaluator found that the TOE does not support any services available to an external remote entity, therefore there are no specific requirements for this assurance activity.

Test 3:
The evaluator used a computer running Ubuntu Linux as a ssh client. The evaluator used the VMWare client to gain access to the serial console of the TOE. The evaluator could login successfully via the serial console of the TOE as a local user. The evaluator attempted to login to the TOE with incorrect password but failed (as expected).

Test 4:
The evaluator found that the TOE is not a distributed TOE, therefore there are no specific requirements for this assurance activity.

2.1.3.6 X.509 Certificate Validation (FIA_X509_EXT.1/Rev)

TSS Assurance Activities

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

The evaluator shall ensure the TSS describes where the check of validity of the certificates takes place, and that the TSS identifies any of the rules for extendedKeyUsage fields (in FIA_X509_EXT.1.1) that are not supported by the TOE (i.e. where the ST is therefore claiming that they are trivially satisfied). It is expected that revocation checking is performed when a certificate is used in an authentication step and when performing trusted updates (if selected). It is not necessary to verify the revocation status of X.509 certificates during power-up self-tests (if the option for using X.509 certificates for self-testing is selected).

The TSS shall describe when revocation checking is performed and on what certificates. If the revocation checking during authentication is handled differently depending on whether a full certificate chain or only a leaf certificate is being presented, any differences must be summarized in the TSS section and explained in the Guidance.

Summary

Per the section 6.2.3.6 *FIA_X509_EXT.1/Rev X.509 Certificate Validation* of [ST] , the TOE supports all the rules for extendedKeyUsage fields defined in FIA_X509_EXT.1.1, thus the TSS does not identify any unsupported rules.

Per the TSS (section 7.3.2 *Certificate Validation* of [ST]), the TOE checks the validity of the certificates when the profile using the certificate is loaded and when the certificate is used by the TOE, including during authentication. A certificate chain includes the root CA certificate, certificates of intermediate CAs, and the end entity certificate. The certificate chain consists of all the certificates necessary to validate the end certificate. Administrators can upload trusted device certificates (root CA certificates) into the TOE to identify which certificates are trusted. The TOE performs full certificate chain checking using Public Key Infrastructure X.509, verifies the expiration of the certificate (assuming a reliable time), and verifies its revocation using CRLs.

The TOE supports RSA-based digital signature and not X.509 certificates for trusted updates.

Guidance Assurance Activities

Assurance Activity AA-FIA_X509_EXT.1-REV-AGD-01

The evaluator shall also ensure that the guidance documentation describes where the check of validity of the certificates takes place, describes any of the rules for extendedKeyUsage fields (in FIA_X509_EXT.1.1) that are not supported by the TOE (i.e. where the ST is therefore claiming that they are trivially satisfied) and describes how certificate revocation checking is performed and on which certificate.

Summary

The evaluator examined section 2 "Installation and Configuration Procedures" of [ECG] . In particular section 2.3.9 "Certificate Management" provides guidance on managing X.509 certificates with references to the user guides [K15664] , [K14620] , [K15462] , [K14806] , [K14783] , and [SSLADM] . The evaluator examined these user guides which provide detailed instructions such as to request a certificate from a CA, generation of a Certificate Request Message including instructions for establishing the fields specified in [ST] such as "Common Name", "Organization", and "Country".

OE.TRUSTED _ADMIN: For TOEs supporting X.509v3 certificate-based authentication, the Security Administrator(s) are assumed to monitor the revocation status of all certificates in the TOE's trust store and to remove any certificate from the TOE’s trust store in case such certificate can no longer be trusted.

Section 3.8 "Certificate Validation" of the [ECG] states the following:

> " *The TOE supports validation of X.509 digital certificates using a certificate revocation list (CRL) as specified in [RFC 5280] Section 5. Administrators create profiles which are used to define the parameters used to communicate with an external entity. These parameters include the ability to require the use of TLS and peer or mutual authentication and a definition of the certificate to use for authentication. This capability is used to create a mutually authenticated connection with the external audit server. The external audit server provides a certificate to the TOE during establishment of the TLS connection in order to authenticate the external audit server.*
>
> *The TOE offers administrative interfaces for creating a private key and certificate signing request (CSR). The CSR may include the following information: public key, common name, organization, organizational unit, country, locality, state / province, country, e-mail address, subject alternative name. After the CSR is created, the administrator must export the CSR outside the TOE. Outside the scope of the TOE, the administrator provides the CSR to the CA and then the CA returns the certificate to the administrator. Using the administrative interface, the administrator can then import the certificate into the TOE. The only method supported by the TOE for obtaining a CA certificate is for the administrator to save a certificate to a text file and import it into the TOE.*

The certificates are stored in a text file. The TOE is capable of importing X.509v3 certificates and certificates in the PKCS12 format. The TOE is also capable of creating and using a self-signed certificate.

The TOE checks the validity of the certificates when the profile using the certificate is loaded and when the certificate is used by the TOE, including during authentication. If the certificates are modified, the digital signature verification would detect that the certificate had been tampered with and the certificate would be invalid. Administrators can ensure that the certificates presented have not been revoked by importing a certificate revocation list (CRL) into the TOE.

A certificate chain includes the root CA certificate, certificates of intermediate CAs, and the end entity certificate. The certificate chain consists of all the certificates necessary to validate the end certificate. Administrators can upload trusted device certificates (root CA certificates) into the TOE to identify which certificates are trusted. The TOE performs full certificate chain checking using Public Key Infrastructure X.509, verifies the expiration of the certificate (assuming a reliable time), and verifies its revocation using CRLs.

When the validity of a certificate cannot be established, the TOE will allow the administrator to choose whether or not to accept the certificate. "

Test Assurance Activities

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

The evaluator shall demonstrate that checking the validity of a certificate is performed when a certificate is used in an authentication step or when performing trusted updates (if FPT_TUD_EXT.2 is selected). It is not sufficient to verify the status of a X.509 certificate only when it is loaded onto the TOE. It is not necessary to verify the revocation status of X.509 certificates during power-up self-tests (if the option for using X.509 certificates for self-testing is selected). The evaluator shall perform the following tests for FIA_X509_EXT.1.1/Rev. These tests must be repeated for each distinct security function that utilizes X.509v3 certificates. For example, if the TOE implements certificate-based authentication with IPSEC and TLS, then it shall be tested with each of these protocols:

a) Test 1a: The evaluator shall present the TOE with a valid chain of certificates (terminating in a trusted CA certificate) as needed to validate the leaf certificate to be used in the function and shall use this chain to demonstrate that the function succeeds. Test 1a shall be designed in a way that the chain can be "broken" in Test 1b by either being able to remove the trust anchor from the TOEs trust store, or by setting up the trust store in a way that at least one intermediate CA certificate needs to be provided, together with the leaf certificate from outside the TOE, to complete the chain (e.g. by storing only the root CA certificate in the trust store).

b) Test 1b: The evaluator shall then "break" the chain used in Test 1a by either removing the trust anchor in the TOE's trust store used to terminate the chain, or by removing one of the intermediate CA certificates (provided together with the leaf certificate in Test 1a) to complete the chain. The evaluator shall show that an attempt to validate this broken chain fails.

c) Test 2: The evaluator shall demonstrate that validating an expired certificate results in the function failing.

d) Test 3: The evaluator shall test that the TOE can properly handle revoked certificates-conditional on whether CRL or OCSP is selected; if both are selected, then a test shall be performed for each method. The evaluator shall test revocation of the peer certificate and revocation of the peer intermediate CA certificate i.e. the intermediate CA certificate should be revoked by the root CA. The evaluator shall 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. Revocation checking is only applied to certificates that are not designated as trust anchors. Therefore, the revoked certificate(s) used for testing shall not be a trust anchor.

e) Test 4: If OCSP 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.

f) 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.)

g) *Test 6: The evaluator shall modify any byte in the certificate signatureValue field (see RFC5280 Sec. 4.1.1.3), which is normally the last field in the certificate, and demonstrate that the certificate fails to validate. (The signature on the certificate will not validate.)*

h) *Test 7: The evaluator shall modify any byte in the public key of the certificate and demonstrate that the certificate fails to validate. (The hash of the certificate will not validate.)*

i) *Test 8: [TD0527] (Conditional on support for EC certificates as indicated in FCS_COP.1/SigGen). 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. The evaluator shall replace the intermediate certificate in the certificate chain for Test 8 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 8, 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.*

The evaluator shall perform the following tests for FIA_X509_EXT.1.2/Rev. The tests described must be performed in conjunction with the other certificate services assurance activities, including the functions in FIA_X509_EXT.2.1/Rev. The tests for the extendedKeyUsage rules are performed in conjunction with the uses that require those rules. Where the TSS identifies any of the rules for extendedKeyUsage fields (in FIA_X509_EXT.1.1) that are not supported by the TOE (i.e. where the ST is therefore claiming that they are trivially satisfied) then the associated extendedKeyUsage rule testing may be omitted.

The goal of the following tests is to verify that the TOE accepts a certificate as a CA certificate only if it has been marked as a CA certificate by using basicConstraints with the CA flag set to True (and implicitly tests that the TOE correctly parses the basicConstraints extension as part of X509v3 certificate chain validation).

For each of the following tests the evaluator shall create a chain of at least three certificates: a self-signed root CA certificate, an intermediate CA certificate and a leaf (node) certificate. The properties of the certificates in the chain are adjusted as described in each individual test below (and this modification shall be the only invalid aspect of the relevant certificate chain).

a) *Test 1: The evaluator shall ensure that at least one of the CAs in the chain does not contain the basicConstraints extension. The evaluator confirms that the TOE rejects such a certificate at one (or both) of the following points: (i) as part of the validation of the leaf certificate belonging to this chain; (ii) when attempting to add a CA certificate without the basicConstraints extension to the TOE's trust store (i.e. when attempting to install the CA certificate as one which will be retrieved from the TOE itself when validating future certificate chains).*

b) *Test 2: The evaluator shall ensure that at least one of the CA certificates in the chain has a basicConstraints extension in which the CA flag is set to FALSE. The evaluator confirms that the TOE rejects such a certificate at one (or both) of the following points: (i) as part of the validation of the leaf certificate belonging to this chain; (ii) when attempting to add a CA certificate with the CA flag set to FALSE to the TOE's trust store (i.e. when attempting to install the CA certificate as one which will be retrieved from the TOE itself when validating future certificate chains).*

The evaluator shall repeat these tests for each distinct use of certificates. Thus, for example, use of certificates for TLS connection is distinct from use of certificates for trusted updates so both of these uses would be tested. But there is no need to repeat the tests for each separate TLS channel in FTP_ITC.1 and FTP_TRP.1/Admin (unless the channels use separate implementations of TLS).

Summary

Test 1a:
The evaluator used a computer running Kali Linux. The evaluator created a root CA certificate, two intermediate CAs, a server certificate for openssl s_server server and a client certificate for the TOE and installed the certificates accordingly.The evaluator configured the TOE to act as a TLS client using a Virtual Server and Client SSL profile, as per [K14783]. The evaluator used Wireshark to record and analyze the traffic between the s_server and the TOE. The analysis showed that the connection could be established (as expected).

Test 1b:

The evaluator removed one of the intermediate CA certificates and configured the openssl s_server for TLS communication with the TOE. The evaluator configured the TOE to act as a TLS client using a Virtual Server and Client SSL profile, as per [K14783]. The evaluator used Wireshark to record and analyze the traffic between the s_server and the TOE. The analysis showed that the connection could not be established (as expected).

Test 2:
The evaluator used a computer running Kali Linux. The evaluator created a root CA certificate, two intermediate CAs, an expired server certificate for openssl s_server and a client certificate for the TOE and installed the certificates accordingly. The evaluator configured the TOE to act as a TLS client using a Virtual Server and Client SSL profile, as per [K14783]. The evaluator used Wireshark to record and analyze the traffic between the s_server and the TOE. The analysis showed that the connection failed (as expected).

Test 3:
The evaluator used a computer running Kali Linux. The evaluator created a root CA certificate, one revoked intermediate CA, a CRL certificate and a server certificate and installed the certificates accordingly. The evaluator configured an openssl s_server and also configured the TOE to act as a TLS client using a Virtual Server and Client SSL profile, as per [K14783]. The evaluator checked if the connection would be established, but it failed (as expected). The evaluator used a computer running Kali Linux. The evaluator created a root CA certificate, one intermediate CA, a revoked server certificate and a CRL certificate and installed the certificates accordingly. The evaluator configured an openssl s_server and also configured the TOE to act as a TLS client using a Virtual Server and Client SSL profile, as per [K14783]. The evaluator checked if the connection would be established, but it failed (as expected).

Test 4:
The evaluator used a computer running Kali Linux. The evaluator created a root CA certificate, two intermediate CAs, a CRL certificate which has been signed by a CA without cRLsign and installed the certificates accordingly. The evaluator configured an openssl s_server and also configured the TOE to act as a TLS client using a Virtual Server and Client SSL profile, as per [K14783]. The evaluator checked if the validation of the CRL would be successful, but it failed (as expected).

Test 5:
The evaluator used a computer running Kali Linux. The evaluator created a root CA certificate, two intermediate CAs, a server certificate for openssl s_server and a client certificate for the TOE and installed the certificates accordingly. The evaluator modified any byte in the first eight bytes of the server certificate. The evaluator configured an openssl s_server and also configured the TOE to act as a TLS client using a Virtual Server and Client SSL profile, as per [K14783]. The evaluator checked if the connection would be established, but it failed (as expected).

Test 6:
The evaluator used a computer running Kali Linux. The evaluator created a root CA certificate, two intermediate CAs, a server certificate for openssl s_server and a client certificate for the TOE and installed the certificates accordingly. The evaluator modified any byte in the last byte of the server certificate. The evaluator configured an openssl s_server and also configured the TOE to act as a TLS client using a Virtual Server and Client SSL profile, as per [K14783]. The evaluator checked if the connection would be established, but it failed (as expected).

Test 7:
The evaluator used a computer running Kali Linux. The evaluator created a root CA certificate, two intermediate CAs, a server certificate for openssl s_server and a client certificate for the TOE and installed the certificates accordingly. The evaluator modified any byte in the last byte of the server public key. The evaluator configured an openssl s_server and also configured the TOE to act as a TLS client using a Virtual Server and Client SSL profile, as per [K14783]. The evaluator checked if the connection would be established, but it failed (as expected).

a) Test 1:
The evaluator used a computer running Kali Linux. The evaluator created a root CA certificate, one intermediate CA without the basic constraints extension, a second intermediate CA with the basic constraints extension, a server certificate for openssl s_server and a client certificate for the TOE and installed the certificates accordingly. The evaluator configured an openssl s_server and also configured the TOE to act as a TLS client using a Virtual Server and Client SSL profile, as per [K14783]. The evaluator checked if the connection would be established when providing the intermediate certificate without basic constraint as part of the chain, but it failed (as expected).

b) Test 2:
The evaluator used a computer running Kali Linux. The evaluator created a root CA certificate, one intermediate CA with the CA flag set to false, a second intermediate CA, a server certificate for the openssl s_server and a client certificate for the TOE and installed the certificates accordingly. The evaluator configured an openssl s_server and also configured the TOE to act as a TLS client using a Virtual Server and Client SSL profile, as per [K14783]. The evaluator checked if the connection would be established when providing the intermediate certificate with basic constraint set to FALSE as part of the chain, but it failed (as expected).

2.1.3.7 X.509 Certificate Authentication (FIA_X509_EXT.2)

TSS Assurance Activities

Assurance Activity AA-FIA_X509_EXT.2-ASE-01

> *The evaluator shall check the TSS to ensure that it describes how the TOE chooses which certificates to use, and any necessary instructions in the administrative guidance for configuring the operating environment so that the TOE can use the certificates.*
>
> *The evaluator shall examine the TSS to confirm that it describes the behaviour of the TOE when a connection cannot be established during the validity check of a certificate used in establishing a trusted channel. The evaluator shall verify that any distinctions between trusted channels are described. If the requirement that the administrator is able to specify the default action, then the evaluator shall ensure that the guidance documentation contains instructions on how this configuration action is performed.*

Summary

Chapter 7 of [ST] contains the TSS. Section 7.3.2 *Certificate Validation* describes how the TOE perform certificate validation for TLS and HTTPS sessions. It states:

- The TOE implements certificate validation using the OpenSSL crypto library for TLS and HTTP sessions.
- The TOE validates X.509 certificates using a certificate revocation list (CRL) as specified in RFC5280 Section 5. Administrators create profiles which are used to define the parameters used to communicate with an external entity. These parameters include the ability to require the use of TLS and peer or mutual authentication and a definition of the certificate to use for authentication. This capability is used to create a mutually authenticated connection with the external audit server. The external audit server provides a certificate to the TOE during establishment of the TLS connection in order to authenticate the external audit server.
- The TOE offers administrative interfaces for creating a private key and certificate signing request (CSR). The CSR may include the following information: public key, common name, organization, organizational unit, country, locality, state / province, country, e-mail address, subject alternative name. After the CRS is created, the administrator must export the CSR

outside the TOE. Outside the scope of the TOE, the administrator provides the CSR to the CA and then the CA returns the certificate to the administrator. Using the administrative interface, the administrator can then import the certificate into the TOE.

- The only method supported by the TOE for obtaining a CA certificate is for the administrator to save a certificate to a text file and import it into the TOE. The certificates are stored in a text file. The TOE is capable of importing X.509v3 certificates and certificates in the PKCS12 format. The TOE is also capable of creating and using a self-signed certificate.
- The TOE checks the validity of the certificates when the profile using the certificate is loaded and when the certificate is used by the TOE, including during authentication. If the certificates are modified, the digital signature verification would detect that the certificate had been tampered with and the certificate would be invalid. Administrators can ensure that the certificates presented have not been revoked by importing a certificate revocation list (CRL) into the TOE.
- A certificate chain includes the root CA certificate, certificates of intermediate CAs, and the end entity certificate. The certificate chain consists of all the certificates necessary to validate the end certificate. Administrators can upload trusted device certificates (root CA certificates) into the TOE to identify which certificates are trusted. The TOE performs full certificate chain checking using Public Key Infrastructure X.509, verifies the expiration of the certificate (assuming a reliable time), and verifies its revocation using CRLs.

The evaluator found the description above provided in the TSS does explicitly states that when the validity of a certificate cannot be established, the TOE will allow the administrator to choose whether or not to accept the certificate. The evaluator also conformed that the TSS makes no distinctions between TLS and HTTPS. It was indeed made clear in the TSS that the description apply to both trusted channels.

While performing the evaluation of guidance evaluation, the evaluator will determined that sufficient guidance is provided to administrator to configure TLS and HTTPS connections including configuring certificate validation.

Guidance Assurance Activities

Assurance Activity AA-FIA_X509_EXT.2-AGD-01

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

Summary

The evaluator examined section 2 "Installation and Configuration Procedures" of [ECG] . In particular section 2.3.9 "Certificate Management" provides guidance on managing X.509 certificates with references to the user guides

- Device certificate overview: [K15664] K15664: Overview of BIG-IP device certificates (11.x - 16.x)
- SSL certificate management: SSL Certificate Management section of [SSLADM] BIG-IP System: SSL Administration The same document contains sections on creating and requesting certificates, SSL traffic management, and configuring client- and server-side traffic.
- Certificate management through the GUI: [K14620] K14620: Manage SSL certificates for BIG-IP systems using the Configuration utility

- Certificate management through the tmsh: [K15462] K15462: Managing SSL certificates for BIG-IP systems using tmsh

The section also states to ensure that the revocation of intermediate certificates causes a connection to fail, the intermediate CAs must NOT be in Trusted Certificate Authorities. Thus, when configuring the SSL profile, follow the instructions in follow the instructions in [K14806] K14806: Overview of the Server SSL profile (11.x – 17.x), [K14783] K14783: Overview of the Client SSL profile (11.x – 17.x), and [K13302] K13302: Configuring the BIG-IP system to use an SSL chain certificate (11.x – 16.x) to define only the root CA as the trust anchor.

Test Assurance Activities

Assurance Activity AA-FIA_X509_EXT.2-ATE-01

> *The evaluator shall perform the following test for each trusted channel:*
>
> *The evaluator shall demonstrate that using a valid certificate that requires certificate validation checking to be performed in at least some part by communicating with a non-TOE IT entity. The evaluator shall then manipulate the environment so that the TOE is unable to verify the validity of the certificate and observe that the action selected in FIA_X509_EXT.2.2 is performed. If the selected action is administrator-configurable, then the evaluator shall follow the guidance documentation to determine that all supported administrator-configurable options behave in their documented manner.*

Summary

The evaluator used a computer running Kali Linux. The evaluator created a CA certificate, a server certificate for the openssl s_server server and a client certificate for the TOE and installed the certificates accordingly. The evaluator configured an openssl s_server and also configured the TOE to act as a TLS client using a Virtual Server and Client SSL profile, as per [K14783]. The evaluator used Wireshark to record and analyze the traffic between the openssl s_server and the TOE. The analysis showed that the connection was established.

The evaluator configured an openssl s_server and also configured the TOE to act as a TLS client without the root certificate using a Virtual Server and Client SSL profile, as per [K14783]. The evaluator used Wireshark to record and analyze the traffic between the openssl s_server and the TOE. The analysis showed that the connection could not be established (as expected).

2.1.3.8 X509 Certificate Requests (FIA_X509_EXT.3)

TSS Assurance Activities

Assurance Activity AA-FIA_X509_EXT.3-ASE-01

> *If the ST author selects "device-specific information", the evaluator shall verify that the TSS contains a description of the device-specific fields used in certificate requests.*

Summary

Section 6.2.3.8 *FIA_X509_EXT.3 X.509 Certificate Requests* of [ST] defines FIA_X509_EXT.3 as follows:

> *The TSF shall generate a Certificate Request Message as specified by RFC 2986 and be able to provide the following information in the request: public key and [Common Name, Organization, Organizational Unit, Country].*

As seen above, the evaluator determined that the ST does not select "device-specific information", thus, no corresponding description in the TSS is required.

Guidance Assurance Activities

Assurance Activity AA-FIA_X509_EXT.3-AGD-01

The evaluator shall check to ensure that the guidance documentation contains instructions on requesting certificates from a CA, including generation of a Certificate Request. If the ST author selects "Common Name", "Organization", "Organizational Unit", or "Country", the evaluator shall ensure that this guidance includes instructions for establishing these fields before creating the Certification Request.

Summary

The evaluator examined section 2 "Installation and Configuration Procedures" of [ECG] . In particular section 2.3.9 "Certificate Management" provides guidance on managing X.509 certificates with references to the user guides [K15664] , [K14620] , [K15462] , [K14806] , [K14783] , and [SSLADM] . The evaluator examined these user guides which provide detailed instructions such as to request a certificate from a CA, generation of a Certificate Request Message including instructions for establishing the fields specified in [ST] such as "Common Name", "Organization", and "Country".

Test Assurance Activities

Assurance Activity AA-FIA_X509_EXT.3-ATE-01

The evaluator shall perform the following tests:

a) Test 1: The evaluator shall use the guidance documentation to cause the TOE to generate a Certification Request. The evaluator shall capture the generated message and ensure that it conforms to the format specified. The evaluator shall confirm that the Certification Request provides the public key and other required information, including any necessary user-input information.

b) Test 2: The evaluator shall demonstrate that validating a response message to a Certification Request without a valid certification path results in the function failing. The evaluator shall then load a certificate or certificates as trusted CAs needed to validate the certificate response message and demonstrate that the function succeeds.

Summary

Test 1 :
The evaluator used a computer running Kali Linux. The evaluator authenticated to the TOE via graphical user interface and created a CSR certificate in the TOE. The evaluator verified that the CSR certificate conforms to the specified format.

Test 2 :
The evaluator used a computer running Kali Linux. The evaluator created a root CA certificate, two intermediate CAs certificates , and a server certificate for the openssl s_server. The evaluator installed the CA certificates on the TOE. The evaluator configured an openssl s_server and also configured the TOE to act as a TLS client using a Virtual Server and Client SSL profile, as per [K14783], using the CA bundle with a broken chain i.e, without the second intermediate CA certificate in the bundle. The evaluator used Wireshark to record and analyze the traffic between the openssl s_server and the TOE. The analysis showed that the connection could not be established (as expected).
The evaluator installed all the certificates accordingly. The evaluator configured the TOE to act as a TLS client using a Virtual Server and Client SSL profile, as per [K14783], using the complete bundle. The evaluator used Wireshark to record and analyze the traffic between the openssl s_server and the TOE. The analysis showed that the connection could be established.

2.1.4 Security management (FMT)

2.1.4.1 Management of Security Functions Behaviour (FMT_MOF.1/ManualUpdate)

TSS Assurance Activities

Assurance Activity AA-FMT_MOF.1-MU-ASE-01

There are no specific requirements for non-distributed TOEs.

Summary

According to the Security Target [ST] , the TOE is not distributed.

Assurance Activity AA-FMT_MOF.1-MU-ASE-02

For distributed TOEs it is required to verify the TSS to ensure that it describes how every function related to security management is realized for every TOE component and shared between different TOE components. The evaluator shall confirm that all relevant aspects of each TOE component are covered by the FMT SFRs.

Summary

According to the Security Target [ST] , the TOE is not distributed therefore this Evaluation Activity is not applicable.

Guidance Assurance Activities

Assurance Activity AA-FMT_MOF.1-MU-AGD-01

The evaluator shall examine the guidance documentation to determine that any necessary steps to perform manual update are described. The guidance documentation shall also provide warnings regarding functions that may cease to operate during the update (if applicable).

Summary

Section 2.2.2 "Re-install the BIG-IP software" of [ECG] provides relevant guidance for performing trusted updates. Section 2.2.2.3 "Updating BIG-IP software after initial configuration" contains an explicit statement:

> *For hardware and vCMP, the process of updating BIG-IP is the same as the initial install, except the administrator does not need to verify the image.*

Since the ccmode command has already been run during the initial install, the BIG-IP will automatically verify the new ISO using the digital signature as part of the upload and installation process initiated by the administrative-user.

The initial installation as described in section 2.2.2 involves the administrator manually downloading the appropriate TOE image filest from F5 support website and verifying the download using digital signature prior to installation.

Section 2.2.2 "Re-install the BIG-IP software" of [ECG] also refers to [SWUPDATE] which has the instructions on updating BIG-IP VE. A note in the paragraph indicates that [SWUPDATE] only describes validating the download with the MD5 checksum; digital signature files are available for validation and that validation is performed as described for the image file download in section 2.2.2.2 Verifying the product ISO using the digital signature.

Assurance Activity AA-FMT_MOF.1-MU-AGD-02

For distributed TOEs the guidance documentation shall describe all steps how to update all TOE components. This shall contain description of the order in which components need to be updated if the order is relevant to the update process. The guidance documentation shall also provide warnings regarding functions of TOE components and the overall TOE that may cease to operate during the update (if applicable).

Summary

According to [ST] , the TOE is not distributed, therefore this assurance activity does not apply.

Test Assurance Activities

Assurance Activity AA-FMT_MOF.1-MU-ATE-01

The evaluator shall try to perform the update using a legitimate update image without prior authentication as Security Administrator (either by authentication as a user with no administrator privileges or without user authentication at all depending on the configuration of the TOE). The attempt to update the TOE shall fail.

The evaluator shall try to perform the update with prior authentication as Security Administrator using a legitimate update image. This attempt should be successful. This test case should be covered by the tests for FPT_TUD_EXT.1 already.

Summary

The evaluator used a computer running Ubuntu Linux as a HTTPS client. The evaluator authenticated as a user without administrator privileges. The evaluator checked if a software update was possible but failed (as expected).

The evaluator used a computer running Ubuntu Linux as a HTTPS client. The evaluator authenticated as a user with administrator privileges. The evaluator performed a software update and succeeded. This test was performed as part of FPT_TUD_EXT.1

2.1.4.2 Management of Security Functions Behaviour (Services) (FMT_MOF.1/Services)

TSS Assurance Activities

Assurance Activity AA-FMT_MOF.1-SRV-ASE-01

For distributed TOEs it is required to verify the TSS to ensure that it describes how every function related to security management is realized for every TOE component and shared between different TOE components. The evaluator shall confirm that all relevant aspects of each TOE component are covered by the FMT SFRs.

Summary

According to the Security Target [ST] , the TOE is not distributed therefore this Evaluation Activity is not applicable.

Assurance Activity AA-FMT_MOF.1-SRV-ASE-02

For non-distributed TOEs, the evaluator shall ensure the TSS lists the services the Security Administrator is able to start and stop and how that operation is performed.

Summary

Section 7.4 in the TSS of [ST] provides the related description. It states the following:

- The Security Administrator is able to start and stop the following services using the “bigstart <stop, start, restart> <service>” command or the following tmsh command “tmsh <stop, start, restart> /sys service <service>”
- The list of services that can be started and stopped are found in https://support.f5.com/csp/article/K67197865

The evaluator examined the referenced article and identified a list of services (i.e., BIP-IP daemons) along with their descriptions which perform a variety of functions.

Guidance Assurance Activities

Assurance Activity AA-FMT_MOF.1-SRV-AGD-01

For distributed TOEs it is required to verify the Guidance Documentation to describe management of each TOE component. The evaluator shall confirm that all relevant aspects of each TOE component are covered by the FMT SFRs.

Summary

This assurance activity is not applicable and therefore considered to be satisfied.

Assurance Activity AA-FMT_MOF.1-SRV-AGD-02

For non-distributed TOEs, the evaluator shall also ensure the Guidance Documentation describes how the TSS lists the services the Security Administrator is able to start and stop and how that how that operation is performed.

Summary

The evaluator examined section 3 "Operational Procedures" of [ECG] . In particular section 3.9 "Starting and Stopping Services" provides guidance on starting and stopping services with reference to the user guide [K48615077] K48615077: BIG-IP Daemons (15.x – 16.x). Section 3.9 states that the scurity administrator is able to stop, start and restart services with either the bigstart command or the tmsh command. The list of services that can be started and stopped, and the details for doing so, are found in [K48615077] K48615077: BIG-IP Daemons (15.x – 16.x).

Test Assurance Activities

Assurance Activity AA-FMT_MOF.1-SRV-ATE-01

The evaluator shall examine the TSS, Guidance Documentation and the TOE as observed during all other testing and shall confirm that the management functions specified in FMT_SMF.1 are provided by the TOE. The evaluator shall confirm that the TSS details which security management functions are available through which interface(s) (local administration interface, remote administration interface).

The evaluator shall try to enable and disable at least one of the services as defined in the Application Notes for FAU_GEN.1.1 (whichever is supported by the TOE) without prior authentication as Security Administrator (either by authenticating as a user with no administrator privileges, if possible, or without prior authentication at all). The attempt to enable/disable this service/these services should fail. According to the implementation no other users than the Security Administrator might be defined and without any user authentication the user might not be able to get to the point where the attempt to enable/disable this service/these services can be executed. In that case it shall be demonstrated that access control mechanisms prevent execution up to the step that can be reached without authentication as Security Administrator.

The evaluator shall try to enable and disable at least one of the services as defined in the Application Notes for FAU_GEN.1.1 (whichever is supported by the TOE) with prior authentication as Security Administrator. The attempt to enable/disable this service/these services should be successful.

Summary

The evaluator used a computer running Ubuntu Linux as a HTTPS client. The evaluator authenticated as user without administrator privileges. The evaluator checked if enabling or disabling the TOE services was possible but failed (as expected).

The evaluator used a computer running Ubuntu Linux as a HTTPS client. The evaluator authenticated as user with administrator privileges. The evaluator checked if enabling or disabling the TOE services was possible and succeeded.

2.1.4.3 Management of TSF Data (FMT_MTD.1/CoreData)

TSS Assurance Activities

Assurance Activity AA-FMT_MTD.1-CD-ASE-01

> *The evaluator shall examine the TSS to determine that, for each administrative function identified in the guidance documentation; those that are accessible through an interface prior to administrator log-in are identified. For each of these functions, the evaluator shall also confirm that the TSS details how the ability to manipulate the TSF data through these interfaces is disallowed for non-administrative users.*
>
> *If the TOE supports handling of X.509v3 certificates and implements a trust store, the evaluator shall examine the TSS to determine that it contains sufficient information to describe how the ability to manage the TOE's trust store is restricted.*

Summary

Chapter 7 of [ST] contains the TSS. Section 7.4 *Security Function Management* describes how the TOE manage security functions. It states that the TOE can be administered both locally and remotely. Local administration is performed via direct connection to the management port on the device via Ethernet cable. Remote administration to the management interface is only through dedicated management network ports of the device. The TOE offers the following four different methods to configure and manage the TSF:

- Configuration Utility (Web-based GUI) - browser-based GUI interface with normal GUI panels and selections. The client browser talks to the Apache HTTP server over HTTPS; then the request passes through tomcat and to the BIG-IP.
- tmsh shell commands – provide a command line interface, accessible through an SSH client or a serial port console.
- iControl API – SOAP-based programming interface over HTTPS.
- iControl REST API – REST-based programming interface over HTTPS.

Also, section 7.4 contains the following statement:

> *These four administrative interfaces require users to identify and authenticate themselves prior to performing any administrative functions.*

In order to manipulate the TSF data, the users (namely, administrative users) must be successfully authenticated (via the authentication methods identified above) through these interfaces. Additionally, access of users to these interfaces is restricted based on predefined roles which are described in table 9 "BIG-IP User Roles" of [ST] . These roles, which cannot be changed by the TOE administrators, define which types of tasks the authorized users can perform.

Per FMT_SMF.1 defined in section 6.2.4.5 of [ST] , the TOE does not implement a trust store or designate X509.v3 certificates as trust anchors, thus no corresponding TSS description is required.

Assurance Activity AA-FMT_MTD.1-CD-ASE-02

For distributed TOEs it is required to verify the TSS to ensure that it describes how every function related to security management is realized for every TOE component and shared between different TOE components. The evaluator shall confirm that all relevant aspects of each TOE component are covered by the FMT SFRs.

Summary

According to the Security Target [ST] , the TOE is not distributed therefore this Evaluation Activity is not applicable.

Guidance Assurance Activities

Assurance Activity AA-FMT_MTD.1-CD-AGD-01

The evaluator shall review the guidance documentation to determine that each of the TSF-data-manipulating functions implemented in response to the requirements of the cPP is identified, and that configuration information is provided to ensure that only administrators have access to the functions.

If the TOE supports handling of X.509v3 certificates and provides a trust store, the evaluator shall review the guidance documentation to determine that it provides sufficient information for the administrator to configure and maintain the trust store in a secure way. If the TOE supports loading of CA certificates, the evaluator shall review the guidance documentation to determine that it provides sufficient information for the administrator to securely load CA certificates into the trust store. The evaluator shall also review the guidance documentation to determine that it explains how to designate a CA certificate a trust anchor.

Summary

While examining sections 2 and 3 of [ECG] , the evaluator determined that the TSF-data-manipulating functions implemented by the TOE in response to the requirements of [NDcPPv2.2e] are identified and described throughout sections 3 and 4. The evaluator examined the administrative guide as part of the activities associated with ensuring the assurance activities defined for the SFRs are satisfied.

The evaluator also determined that only security administrators have the privileges to manage the TSF data through the TOE functionality.

Section 2.3.9 "Certificate Management" of [ECG] provides guidance for managing X.509 certificates using for TLS communications. This section also refers to the following guidance documents for further details/instructions:

- [K15664] : Overview of BIG-IP device certificates (11.x - 16.x)
- [SSLADM] : BIG-IP System: SSL Administration (section "SSL Certificate Management")
- [K14620] : Manage SSL certificates for BIG-IP systems using the Configuration utility
- [K15462] : K15462: Managing SSL certificates for BIG-IP systems using tmsh
- [K14806] : Overview of the Server SSL profile (11.x - 17.x)
- [K14783] : Overview of the Client SSL profile (11.x - 17.x)
- [K13302] : Configuring the BIG-IP system to use an SSL chain certificate (11.x - 16.x)

The evaluator examined both the guidance provided in [ECG] and the referenced supplemental guides and found that they contain sufficient information for managing certificates used for TLS communications.

Test Assurance Activities

Assurance Activity AA-FMT_MTD.1-CD-ATE-01

No separate testing for FMT_MTD.1/CoreData is required unless one of the management functions has not already been exercised under any other SFR.

Summary

The evaluator performed the test as part of other SFR, FMT_SMF.1.

2.1.4.4 Management of TSF Data (FMT_MTD.1/CryptoKeys)

TSS Assurance Activities

Assurance Activity AA-FMT_MTD.1-CK-ASE-01

For distributed TOEs it is required to verify the TSS to ensure that it describes how every function related to security management is realized for every TOE component and shared between different TOE components. The evaluator shall confirm that all relevant aspects of each TOE component are covered by the FMT SFRs.

Summary

According to the Security Target [ST] , the TOE is not distributed therefore this Evaluation Activity is not applicable.

Assurance Activity AA-FMT_MTD.1-CK-ASE-02

For non-distributed TOEs, the evaluator shall ensure the TSS lists the keys the Security Administrator is able to manage to include the options available (e.g. generating keys, importing keys, modifying keys or deleting keys) and how that how those operations are performed.

Summary

Section 7.4.1 *Security Roles* in the TSS of [ST] states the following:

- The "tmsh sys crypto key" command can be used by the System Administrator to manage cryptographic keys on the TOE.
- The Security Administrator can manage the SSH key pairs, TLS key pairs, and pre-shared keys.
- The Security Administrator is able to perform the following operations on the keys:
  - Importing of SSH public keys
  - Generating SSL keys
  - Changing keys
  - Deleting keys
  - Installing keys

The evaluator verified that the TSS contains the required description.

Guidance Assurance Activities

Assurance Activity AA-FMT_MTD.1-CK-AGD-01

For distributed TOEs it is required to verify the Guidance Documentation to describe management of each TOE component. The evaluator shall confirm that all relevant aspects of each TOE component are covered by the FMT SFRs.

Summary

This assurance activity is not applicable and therefore considered to be satisfied.

Assurance Activity AA-FMT_MTD.1-CK-AGD-02

> *For non-distributed TOEs, the evaluator shall also ensure the Guidance Documentation lists the keys the Security Administrator is able to manage to include the options available (e.g. generating keys, importing keys, modifying keys or deleting keys) and how that how those operations are performed.*

Summary

Section 7.4.1 "Security Roles" in the ST states that: a) the "tmsh sys crypto key" command can be used to manage the crypto keys; b) the keys that can be managed are SSH key pairs, TLS key pairs and pre-shared keys; c) operations can be performed are

- Importing of SSH public keys
- Generating SSL keys
- Changing keys
- Deleting keys
- Installing keys

The evaluator examined section 2 "Installation and Configuration Procedures" of [ECG]. In particular section 2.3.9 "Certificate Management" provides guidance on managing certificates with references to the user guides [K15664] , [K14620] , [K15462] , [K14806] , [K14783] , and [SSLADM] . The evaluator examined these user guides and in particular [K15462] "15462: Managing SSL certificates for BIG-IP systems using tmsh". The guide provides tmsh commands along with examples on how to import, create, delete, change, and install SSL certificates/keys.

Test Assurance Activities

Assurance Activity AA-FMT_MTD.1-CK-ATE-01

> *The evaluator shall examine the TSS, Guidance Documentation and the TOE as observed during all other testing and shall confirm that the management functions specified in FMT_SMF.1 are provided by the TOE. The evaluator shall confirm that the TSS details which security management functions are available through which interface(s) (local administration interface, remote administration interface).*
>
> *The evaluator shall try to perform at least one of the related actions (modify, delete, generate/import) without prior authentication as Security Administrator (either by authentication as a non-administrative user, if supported, or without authentication at all). Attempts to perform related actions without prior authentication should fail. According to the implementation no other users than the Security Administrator might be defined and without any user authentication the user might not be able to get to the point where the attempt to manage cryptographic keys can be executed. In that case it shall be demonstrated that access control mechanisms prevent execution up to the step that can be reached without authentication as Security Administrator.*
>
> *The evaluator shall try to perform at least one of the related actions with prior authentication as Security Administrator. This attempt should be successful.*

Summary

The evaluator used a computer running Ubuntu Linux as a HTTPS client. The evaluator authenticated as user without administrator privileges. The evaluator checked if security management actions (import, creation, deletions) of cryptographic keys were available to a non administrator user but failed (as expected). The evaluator used a computer running Ubuntu Linux as a HTTPS client. The evaluator authenticated as user with administrator privileges. The evaluator checked if security management actions (import, creation, deletions) of cryptographic keys were available to a administrator user and succeeded.

2.1.4.5 Specification of Management Functions (FMT_SMF.1)

TSS Assurance Activities

Assurance Activity AA-FMT_SMF.1-ASE-01

The security management functions for FMT_SMF.1 are distributed throughout the cPP and are included as part of the requirements in FTA_SSL_EXT.1, FTA_SSL.3, FTA_TAB.1, FMT_MOF.1/ManualUpdate, FMT_MOF.1/AutoUpdate (if included in the ST), FIA_AFL.1, FIA_X509_EXT.2.2 (if included in the ST), FPT_TUD_EXT.1.2 & FPT_TUD_EXT.2.2 (if included in the ST and if they include an administrator-configurable action), FMT_MOF.1/Services, and FMT_MOF.1/Functions (for all of these SFRs that are included in the ST), FMT_MTD, FPT_TST_EXT, and any cryptographic management functions specified in the reference standards. Compliance to these requirements satisfies compliance with FMT_SMF.1.

Summary

Chapter 7 of [ST] contains the TSS. Section 7.4 *Security Function Management* describes how the TOE manage security functions.

While performing Evaluation Activities for other SFRs, the evaluator also verified that appropriate management is also performed for those relevant SFRs.

Assurance Activity AA-FMT_SMF.1-ASE-02

The evaluator shall examine the TSS, Guidance Documentation and the TOE as observed during all other testing and shall confirm that the management functions specified in FMT_SMF.1 are provided by the TOE. The evaluator shall confirm that the TSS details which security management functions are available through which interface(s) (local administration interface, remote administration interface).

The evaluator shall examine the TSS and Guidance Documentation to verify they both describe the local administrative interface. The evaluator shall ensure the Guidance Documentation includes appropriate warnings for the administrator to ensure the interface is local.

Summary

Chapter 7 of [ST] contains the TSS. Section 7.4 *Security Function Management* which states the following:

- The TOE provides the ability to administer the TOE both locally and remotely.
- Local administration is performed via the serial port console.
- Remote access to the management interfaces is only made available on the dedicated management network port of a BIG-IP system which offers four different methods:
  - Configuration Utility (Web-based GUI) - browser-based GUI interface with normal GUI panels and selections. The client browser talks to the Apache HTTP server over HTTPS; then the request passes through tomcat and to the BIG-IP.
  - tmsh shell commands – provide a command line interface, accessible through an SSH client
  - iControl API – SOAP-based programming interface over HTTPS.
  - iControl REST API – REST-based programming interface over HTTPS.

Additional description of the local interface is provided in the TSS section 7.6, *TOE Access* and section 7.3, *Identification and Authentication* .

In addition, sction 7.4 lists the security functions available through therese interfaces including the ability to administer the TOE locally and remotely and ability to update the TOE/verify the udpates as required by FMT_SMF.1 from [NDcPPv2.2e] .

Furthermore, while performimg guidance evaluation the evaluator will verify that the guidance documentation provides information about the local administrative intrface including setting up the serial console.

Assurance Activity AA-FMT_SMF.1-ASE-03

For distributed TOEs with the option "ability to configure the interaction between TOE components" the evaluator shall examine that the ways to configure the interaction between TOE components is detailed in the TSS and Guidance Documentation. The evaluator shall check that the TOE behaviour observed during testing of the configured SFRs is as described in the TSS and Guidance Documentation.

Summary

According to the Security Target [ST] , the TOE is not distributed therefore this Evaluation Activity is not applicable.

Assurance Activity AA-FMT_SMF.1-ASE-04

For distributed TOEs it is required to verify the TSS to ensure that it describes how every function related to security management is realized for every TOE component and shared between different TOE components. The evaluator shall confirm that all relevant aspects of each TOE component are covered by the FMT SFRs.

Summary

According to the Security Target [ST] , the TOE is not distributed therefore this Evaluation Activity is not applicable.

Guidance Assurance Activities

Assurance Activity AA-FMT_SMF.1-AGD-01

The evaluator shall examine the TSS, Guidance Documentation and the TOE as observed during all other testing and shall confirm that the management functions specified in FMT_SMF.1 are provided by the TOE. The evaluator shall confirm that the TSS details which security management functions are available through which interface(s) (local administration interface, remote administration interface).

The evaluator shall examine the TSS and Guidance Documentation to verify they both describe the local administrative interface. The evaluator shall ensure the Guidance Documentation includes appropriate warnings for the administrator to ensure the interface is local.

Summary

While performing other assurance activities, especially the testing activities, the evaluator made sure that the assurance activities on the following management SFRs are fulfilled: FTA_TAB.1, FTA_SSL.3, FIA_X509_EXT.2.2, FMT_MOF.1/Services, FMT_MOF.1/ManualUpdate, FMT_MTD/CoreData, FMT_MTD/CryptoKeys, and FPT_TUD_EXT.1.2.

Local administrative interface is described in the following sections of [ECG] :

- section 2.3.3.1 "Administrative users": this section describes how to configure local administrative users.
- section 2.3.12 "Session Inactivity Termination": this section describes how to configure session inactivity termination for local administrative user sessions including for Virtual Edition environments of the TOE.
- section 2.2.5 "Login welcome banners": this section provides instructions for setting up the banner for the GUI and tmsh sessions.

- section 3.2 "Maximum Failed Login Attempts": this section provides guidance for configuring maximum failed login attempts for all interfaces including the local administrative interface.

The TSS of [ST] in section 7.3 "Identification and Authentication" and section 7.4 "Security Function Management" describe the local administrative interface. It states that the local administration is via the serial port console.

Section 2.2.7 "Create an Administrative User with tmsh access" of [ECG] provides the following warning:

> *NOTE: It is strongly recommended that, in addition to the administrative-user created above, you configure the primary administrative user (generally “admin”) with tmsh access as well, as this user is the only administrative-user able to login locally if otherwise locked out.*

Section 2.3.3.1 "Administrative users" of [ECG] provides a similar warning:

> *It is strongly recommended that the primary administrative user (generally “admin”) have tmsh access, as this user is the only administrative-user able to login locally if otherwise locked out.*

Assurance Activity AA-FMT_SMF.1-AGD-02

For distributed TOEs it is required to verify the Guidance Documentation to describe management of each TOE component. The evaluator shall confirm that all relevant aspects of each TOE component are covered by the FMT SFRs.

Summary

According to [ST] , the TOE is not a distributed TOE, therefore this assurance activity does not apply.

Test Assurance Activities

Assurance Activity AA-FMT_SMF.1-ATE-01

The evaluator shall examine the TSS, Guidance Documentation and the TOE as observed during all other testing and shall confirm that the management functions specified in FMT_SMF.1 are provided by the TOE. The evaluator shall confirm that the TSS details which security management functions are available through which interface(s) (local administration interface, remote administration interface).

Summary

The evaluator observed during testing that:

- A privileged user can administer the TOE locally and remotely.
- A privileged user can configure the access banner.
- A privileged user can configure the inactivity time before session termination.
- A privileged user can update the TOE and verify updates using digital signatures.
- A privileged user can configure authentication failure parameters for FIA_AFL.1.
- A privileged user can configure the time used for timestamps.
- A privileged user can start and stop services.
- A privileged user can configure audit behaviour.
- A privileged user can manage cryptographic keys.
- A privileged user can configure cryptographic functionality.
- A privileged user can configure threshold for SSH re-keying.

Assurance Activity AA-FMT_SMF.1-ATE-02

The evaluator tests management functions as part of testing the SFRs included in the ST. No separate testing for FMT_SMF.1 is required unless one of the management functions in FMT_SMF.1.1 has not already been exercised under any other SFR.

Summary

The evaluator performed the tests of the management functions as part of other SFRs, FTA_SSL_EXT.1, FTA_SSL.3, FTA_TAB.1, FMT_MOF.1/ManualUpdate, FIA_AFL.1, FIA_X509_EXT.2, FPT_TUD_EXT.1, FMT_MOF.1/Services, FMT_MTD.1/CryptoKeys, FPT_TST_EXT.1

Assurance Activity AA-FMT_SMF.1-ATE-03

Tests defined to verify the correct implementation of security management functions shall be performed for every TOE component. For security management functions that are implemented centrally, sampling should be applied when defining the evaluator's tests (ensuring that all components are covered by the sample).

Summary

The evaluator found that the TOE is not a distributed TOE, therefore there are no specific requirements for this assurance activity.

2.1.4.6 Restrictions on security roles (FMT_SMR.2)

TSS Assurance Activities

Assurance Activity AA-FMT_SMR.2-ASE-01

The evaluator shall examine the TSS to determine that it details the TOE supported roles and any restrictions of the roles involving administration of the TOE.

Summary

Chapter 7 of [ST] contains the TSS. Section 7.4.1 *Security Roles* describes roles supported by the TOE. It states the following:

- Access of individual users to the web-based Configuration utility, tmsh, iControl API, and iControl REST API is restricted based on pre-defined roles. These roles define which type of objects a user has access to and which type of tasks he or she can perform. The role definitions cannot be changed by TOE administrators.
- The TOE allows security administrators to define the type of terminal access that individual users have, i.e., whether they have access to the tmsh via SSH or not. The TOE can be administered either locally via a serial console or remotely via a trusted path connection.

Guidance Assurance Activities

Assurance Activity AA-FMT_SMR.2-AGD-01

The evaluator shall review the guidance documentation to ensure that it contains instructions for administering the TOE both locally and remotely, including any configuration that needs to be performed on the client for remote administration.

Summary

The evaluator examined section 2.1.1.2 "Establishing Administrative Access" of [ECG] which indicates that the TOE can be managed using the following interfaces across either a local (direct Ethernet) or remote (over the management network) connection to the TOE:

- Traffic management shell (tmsh) over SSH
- Web GUI over HTTPS
- iControl SOAP or iControl REST (both programmatic interfaces) over TLS

The evaluator further examined section 2.3.4 "Login to the BIG-IP" of [ECG] which provides instructions for administering the TOE using tmsh and Web GUI. As described, the TOE has an "admin" user with an Administrative role, by default. Other than configuration of administrative user, all TOE administration can be performed remotely (via tmsh or Web GUI).

Instructions to configure the management port/network is provided in the user manual BIG-IP System Essentials [ESSEN] which the evaluator examined and considered to be sufficiently detailed.

Instructions to configure SSH is provided in section 2.3.4.1 "SSH" and the GUI in section 2.3.4.2 "GUI".

Test Assurance Activities

Assurance Activity AA-FMT_SMR.2-ATE-01

In the course of performing the testing activities for the evaluation, the evaluator shall use all supported interfaces, although it is not necessary to repeat each test involving an administrative action with each interface. The evaluator shall ensure, however, that each supported method of administering the TOE that conforms to the requirements of this cPP be tested; for instance, if the TOE can be administered through a local hardware interface; SSH; and TLS/HTTPS; then all three methods of administration must be exercised during the evaluation team's test activities.

Summary

The evaluator performed the test as part of other SFRs, FTA_SSL.3, for GUI and SSH, FIA_UIA_EXT.1 for IControl and IControl Rest, FTA_SSL_EXT.1 for Serial Console

2.1.5 Protection of the TSF (FPT)

2.1.5.1 Protection of Administrator Passwords (FPT_APW_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FPT_APW_EXT.1-ASE-01

The evaluator shall examine the TSS to determine that it details all authentication data that are subject to this requirement, and the method used to obscure the plaintext password data when stored. The TSS shall also detail passwords are stored in such a way that they are unable to be viewed through an interface designed specifically for that purpose, as outlined in the application note.

Summary

Chapter 7 describes the TSS. Section 7.5.1 *Protection of Sensitive Data* describes how sensitive data is protected. It states that the TOE protect passwords used for the authentication of administrative users as follows:

In storage for local user authentication, the TOE's administrative interfaces do not offer any interface to retrieve user passwords or configuration files.

Additionally, the TOE does not offer an interface to retrieve the contents of its configuration files. Passwords are stored in a salted SHA-512 hashed format.

The evaluator verified that the TSS contains the required information.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

No assurance activities defined.

2.1.5.2 Protection of TSF Data (for reading of all symmmetric keys) (FPT_SKP_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FPT_SKP_EXT.1-ASE-01

> *The evaluator shall examine the TSS to determine that it details how any pre-shared keys, symmetric keys, and private keys are stored and that they are unable to be viewed through an interface designed specifically for that purpose, as outlined in the application note. If these values are not stored in plaintext, the TSS shall describe how they are protected/obscured.*

Summary

Chapter 7 of [ST] contains the TSS. The evaluator examined the TSS which contains the following information.

Section 7.2.2 *Zeroization of Critical Security Parameters* describes how critical security parameters such as secret and private keys are zeroized. It states that only TLS and SSH session keys are stored in plaintext form. The rest of the keys are stored in encrypted format.

Encrypted keys are stored using the F5 Secure Vault as described in section 7.2.2.

Section 7.5.1 *Protection of Sensitive Data* describes how sensitive data is protected stating that pre-shared keys (such as credentials for remote servers), symmetric keys, and private keys are stored in the TOE's configuration files. The TOE does not offer an interface to retrieve the contents of its configuration files. Passwords are stored in a salted SHA-512 hashed format.

The evaluator verified that the TSS contains the required information.

Guidance Assurance Activities

No assurance activities defined.

Test Assurance Activities

No assurance activities defined.

2.1.5.3 Reliable Time Stamps (FPT_STM_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FPT_STM_EXT.1-ASE-01

The evaluator shall examine the TSS to ensure that it lists each security function that makes use of time, and that it provides a description of how the time is maintained and considered reliable in the context of each of the time related functions.

[TD0632] If "obtain time from the underlying virtualization system" is selected, the evaluator shall examine the TSS to ensure that it identifies the VS interface the TOE uses to obtain time. If there is a delay between updates to the time on the VS and updating the time on the TOE, the TSS shall identify the maximum possible delay.

Summary

Chapter 7 of [ST] contains the TSS. Section 7.5.4 *Time Source* describes time source. It states that the TOE provides reliable time stamps for its own use. For F5 devices and vCMP, the TOE hardware includes a hardware-based clock and the TOE's operating system makes the real-time clock available through a mcpd-maintained time stamp. Administrators have the ability to set the hardware-based clock on F5 devices and vCMP and to set the BIG-IP system clock on hypervisors.

The security functions that rely on the time stamp include:

- generation of audit record
- session locking for administrative users
- timeouts or remote sessions
- certificate validation / revocation

The evaluator verified that the TSS contains the necessary information.

Guidance Assurance Activities

Assurance Activity AA-FPT_STM_EXT.1-AGD-01

The evaluator examines the guidance documentation to ensure it instructs the administrator how to set the time. If the TOE supports the use of an NTP server, the guidance documentation instructs how a communication path is established between the TOE and the NTP server, and any configuration of the NTP client on the TOE to support this communication.

[TD0632] If the TOE supports obtaining time from the underlying VS, the evaluator shall verify the Guidance Documentation specifies any configuration steps necessary. If no configuration is necessary, no statement is necessary in the Guidance Documentation. If there is a delay between updates to the time on the VS and updating the time on the TOE, the evaluator shall ensure the Guidance Documentation informs the administrator of the maximum possible delay.

Summary

According to [ST] , the TOE does not support the use of an NTP server for time services.

Test Assurance Activities

Assurance Activity AA-FPT_STM_EXT.1-ATE-01

The evaluator shall perform the following tests:

a) Test 1: If the TOE supports direct setting of the time by the Security Administrator then the evaluator uses the guidance documentation to set the time. The evaluator shall then use an available interface to observe that the time was set correctly.

b) Test 2: If the TOE supports the use of an NTP server; the evaluator shall use the guidance documentation to configure the NTP client on the TOE and set up a communication path with the NTP server. The evaluator will observe that the NTP server has set the time to what is expected. If the TOE supports multiple protocols for establishing a connection with the NTP server, the evaluator shall perform this test using each supported protocol claimed in the guidance documentation.

c) [TD0632] Test 3: [conditional] If the TOE obtains time from the underlying VS, the evaluator shall record the time on the TOE, modify the time on the underlying VS, and verify the modified time is reflected by the TOE. If there is a delay between the setting the time on the VS and when the time is reflected on the TOE, the evaluator shall ensure this delay is consistent with the TSS and Guidance.

If the audit component of the TOE consists of several parts with independent time information, then the evaluator shall verify that the time information between the different parts are either synchronized or that it is possible for all audit information to relate the time information of the different part to one base information unambiguously.

Summary

a) Test 1:
The evaluator used a computer running Ubuntu Linux as a remote SSH client. The evaluator logged in successfully to the TOE as an administrator user through SSH. The evaluator configured the TOE clock according to see [ECG] "Time Changes". The evaluator was able set up the time successfully.

b) Test 2:
The evaluator found that the TOE according to see [ST] 7.5.4 "Time Source" does not support the use of NTP server, therefore there are no specific requirements for this assurance activity.

The evaluator found that the audit component of the TOE does not consist of several parts with independent time information, therefore there are no specific requirements for this assurance activity.

2.1.5.4 Extended: TSF Testing (FPT_TST_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FPT_TST_EXT.1-ASE-01

The evaluator shall examine the TSS to ensure that it details the self-tests that are run by the TSF; this description should include an outline of what the tests are actually doing (e.g., rather than saying "memory is tested", a description similar to "memory is tested by writing a value to each memory location and reading it back to ensure it is identical to what was written" shall be used). The evaluator shall ensure that the TSS makes an argument that the tests are sufficient to demonstrate that the TSF is operating correctly.

Summary

Chapter 7 contains the TSS. Section 7.5.2 *Self-tests* describes self-tests. The evaluator examined the TSS and it states that the TOE provides the following self-tests:

- On F5 devices and vCMP. the BIOS Power-On Self-Test POST test is only run at power on
- The OpenSSL integrity tests are run at power on and reboot (during OpenSSL initialization) for OpenSSL.
- The software integrity check (i.e., sys-icheck utility) is run at power on and reboot to check the integrity of the RPMs. This self-test can be run at any time.
- The cryptographic algorithm self-tests provided by OpenSSL are run at power on and reboot (during OpenSSL initialization).

This section also outlines what the tests are doing as follows:

The BIOS POST is a diagnostic program that checks the basic components required for the hardware to operate, tests the memory, and checks the disk controller, the attached disks, and the network controllers. The BIOS POST tests cannot be run on demand.

The fipscheck utility is a standard Open Source utility for verifying the integrity of OpenSSL during initialization.

The sys-icheck utility provides software integrity testing by comparing the current state of files in the system to a database created at install time and modified only through authorized system update mechanisms. When a discrepancy is detected, the utility reports that discrepancy. The utility can be run at any time during system operation, and will just report errors. However, once the system is placed into the Common Criteria configuration it is enabled to run at each boot, and will halt the boot if errors are found. The TOE will execute self-tests at power-on to test the cryptographic algorithms and random number generation using known answer tests for each of the algorithms. If a power-on test fails, the boot process will halt.

The TSS provides the following argument indicating that the tests are sufficient to demonstrate that the TSF is operating correctly:

> *The self-tests implemented by the TOE which are described in this section cover all aspects of the TSF are therefore and are sufficient for demonstrating that the TSF is operating correctly in the intended environment.*

The evaluator considered the argument reasonable as the self-tests cover many aspects starting at power on with BIOS POST test as well as integrity tests and known-answer cryptographic algorithm tests for the OpenSSL cryptographic module to integrity tests to verify the TOE software.

Assurance Activity AA-FPT_TST_EXT.1-ASE-02

For distributed TOEs the evaluator shall examine the TSS to ensure that it details which TOE component performs which self-tests and when these self-tests are run.

Summary

According to the Security Target [ST] , the TOE is not distributed therefore this Evaluation Activity is not applicable.

Guidance Assurance Activities

Assurance Activity AA-FPT_TST_EXT.1-AGD-01

The evaluator shall also ensure that the guidance documentation describes the possible errors that may result from such tests, and actions the administrator should take in response; these possible errors shall correspond to those described in the TSS.

Summary

The evaluator examined section 2.1 "Preparing for BIG-IP Installation and Configuration" of [ECG] and determined that the guidance describes the possible error that may result from such tests and actions that administrator should take in response:

- The sys-icheck utility provides software integrity testing by comparing the current state of files in the system with a database created at install time and reports all discrepancies. This is run automatically by the ccmode utility and at each system boot, and can be run from the tmsh shell on demand. When run from ccmode or on demand, the utility reports the errors to the session issuing the utility command; the administrator should confirm that any modifications are expected and if they are not, reinstall the system. When sys-icheck is run during boot, the boot will halt if an error is found, and the administrator should reinstall.

- OpenSSL, cryptographic algorithm, and random number generation tests are run at boot time. They will halt the boot if failure occurs, and the administrator should reinstall.

The evaluator determined that the possible self-test errors described in [ECG] correspond to those described in the TSS in [ST] .

Assurance Activity AA-FPT_TST_EXT.1-AGD-02

For distributed TOEs the evaluator shall ensure that the guidance documentation describes how to determine from an error message returned which TOE component has failed the self-test.

Summary

According to [ST] , the TOE is not a distributed TOE, therefore this assurance activity does not apply.

Test Assurance Activities

Assurance Activity AA-FPT_TST_EXT.1-ATE-01

It is expected that at least the following tests are performed:

a) Verification of the integrity of the firmware and executable software of the TOE

b) Verification of the correct operation of the cryptographic functions necessary to fulfil any of the SFRs.

Although formal compliance is not mandated, the self-tests performed should aim for a level of confidence comparable to:

a) [FIPS 140-2], chap. 4.9.1, Software/firmware integrity test for the verification of the integrity of the firmware and executable software. Note that the testing is not restricted to the cryptographic functions of the TOE.

b) [FIPS 140-2], chap. 4.9.1, Cryptographic algorithm test for the verification of the correct operation of cryptographic functions. Alternatively, national requirements of any CCRA member state for the security evaluation of cryptographic functions should be considered as appropriate.

The evaluator shall either verify that the self-tests described above are carried out during initial start-up or that the developer has justified any deviation from this.

Summary

[ST] in chapter 7.5.2 "Self-tests" states that the TOE will execute self-tests at power-on to test the cryptographic functions and integrity of the firmware and executable software. If a power-on test fails, the boot process will halt. The evaluator executed the self test on demand to verify that the test could run and that the output was displayed. The output showed no errors found.

Assurance Activity AA-FPT_TST_EXT.1-ATE-02

For distributed TOEs the evaluator shall perform testing of self-tests on all TOE components according to the description in the TSS about which self-test are performed by which component.

Summary

Since the TOE is not distributed, the evaluator determined this assurance activity to be not applicable.

2.1.5.5 Trusted Update (FPT_TUD_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FPT_TUD_EXT.1-ASE-01

The evaluator shall verify that the TSS describe how to query the currently active version. If a trusted update can be installed on the TOE with a delayed activation, the TSS needs to describe how and when the inactive version becomes active. The evaluator shall verify this description.

The evaluator shall verify that the TSS describes all TSF software update mechanisms for updating the system firmware and software (for simplicity the term "software" will be used in the following although the requirements apply to firmware and software). The evaluator shall verify that the description includes a digital signature verification of the software before installation and that installation fails if the verification fails. Alternatively, an approach using a published hash can be used. In this case the TSS shall detail this mechanism instead of the digital signature verification mechanism. The evaluator shall verify that the TSS describes the method by which the digital signature or published hash is verified to include how the candidate updates are obtained, the processing associated with verifying the digital signature or published hash of the update, and the actions that take place for both successful and unsuccessful signature verification or published hash verification.

If the options "support automatic checking for updates" or "support automatic updates" are chosen from the selection in FPT_TUD_EXT.1.2, the evaluator shall verify that the TSS explains what actions are involved in automatic checking or automatic updating by the TOE, respectively.

Summary

Chapter 7 of [ST] contains the TSS. Section 7.5.3 *Update Verification* describes the TOE software update process. It states that the TOE provides functionality that supports administrators in verifying the integrity and authenticity of patches provided by the developer, F5. The TOE version can be queried using the administrative interface Configuration Utility or tsmh.

On hardware-based (F5 devices) and vCMP, the TOE validates digital signatures of update provided by F5. F5 places the ISO files (updates) and signature files on their website. The administrative guidance instructs the customer to download the the ISO and digital signature file and verify the ISO against the digital signature file prior to installing the update.

On hypervisors, the digital signatures of updates provided by F5 must be verified by the Security Administrator prior to being installed. The digital signatures of the image files and the product ISO update and image files are located on the F5 customer download website. The administrative guidance instructs the customer to download and verify the image files or product ISO and digital signatures prior to installing.

The TSS states that a signature file exists for each ISO, update, or image file provided by F5. The content of the signature file is a digital signature of a SHA256 digest of the ISO image file. The private and public keys are generated using the OpenSSL utility. The signing key is a 2048 bit RSA private key that is stored at F5 CM and only available for official F5 builds. The public key is included in the TMOS filesystem and is available on the F5 official site adjacent to the software archives. Note: The update verification implementation does not utilize certificates; only digital signatures.

The BIG-IP verifies the SHA256 hash of software archives, using 2048-bit RSA digital signature algorithm. If the signature is verified, the software update is installed. If the signature does not verify, the software update installation fails / aborts. The administrative guidance will instruct the customer to download the update again or contact F5 support.

The evaluator notes that the ST does not indicate that the TOE can be installed with a delayed activation, thus, no TSS description was deemed necessary.

Assurance Activity AA-FPT_TUD_EXT.1-ASE-02

For distributed TOEs, the evaluator shall examine the TSS to ensure that it describes how all TOE components are updated, that it describes all mechanisms that support continuous proper functioning of the TOE during update (when applying updates separately to individual TOE components) and how verification of the signature or checksum is performed for each TOE component. Alternatively, this description can be provided in the guidance documentation. In that case the evaluator should examine the guidance documentation instead.

Summary

According to the Security Target [ST] , the TOE is not distributed therefore this Evaluation Activity is not applicable.

Assurance Activity AA-FPT_TUD_EXT.1-ASE-03

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

If a published hash is used to protect the trusted update mechanism, then the evaluator shall verify that the trusted update mechanism does involve an active authorization step of the Security Administrator, and that download of the published hash value, hash comparison and update is not a fully automated process involving no active authorization by the Security Administrator. In particular, authentication as Security Administration according to FMT_MOF.1/ManualUpdate needs to be part of the update process when using published hashes.

Summary

Per [ST] the TOE's trusted update mechanism validates digital signatures of updates (available as ISO image or image files). Each update is provided with an associated digital signature file that contains a digital signature of a SHA-256 digest of the ISO image or image files. This is consistent with FPT_TUD_EXT.1.3 which selects digital signature mechanism as the only method supported by the TOE.

Guidance Assurance Activities

Assurance Activity AA-FPT_TUD_EXT.1-AGD-01

The evaluator shall verify that the guidance documentation describes how to query the currently active version. If a trusted update can be installed on the TOE with a delayed activation, the guidance documentation needs to describe how to query the loaded but inactive version.

The evaluator shall verify that the guidance documentation describes how the verification of the authenticity of the update is performed (digital signature verification or verification of published hash). The description shall include the procedures for successful and unsuccessful verification. The description shall correspond to the description in the TSS.

If a published hash is used to protect the trusted update mechanism, the evaluator shall verify that the guidance documentation describes how the Security Administrator can obtain authentic published hash values for the updates.

Summary

The evaluator examined section 2.2.2.3 "Updating BIG-IP software after initial configuration" of [ECG] which states that for hardware and vCMP, the process of updating BIG-IP is the same as the initial install, except the administrator does not need to verify the image. Since the ccmode command has already been run during the initial install, the BIG-IP will automatically verify the new ISO using the digital signature as part of the upload and installation process initiated by the administrative-user. Section 2.2.2.3 "Updating BIG-IP software after initial configuration" also refers to [SWUPDATE] for instructions to update the BIG-IP VE software after initial installation. For the update, it's available as an ISO image. Section 2.2.2.3 refers to 2.2.2.2 for verifying the integrity of the ISO image which is the same instructions as for verifying the integrity of the product file (image fileset). In both formats, the TOE software download is digitally signed where the administrator can manually verify by following the instructions provided in section 2.2.2.2 "Verifying the product ISO using the digital signature". This section also provides the following instructions:

If the signature verification on the BIG-IP fails, the software update installation will fail. In this case, try to download the image fileset again. If the signature verification fails a second time, contact F5 Support.

Successful verification can be demonstrated by checking the installed software via the command "tmsh show sys software status" as described in section 2.2.1.3 "Verifying the Installed / Running Versions of the Software".

The TSS (section 7.5.3 "Update Verification" of [ST]) described that the TOE allows updates of the TOE software, verifying its trust and integrity using digital signature before being installed. Therefore, the evaluator determined that the guidance description corresponds to the description in the TSS.

Also, the evaluator did not find any statement in [ST] , particularly in the TSS specifying that a trusted update can be installed on the TOE with a delayed activity, thus, the evaluator determined that no guidance was needed to describe how to query the loaded but inactive version.

Moreover, [ST] section 6.2.5.6 provides the definition FPT_TUD_EXT.1 which states that the TOE provides means to authenticate firmware/software updates to the TOE using a digital signature mechanism prior to installing those updates, i.e., the TOE does not support the hash mechanism for security updates.

Based on the provided guidance, the evaluator noted that the instructions for performing an update to the TOE is manual which requires the involvement of an administrator.

Assurance Activity AA-FPT_TUD_EXT.1-AGD-02

For distributed TOEs the evaluator shall verify that the guidance documentation describes how the versions of individual TOE components are determined for FPT_TUD_EXT.1, how all TOE components are updated, and the error conditions that may arise from checking or applying the update (e.g. failure of signature verification, or exceeding available storage space) along with appropriate recovery actions. . The guidance documentation only has to describe the procedures relevant for the user; it does not need to give information about the internal communication that takes place when applying updates.

Summary

According to [ST] , the TOE is not a distributed TOE, therefore this assurance activity does not apply.

Assurance Activity AA-FPT_TUD_EXT.1-AGD-03

If this was information was not provided in the TSS: For distributed TOEs, the evaluator shall examine the Guidance Documentation to ensure that it describes how all TOE components are updated, that it describes all mechanisms that support continuous proper functioning of the TOE during update (when applying updates separately to individual TOE components) and how verification of the signature or checksum is performed for each TOE component.

Summary

According to [ST] , the TOE is not a distributed TOE, therefore this assurance activity does not apply.

Assurance Activity AA-FPT_TUD_EXT.1-AGD-04

If this was information was not provided in the TSS: If the ST author indicates that a certificate-based mechanism is used for software update digital signature verification, the evaluator shall verify that the Guidance Documentation contains a description of how the certificates are contained on the device. The evaluator also ensures that the Guidance Documentation describes how the certificates are installed/updated/selected, if necessary.

Summary

According to the TSS, the update verification implementation does not utilize certificates. In other words, only digital signatures are used. Thus, this assurance activity is not applicable.

Test Assurance Activities

Assurance Activity AA-FPT_TUD_EXT.1-ATE-01

The evaluator shall perform the following tests:

a) Test 1: The evaluator performs the version verification activity to determine the current version of the product. If a trusted update can be installed on the TOE with a delayed activation, the evaluator shall also query the most recently installed version (for this test the TOE shall be in a state where these two versions match). The evaluator obtains a legitimate update using procedures described in the guidance documentation and verifies that it is successfully installed on the TOE. For some TOEs loading the update onto the TOE and activation of the update are separate steps ("activation" could be performed e.g. by a distinct activation step or by rebooting the device). In that case the evaluator verifies after loading the update onto the TOE but before activation of the update that the current version of the product did not change but the most recently installed version has changed to the new product version. After the update, the evaluator performs the version verification activity again to verify the version correctly corresponds to that of the update and that current version of the product and most recently installed version match again.

b) Test 2 [conditional]: If the TOE itself verifies a digital signature to authorize the installation of an image to update the TOE the following test shall be performed (otherwise the test shall be omitted). The evaluator first confirms that no updates are pending and then performs the version verification activity to determine the current version of the product, verifying that it is different from the version claimed in the update(s) to be used in this test. The evaluator obtains or produces illegitimate updates as defined below and attempts to install them on the TOE. The evaluator verifies that the TOE rejects all of the illegitimate updates. The evaluator performs this test using all of the following forms of illegitimate updates:

1) A modified version (e.g. using a hex editor) of a legitimately signed update

2) An image that has not been signed

3) An image signed with an invalid signature (e.g. by using a different key as expected for creating the signature or by manual modification of a legitimate signature)

4) If the TOE allows a delayed activation of updates the TOE must be able to display both the currently executing version and most recently installed version. The handling of version information of the most recently installed version might differ between different TOEs depending on the point in time when an attempted update is rejected. The evaluator shall verify that the TOE handles the most recently installed version information for that case as described in the guidance documentation. After the TOE has rejected the update the evaluator shall verify, that both, current version and most recently installed version, reflect the same version information as prior to the update attempt.

c) Test 3 [conditional]: If the TOE itself verifies a hash value over an image against a published hash value (i.e. reference value) that has been imported to the TOE from outside such that the TOE itself authorizes the installation of an image to update the TOE, the following test shall be performed (otherwise the test shall be omitted. If the published hash is provided to the TOE by the Security Administrator and the verification of the hash value over the update file(s) against the published hash is performed by the TOE, then the evaluator shall perform the following tests. The evaluator first confirms that no update is pending and then performs the version verification activity to determine the current version of the product, verifying that it is different from the version claimed in the update(s) to be used in this test.

1) The evaluator obtains or produces an illegitimate update such that the hash of the update does not match the published hash. The evaluator provides the published hash value to the TOE and calculates the hash of the update either on the TOE itself (if that functionality is provided by the TOE), or else outside the TOE. The evaluator confirms that the hash values are different, and attempts to install the update on the TOE, verifying that this fails because of the difference in hash values (and that the failure is logged). Depending on the implementation of the TOE, the TOE might not allow the user to even attempt updating the TOE after the verification of the hash value fails. In that case the verification that the hash comparison fails is regarded as sufficient verification of the correct behaviour of the TOE

2) The evaluator uses a legitimate update and tries to perform verification of the hash value without providing the published hash value to the TOE. The evaluator confirms that this attempt fails. Depending on the implementation of the TOE it might not be possible to attempt the verification of the hash value without providing a hash value to the TOE, e.g. if the hash value needs to be handed over to the TOE as a parameter in a command line message and the syntax check of the command prevents the execution of the command without providing a hash value. In that case the mechanism that prevents the execution of this check shall be tested accordingly, e.g. that the syntax check rejects the command without providing a hash value, and the rejection of the attempt is regarded as sufficient verification of the correct behaviour of the TOE in failing to verify the hash. The evaluator then attempts to install the update on the TOE (in spite of the unsuccessful hash verification) and confirms that this fails. Depending on the implementation of the TOE,

the TOE might not allow to even attempt updating the TOE after the verification of the hash value fails. In that case the verification that the hash comparison fails is regarded as sufficient verification of the correct behaviour of the TOE

3) *If the TOE allows delayed activation of updates, the TOE must be able to display both the currently executing version and most recently installed version. The handling of version information of the most recently installed version might differ between different TOEs. Depending on the point in time when the attempted update is rejected, the most recently installed version might or might not be updated. The evaluator shall verify that the TOE handles the most recently installed version information for that case as described in the guidance documentation. After the TOE has rejected the update the evaluator shall verify, that both, current version and most recently installed version, reflect the same version information as prior to the update attempt.*

If the verification of the hash value over the update file(s) against the published hash is not performed by the TOE, Test 3 shall be skipped.

The evaluator shall perform Test 1, Test 2 and Test 3 (if applicable) for all methods supported (manual updates, automatic checking for updates, automatic updates).

Summary

Test 1:
The evaluator used a computer running Ubuntu Linux. The evaluator downloaded a software image of the TOE software update together with the signature and the public key. The validity of the image was verified by the evaluator, see [ECG] 2 "Installation and configuration procedures". The evaluator via remote sftp and GUI uploaded to the TOE the software update together with the signature and the public key. The evaluator performed a verification of the current version of the product. The evaluator installed and enabled the latest version of the TOE software. After enabling of the latest version, the evaluator compared the current version with the previous that was installed. The evaluator identified that the software version changed to the updated one.

Test 2:
The evaluator via GUI uploaded to the TOE a modified software update together with the signature and the public key. When the installation operation failed, the evaluator compared the latest version installed on the TOE and it was the latest legitimate version that was previously installed i.e, the evaluator identified that the version did not change.
The evaluator via GUI tried to upload to the TOE a software update which was not being signed but the upload failed (as expected).
The evaluator via GUI uploaded to the TOE a software update without a valid signature. The evaluator tried to install and enable the latest version of the TOE software but failed (as expected).

Test 3:
[ST] 7.5.3 "Update Verification" states that the TOE uses only digital signature mechanism in order to authenticate firmware/software. The evaluator therefore considers this requirement not applicable.

Assurance Activity AA-FPT_TUD_EXT.1-ATE-02

For distributed TOEs the evaluator shall perform Test 1, Test 2 and Test 3 (if applicable) for all TOE components.

Summary

The evaluator found that the TOE is not a distributed TOE, therefore there are no specific requirements for this assurance activity.

2.1.6 TOE access (FTA)

2.1.6.1 TSF-initiated Termination (FTA_SSL.3)

TSS Assurance Activities

Assurance Activity AA-FTA_SSL.3-ASE-01

The evaluator shall examine the TSS to determine that it details the administrative remote session termination and the related inactivity time period.

Summary

Chapter 7 of [ST] contains the TSS. Section 7.6 *TOE Access* describes TOE access. It states that the following:

- The TOE terminates local and remote administrative user sessions (Console, Configuration Utility or tmsh) sessions after an administrator-defined period of inactivity.
- Users can also actively terminate their sessions (log out).
- Administrators are able to actively terminate these sessions (i.e., to log out and therefore close an authenticated session).

Guidance Assurance Activities

Assurance Activity AA-FTA_SSL.3-AGD-01

The evaluator shall confirm that the guidance documentation includes instructions for configuring the inactivity time period for remote administrative session termination.

Summary

Section 3.2 "Maximum Failed Login Attempts" of [ECG] provides relevant guidance for administrative session locking. It states the following:

- The administrator can set a parameter that specifies the maximum number of consecutive failed login attempts that can occur before a given user account will be locked out. The default setting is 3. It is highly recommended that the default setting be retained (i.e., not changed).
- If a user becomes locked out, the user account will be unlocked after an administrator-specified duration. The ccmode script sets the default to 600 seconds (10 minutes). This can also be configured manually via the following tmsh command: `tmsh modify /sys db password.unlock_time value <value in seconds>`
- Alternatively, the ccmode script also configures the evaluated configuration to disable the manual unlock (in favor of the timed unlock), and to allow the primary administrative user (generally “admin”) to log on from the local serial console even if the account is locked. This ensures that at least one user account is available at all times. If the primary administrative user does log in locally, its lockout counter will be reset and it will be able to log in remotely as well.

Additional details are provided in [K9908] as pointed out in section 2.3.12 "Session Inactivity Termination" of [ECG] .

Test Assurance Activities

Assurance Activity AA-FTA_SSL.3-ATE-01

For each method of remote administration, the evaluator shall perform the following test:

a) Test 1: The evaluator follows the guidance documentation to configure several different values for the inactivity time period referenced in the component. For each period configured, the evaluator establishes a remote interactive session with the TOE. The evaluator then observes that the session is terminated after the configured time period.

Summary

a) Test 1 :
The evaluator used a computer running Ubuntu Linux as a remote SSH client. The evaluator logged in successfully to the TOE as an administrator user through SSH. The evaluator configured the TOE SSH inactivity timeout.

The evaluator used a computer running Ubuntu Linux as a remote HTTPS client. The evaluator logged in successfully to the TOE as an administrator user through GUI. The evaluator configured the TOE GUI session inactivity timeout.

The evaluator used a computer running Ubuntu Linux as a remote HTTPS client. The evaluator logged in successfully to the TOE as an administrator user through IControl Rest. The evaluator identified that in order to successfully authenticate a user needs to provide valid credentials every time. Termination of the connection is executed right after the administrative action. As a result no session termination is monitored because of time expiration but only for action termination. The evaluator therefore considers this requirement not applicable.

The evaluator used a computer running Ubuntu Linux as a remote HTTPS client. The evaluator logged in successfully to the TOE as an administrator user through IControl. The evaluator identified that in order to successfully authenticate a user needs to provide valid credentials every time. Termination of the connection is executed right after the administrative action. As a result no session termination is monitored because of time expiration but only for action termination. The evaluator therefore considers this requirement not applicable.

2.1.6.2 User-initiated Termination (FTA_SSL.4)

TSS Assurance Activities

Assurance Activity AA-FTA_SSL.4-ASE-01

The evaluator shall examine the TSS to determine that it details how the local and remote administrative sessions are terminated.

Summary

Chapter 7 of [ST] contains the TSS. Section 7.6 *TOE Access* describes TOE access. It states that the following:

- The TOE terminates local and remote administrative user sessions (Console, Configuration Utility or tmsh) sessions after an administrator-defined period of inactivity.
- Users can also actively terminate their sessions (log out).

- Administrators are able to actively terminate these sessions (i.e., to log out and therefore close an authenticated session).

Guidance Assurance Activities

Assurance Activity AA-FTA_SSL.4-AGD-01

The evaluator shall confirm that the guidance documentation states how to terminate a local or remote interactive session.

Summary

Guidance for configuring termination of a local or remote interactive lesson is provided in [K9908] "Configuring an automatic logout for idle sessions" which provides instructions to configure automatic logout for the GUI/Configuration utility sessions, tmsh sessions, and console sessions. This task can be done via the Configuration Utility for Configuration utility sessions, tmsh for tmsh sessions and console sessions.

Test Assurance Activities

Assurance Activity AA-FTA_SSL.4-ATE-01

For each method of remote administration, the evaluator shall perform the following tests:

a) Test 1: The evaluator initiates an interactive local session with the TOE. The evaluator then follows the guidance documentation to exit or log off the session and observes that the session has been terminated.

b) Test 2: The evaluator initiates an interactive remote session with the TOE. The evaluator then follows the guidance documentation to exit or log off the session and observes that the session has been terminated.

Summary

The evaluator performed the test as part of other SFR, FIA_UIA_EXT.1

2.1.6.3 TSF-initiated Session Locking (FTA_SSL_EXT.1)

TSS Assurance Activities

Assurance Activity AA-FTA_SSL_EXT.1-ASE-01

The evaluator shall examine the TSS to determine that it details whether local administrative session locking or termination is supported and the related inactivity time period settings.

Summary

Chapter 7 of [ST] contains the TSS. Section 7.6 *TOE Access* describes TOE access. It states that the following:

- The TOE terminates local and remote administrative user sessions (Console, Configuration Utility or tmsh) sessions after an administrator-defined period of inactivity. Users can also actively terminate their sessions (log out).
- Administrators are able to actively terminate these sessions (i.e., to log out and therefore close an authenticated session).

Guidance Assurance Activities

Assurance Activity AA-FTA_SSL_EXT.1-AGD-01

The evaluator shall confirm that the guidance documentation states whether local administrative session locking or termination is supported and instructions for configuring the inactivity time period.

Summary

Section 3.2 "Maximum Failed Login Attempts" of [ECG] provides relevant guidance for administrative session locking. It states the following:

- If a user becomes locked out, the user account will be unlocked after an administrator-specified duration. The ccmode script sets the default to 600 seconds (10 minutes). This can also be configured manually via the following tmsh command: `tmsh modify /sys db password_time.unlock_time value <value in seconds>`
- Alternatively, the ccmode script also configures the evaluated configuration to disable the manual unlock (in favor of the timed unlock), and to allow the primary administrative user (generally “admin”) to log on from the local serial console even if the account is locked. This ensures that at least one user account is available at all times. If the primary administrative user does log in locally, its lockout counter will be reset and it will be able to log in remotely as well. Additional details are provided in [K9908] as pointed out in section 2.3.12 "Session Inactivity Termination" of [ECG] .

Test Assurance Activities

Assurance Activity AA-FTA_SSL_EXT.1-ATE-01

The evaluator shall perform the following test:

a) Test 1: The evaluator follows the guidance documentation to configure several different values for the inactivity time period referenced in the component. For each period configured, the evaluator establishes a local interactive session with the TOE. The evaluator then observes that the session is either locked or terminated after the configured time period. If locking was selected from the component, the evaluator then ensures that re-authentication is needed when trying to unlock the session.

Summary

Test 1:
The evaluator used a computer running Ubuntu Linux as a ssh client. The evaluator logged in successfully as a local user through vSphere client to the Host TOE machine's serial console. The evaluator configured the TOE with different values for the serial console session timeout and every time the evaluator left the connection idle. The evaluator was able to monitor the termination of the session for each configured value.

2.1.6.4 Default TOE Access Banners (FTA_TAB.1)

TSS Assurance Activities

Assurance Activity AA-FTA_TAB.1-ASE-01

The evaluator shall check the TSS to ensure that it details each administrative method of access (local and remote) available to the Security Administrator (e.g., serial port, SSH, HTTPS). The evaluator shall check the TSS to ensure that all administrative methods of access available to the Security Administrator are listed and that the TSS states that the TOE is displaying an advisory notice and a consent warning message for each administrative method of access. The advisory notice and the consent warning message might be different for different administrative methods of access and might be configured during initial configuration (e.g. via configuration file).

Summary

Chapter 7 of [ST] contains the TSS. Section 7.4 *Security Function Management* identifies the user interfaces to the TOE:

- Configuration Utility (Web-based GUI) - browser-based GUI interface with normal GUI panels and selections. The client browser talks to the Apache HTTP server over HTTPS; then the request passes through tomcat and to the BIG-IP.
- tmsh shell commands – provide a command line interface, accessible through an SSH client
- iControl API – SOAP-based programming interface over HTTPS
- iControl REST API – REST-based programming interface over HTTPS.

The TOE supports local and remote login methods. Local login is described in section 7.3 *Identification and Authentication* and remote login is described in section 7.2 *Cryptographic Support* and section 7.7 *Trusted Path/Channels* of the TSS. Local login is via password-based authentication. Administrative users are identified by a user name and authenticated by an individual password associated with that user's account.

The TOE provides four interfaces for administrators to login remotely: Configuration Utility, iControl API, and iControl Rest API, and tmsh (via SSH). These interfaces communicate with the TOE via TLS (HTTPS). Connection to tmsh is via SSH version 2 or serial port console.

For local password-based authentication, section 7.3 states:

> *Administrative users (i.e., all users authorized to access the TOE's administrative interfaces) are identified by a user name and authenticated by an individual password associated with that user's account. This is true regardless of how the administrative user interfaces with the TOE. If the supplied user name and password match the user name and password pair maintained by the TOE, the administrative session is successfully established. Otherwise, the user receives an error and the session is not established. [...].*

The evaluator examined section 7.6 *TOE access* which states "For interactive user authentication at the web-based Configuration utility via HTTPS and the command line tmsh via SSH or the serial port console, BIG-IP implements the display of administrator-defined banners to users."

The evaluator determined that the TSS contains the necessary information.

Guidance Assurance Activities

Assurance Activity AA-FTA_TAB.1-AGD-01

The evaluator shall check the guidance documentation to ensure that it describes how to configure the banner message.

Summary

Section 2.3.5 "Login welcome banners" and 2.2.4 "Setting up the banner for the serial console" of [ECG] provide guidance on how to configure the banner message for the GUI, tmsh, and console interfaces. The banners for GUI and tmsh interfaces can be configured using via tmsh. Section 2.3.5 states that the warning message is enabled by default for the GUI but disabled by default for tmsh. Thus, instructions are provided in this section for enabling the banner for tmsh. Instructions for configuring the banner for the console is provided in section 2.2.4 which references [K6068] for further details.

Test Assurance Activities

Assurance Activity AA-FTA_TAB.1-ATE-01

The evaluator shall also perform the following test:

a) Test 1: The evaluator follows the guidance documentation to configure a notice and consent warning message. The evaluator shall then, for each method of access specified in the TSS, establish a session with the TOE. The evaluator shall verify that the notice and consent warning message is displayed in each instance.

Summary

The evaluator used a computer running Ubuntu Linux as a remote HTTPS client. The evaluator logged in successfully to the TOE as an administrator user through GUI. The evaluator configured the Security Banner. The evaluator verified the changes in the Security Banner by logging in to the TOE as an administrator user through GUI.

The evaluator used a computer running Ubuntu Linux as a remote SSH client. The evaluator logged in successfully to the TOE as an administrator user through SSH. The evaluator configured the Security Banner. The evaluator verified the changes in the Security Banner by logging in to the TOE as an administrator user through SSH.

The evaluator used a computer running Ubuntu Linux as a remote SSH client. The evaluator logged in successfully to the TOE as a non administrator user through GUI. The evaluator was not able to configure the Security Banner.

The evaluator used a computer running Ubuntu Linux as a ssh client. The evaluator logged in successfully to the Host TOE machine as a local user through vSphere client and then to the serial console. The evaluator configured the Security Banner. The evaluator verified the changes in the Security Banner by logging in to the TOE as administrator user through serial console.

2.1.7 Trusted path/channels (FTP)

2.1.7.1 Inter-TSF Trusted Channel (FTP_ITC.1)

TSS Assurance Activities

Assurance Activity AA-FTP_ITC.1-ASE-01

The evaluator shall examine the TSS to determine that, for all communications with authorized IT entities identified in the requirement, each secure communication mechanism is identified in terms of the allowed protocols for that IT entity, whether the TOE acts as a server or a client, and the method of assured identification of the non-TSF endpoint. The evaluator shall also confirm that all secure communication mechanisms are described in sufficient detail to allow the evaluator to match them to the cryptographic protocol Security Functional Requirements listed in the ST.

Summary

Chapter 7 of [ST] contains the TSS. Section 7.4 *Security Function Management* identifies the user interfaces to the TOE:

- Configuration Utility (Web-based GUI) - browser-based GUI interface with normal GUI panels and selections. The client browser talks to the Apache HTTP server over HTTPS; then the request passes through tomcat and to the BIG-IP.
- tmsh shell commands – provide a command line interface, accessible through an SSH client
- iControl API – SOAP-based programming interface over HTTPS
- iControl REST API – REST-based programming interface over HTTPS.

The TOE supports local and remote login methods. Local login is described in section 7.3 *Identification and Authentication* and remote login is described in section 7.2 *Cryptographic Support* and section 7.7 *Trusted Path/Channels* of the TSS. Local login is via password-based authentication. Administrative users are identified by a user name and authenticated by an individual password associated with that user's account.

The TOE provides four interfaces for administrators to login remotely: Configuration Utility, iControl API, and iControl Rest API, and tmsh (via SSH). These interfaces communicate with the TOE via TLS (HTTPS). Connection to tmsh is via SSH version 2 or serial port console.

For local password-based authentication, section 7.3 states:

> *Administrative users (i.e., all users authorized to access the TOE's administrative interfaces) are identified by a user name and authenticated by an individual password associated with that user's account. This is true regardless of how the administrative user interfaces with the TOE. If the supplied user name and password match the user name and password pair maintained by the TOE, the administrative session is successfully established. Otherwise, the user receives an error and the session is not established. [...].*

The evaluator examined section 7.6 *TOE access* which states "For interactive user authentication at the web-based Configuration utility via HTTPS and the command line tmsh via SSH or the serial port console, BIG-IP implements the display of administrator-defined banners to users."

The evaluator determined that the TSS contains the necessary information.

Guidance Assurance Activities

Assurance Activity AA-FTP_ITC.1-AGD-01

The evaluator shall confirm that the guidance documentation contains instructions for establishing the allowed protocols with each authorized IT entity, and that it contains recovery instructions should a connection be unintentionally broken.

Summary

Section 6.2.7 of [ST] claims that the TOE provides a trusted communication between itself and the audit server via TLS.

The evaluator examined section 2.3.8 "Event (audit) logging" of [ECG] and determined that it provides instructions for setting up the syslog server. This section states that logging must be configured to use a dedicated network interface as described in section 2.3.8.1 "Configuring a dedicated network interface" of [ECG] . The configuration involves running the ccmode command to set up the Common Criteria mode which includes setting up basic network requirements. Also, section 2.3.8 states that should the connection between the BIG-IP and syslog server fail, the TOE will retry the connection an unlimited number of times until the connection can be re-established. During this time, log records will continue to be logged locally.

Test Assurance Activities

Assurance Activity AA-FTP_ITC.1-ATE-01

The developer shall provide to the evaluator application layer configuration settings for all secure communication mechanisms specified by the FTP_ITC.1 requirement. This information should be sufficiently detailed to allow the evaluator to determine the application layer timeout settings for each cryptographic protocol. There is no expectation that this information must be recorded in any public-facing document or report.

The evaluator shall perform the following tests:

a) Test 1: The evaluators shall ensure that communications using each protocol with each authorized IT entity is tested during the course of the evaluation, setting up the connections as described in the guidance documentation and ensuring that communication is successful.

b) Test 2: For each protocol that the TOE can initiate as defined in the requirement, the evaluator shall follow the guidance documentation to ensure that in fact the communication channel can be initiated from the TOE.

c) Test 3: The evaluator shall ensure, for each communication channel with an authorized IT entity, the channel data is not sent in plaintext.

d) Test 4: Objective: The objective of this test is to ensure that the TOE reacts appropriately to any connection outage or interruption of the route to the external IT entities.
The evaluator shall, for each instance where the TOE acts as a client utilizing a secure communication mechanism with a distinct IT entity, physically interrupt the connection of that IT entity for the following durations: i) a duration that exceeds the TOE's application layer timeout setting, ii) a duration shorter than the application layer timeout but of sufficient length to interrupt the network link layer.
The evaluator shall ensure that, when the physical connectivity is restored, communications are appropriately protected and no TSF data is sent in plaintext.
In the case where the TOE is able to detect when the cable is removed from the device, another physical network device (e.g. a core switch) shall be used to interrupt the connection between the TOE and the distinct IT entity. The interruption shall not be performed at the virtual node (e.g. virtual switch) and must be physical in nature.

Further assurance activities are associated with the specific protocols.

Summary

Test 1: As specified by the [ST] FTP_ITC.1 only specifies TLS to an audit server. Using a Kali Linux Linux machine the evaluator created CA and server certificates, and set up an OpenSSL s_server to act as an audit server. The evaluator then set up syslog on the TOE according to [ECG] . The evaluator then used tcpdump to record and analyze traffic between the TOE and the s_server, and verified that communication was successful.

Test 2: As specified by the [ST] FTP_ITC.1 only specifies TLS to an audit server. Using a Kali Linux Linux machine the evaluator created CA and server certificates, and set up an OpenSSL s_server to act as an audit server. The evaluator then set up syslog on the TOE according to [ECG] . The evaluator then used tcpdump to record and analyze traffic between the TOE and the s_server, and verified that the secure connection was initiated from the TOE.

Test 3: As specified by the [ST] FTP_ITC.1 only specifies TLS to an audit server. Using a Kali Linux Linux machine the evaluator created CA and server certificates, and set up an OpenSSL s_server to act as an audit server. The evaluator then set up syslog on the TOE according to [ECG] . The evaluator then used tcpdump to record and analyze traffic between the TOE and the s_server, and verified that data was not sent in plain text.

Test 4: As specified by the [ST] FTP_ITC.1 only specifies TLS to an audit server. Using a Kali Linux Linux machine the evaluator created CA and server certificates, and set up an OpenSSL s_server to act as an audit server. The evaluator then set up syslog on the TOE according to [ECG]. The evaluator then used tcpdump to record and analyze traffic between the TOE and the s_server when the physical connection was disrupted, both for the TOE application layer timeout setting and the MAC layer timeout. The evaluator verified that no plaintext data was transmitted.

Assurance Activity AA-FTP_ITC.1-ATE-02

For distributed TOEs the evaluator shall perform tests on all TOE components according to the mapping of external secure channels to TOE components in the Security Target.

Summary

The evaluator found that the TOE is not a distributed TOE, therefore there are no specific requirements for this assurance activity.

Assurance Activity AA-FTP_ITC.1-ATE-03

The developer shall provide to the evaluator application layer configuration settings for all secure communication mechanisms specified by the FTP_ITC.1 requirement. This information should be sufficiently detailed to allow the evaluator to determine the application layer timeout settings for each cryptographic protocol. There is no expectation that this information must be recorded in any public-facing document or report.

Summary

There are no specific testing requirements for this assurance activity. Please note that the evaluator performed as was able to configure application layer configuration settings for all secure communication mechanisms specified by the FTP_ITC.1 requirement.

2.1.7.2 Trusted Path (FTP_TRP.1/Admin)

TSS Assurance Activities

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

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

Summary

Chapter 7 of [ST] contains the TSS. Section 7.4 *Security Function Management* identifies the user interfaces to the TOE:

- Configuration Utility (Web-based GUI) - browser-based GUI interface with normal GUI panels and selections. The client browser talks to the Apache HTTP server over HTTPS; then the request passes through tomcat and to the BIG-IP.
- tmsh shell commands – provide a command line interface, accessible through an SSH client
- iControl API – SOAP-based programming interface over HTTPS
- iControl REST API – REST-based programming interface over HTTPS.

The TOE supports local and remote login methods. Local login is described in section 7.3 *Identification and Authentication* and remote login is described in section 7.2 *Cryptographic Support* and section 7.7 *Trusted Path/Channels* of the TSS. Local login is via password-based authentication. Administrative users are identified by a user name and authenticated by an individual password associated with that user's account.

The TOE provides four interfaces for administrators to login remotely: Configuration Utility, iControl API, and iControl Rest API, and tmsh (via SSH). These interfaces communicate with the TOE via TLS (HTTPS). Connection to tmsh is via SSH version 2 or serial port console.

For local password-based authentication, section 7.3 states:

> *Administrative users (i.e., all users authorized to access the TOE's administrative interfaces) are identified by a user name and authenticated by an individual password associated with that user's account. This is true regardless of how the administrative user interfaces with the TOE. If the supplied user name and password match the user name and password pair maintained by the TOE, the administrative session is successfully established. Otherwise, the user receives an error and the session is not established. [...].*

The evaluator examined section 7.6 *TOE access* which states "For interactive user authentication at the web-based Configuration utility via HTTPS and the command line tmsh via SSH or the serial port console, BIG-IP implements the display of administrator-defined banners to users."

The evaluator determined that the TSS contains the necessary information.

Guidance Assurance Activities

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

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

Summary

The evaluator examined sections 2.2 "Perform Basic Installation and Configuration" and 2.3 "Common Criteria configuration" of [ECG] and determined it contains instructions for establishing the remote administrative sessions through SSH, which is done by running the ccmode command on tmsh, per section 2.3.4.1 "SSH" of [ECG] . It can also be set up manually using the instructions from the Traffic Management Shell (tmsh) Reference Guide [TMSH-REFv12] (section "sshd") and [TMSH-REFv17] (sys sshd). Logging on to the GUI is via a web browser per section 2.3.4.2 "GUI" of [ECG] .

The evaluator also examined section 3.7 "Additional Management Interfaces" of [ECG] which provides information about the iControl and iControl REST API which are programmatic management interfaces over HTTPS. Additional details about these interfaces are provided in their respective guidance documentation [ICREST] and [ICONTROL] .

Test Assurance Activities

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

> *The evaluator shall perform the following tests:*
>
> *a) Test 1: The evaluators shall ensure that communications using each specified (in the guidance documentation) remote administration method is tested during the course of the evaluation, setting up the connections as described in the guidance documentation and ensuring that communication is successful.*
>
> *b) Test 2: The evaluator shall ensure, for each communication channel, the channel data is not sent in plaintext.*
>
> *Further assurance activities are associated with the specific protocols.*

Summary

The described methods are: SSH, TLS and HTTPS. The evaluator performed the test as part of other SFRs: FCS_SSHS_EXT.1 and FCS_TLS_EXT.1. Each protocol was set up as specified in the guidance documentation and the connection was successful. The evaluator also ensured that no data was sent in plaintext by capturing and analysing the traffic.

Assurance Activity AA-FTP_TRP.1-ADMIN-ATE-02

> *For distributed TOEs the evaluator shall perform tests on all TOE components according to the mapping of trusted paths to TOE components in the Security Target.*

Summary

The toe is not a distributed TOE. Therefore, there are no specific requirements for this assurance activity.

2.2 Security Assurance Requirements

2.2.1 Security Target evaluation (ASE)

2.2.1.1 TOE summary specification (ASE_TSS.1)

Assurance Activity AA-ASE_TSS.1-ASE-01

For distributed TOEs only the SFRs classified as 'all' have to be fulfilled by all TOE parts. The SFRs classified as 'One' or 'Feature Dependent' only have to be fulfilled by either one or some TOE parts, respectively.

To make sure that the distributed TOE as a whole fulfills all the SFRs the following actions for ASE_TSS.1 have to be performed as part of ASE_TSS.1.1E.

Summary

According to the Security Target [ST] , the TOE is not distributed therefore this Evaluation Activity is not applicable.

Assurance Activity AA-ASE_TSS.1-ASE-02

The evaluator shall examine the TSS to determine that it is clear which TOE components contribute to each SFR or how the components combine to meet each SFR.

The evaluator shall verify the sufficiency to fulfil the related SFRs. This includes checking that the TOE as a whole fully covers all SFRs and that all functionality that is required to be audited is in fact audited regardless of the component that carries it out.

This evaluation activity is supplementary to ASE_TSS.1-1.

Summary

While performing the evaluator actions for ASE_TSS.1E, the evaluator verified that the TSS sufficiently described how each SFR is met by the TOE. Also, while performing other Evaluation Activities, the evaluator thoroughly examined the TSS and determined that it clearly described which TOE components contribute to each SFR, for example, the cryptographic support is mainly provided by OpenSSL, audit record generation is provided by the syslog functionality, etc. The evaluator also verified that the TSS describes how all the TOE components combine to meet each and all of the SFRs, for example, OpenSSL works with other components to provide secure communications between TOE and the audit server and likewise between the TOE and the supported administrative interfaces such as the tmsh and Web GUI. Additionally, section 7.1 *Security Audit* of the TSS describes how the TOE audits all the required events defined in FAU_GEN.1.

Assurance Activity AA-ASE_TSS.1-ASE-03

The evaluator shall examine the TSS to identify any extra instances of TOE components allowed in the ST and shall examine the description of how the additional components maintain the SFRs to confirm that it is consistent with the role that the component plays in the evaluated configuration. For example: the secure channels used by the extra component for intra-TOE communications (FPT_ITT) and external communications (FTP_ITC) must be consistent, the audit information generated by the extra component must be maintained, and the management of the extra component must be consistent with that used for the original instance of the component in the minimum configuration.

Summary

While performing other Evaluation Activities, the evaluator thoroughly examined the TSS and determined that the TSS does not identify any extra instances of the TOE components allowed in the ST. In other words, the TSS only describes the TOE as consistent with the TOE description of [ST] . Specifically, section 1.5 *TOE Overview* contains the following statement:

> *BIG-IP products run on appliance or blade hardware provided by F5 listed in Section 1.2.1 or on one of the hypervisors listed in Section 1.2.2. When running on a third-party hypervisor, there may be only one guest virtual machine running on the hypervisor and only one instance of BIG-IP for each hardware platform. In addition, BIG-IP running as a guest instance on F5 devices (appliances or blades) that support F5's Virtual Clustered Multiprocessing (vCMP) environment is included. (vCMP implements a purpose-built embedded hypervisor that allows organizations to run multiple virtual instances of BIG-IP on the same hardware.)*

2.2.2 Development (ADV)

2.2.2.1 Basic functional specification (ADV_FSP.1)

Assurance Activity AA-ADV_FSP.1-ADV-01

The evaluator shall examine the interface documentation to ensure it describes the purpose and method of use for each TSFI that is identified as being security relevant.

This evaluation activity is supplemental for work units ADV_FSP.1-1 and ADV_FSP.1-2.

Summary

Based on the Security Target ([ST]), main administrative guidance ([ECG]), and the other documentation provided by the developer, the evaluator created the following table containing the TSFIs for the TOE, the locations in the guidance where the purpose and use of the TSFI is described, and if the TSFI is SFR-enforcing.

Table 7: TSFI

TSFI	Description
TMSH (a.k.a. CLI) SFR-enforcing	The Traffic Management Shell is the command-line administrative interface to the TOE. The administrator must use a TLS protected secure shell (SSH) connection to the TOE to issue tmsh commands. The following references provide additional descriptive information: • [ECG] section 2.1.1.2, "Establishing Administrative Access" • [ST] sections 1.6.4.4, "Security Management," 7.2.5, "SSH," and 7.7, "Trusted Path/Channels" • [TMSH-REFv17]
Audit log SFR-enforcing	An interface used to manage and store event log data during system operation. Data is transferred to an audit server over a TLS protected connection. The following references provide additional descriptive information: • [ECG] section 2.3.8, "Event (audit) logging," and Appendix 9, "Audit and Event Records." • [ST] sections 1.6.4.1, "Security Audit," and 7.1, "Security Audit." • [LTMMR] section "About Logging."

TSFI	Description
Serial console SFR-enforcing	A local administrative interface for managing the TOE over a serial connection using a null modem. The following references provide additional descriptive information: • [ECG] section 3.2, "Maximum Failed Login Attempts" • [ECG] section 2.2.4, "Setting up the banner for the serial console" • [ST] section 1.6.4.4, "Security Management" • [K6068]
GUI SFR-enforcing	The web-based graphical remote interface is known as the Configuration Utility. The administrator connects to the TOE using a secure connection (TLS over HTTPS). The following references provide additional descriptive information: • [ECG] section 2.1.1.2, "Establishing Administrative Access." • [ST] sections 1.6.4.4, "Security Management," 7.2.7, "HTTPS Protocol," and 7.6, "TOE Access." • [GSG] chapter 1, section "Choosing a configuration tool."
iControl SOAP (API) SFR-enforcing	A programmatic interface supporting the Simple Object Access Protocol (SOAP) XML-based messaging protocol. The administrator sends SOAP requests to the TOE over HTTPS (which is protected by TLS). The following references provide additional descriptive information: • [ECG] section 2.1.1.2, "Establishing Administrative Access." • [ST] sections 1.6.4.4, "Security Management," 7.2.6, "TLS Protocol," and 7.7, "Trusted Path/Channels." • [ICONTROL] iControl Guidance Documentation (available on-line), specifically in sdk/overview_docs/.
iControl REST (API) SFR-enforcing	A programmatic interface supporting the Representational State Transfer (REST) web services architecture. The administrator sends RESTful requests to the TOE over HTTPS (which is protected by TLS). The following references provide additional descriptive information: • [ECG] section 2.1.1.2, "Establishing Administrative Access." • [ST] sections 1.6.4.4, "Security Management," 7.2.6, "TLS Protocol," and 7.7, "Trusted Path/Channels." • [ICREST].
SSH SFR-enforcing	SSH, secure shell, is used to protect the communication traffic between the administrator and the TMSH. Secure shell encrypts the data sent and uses TLS to aid in the protection of data in transit. The following references provide additional descriptive information: • [ST] sections 1.6.4.4, "Security Management," 7.2.5, "SSH," 7.5.1, "Protection of Sensitive Data," and 7.7, "Trusted Path/Channels".
TLS SFR-enforcing	The TLS protocol is used to protect data in transit. Other interfaces depend on this protocol to provide protection. E.g., audit logs, SSH, and HTTPS. The following references provide additional descriptive information:

TSFI	Description
	• [ST] sections 1.6.4.4, "Security Management," 7.2.6, "TLS Protocol," 7.5.1, "Protection of Sensitive Data," and 7.7, "Trusted Path/Channels"
HTTPS SFR-enforcing	The HTTPS protocol is used to protect data in transit. TSFIs that are accessible via the web are protected with HTTPS (which will use TLS). E.g., GUI, iControl REST, and iControl SOAP. The following references provide additional descriptive information: • [ST] sections 1.6.4.4, "Security Management," 7.2.7, "HTTPS Protocol," and 7.4, "Security Function Management"
SYSLOG SFR-enforcing	A programmatic interface supporting the Representational State Transfer (REST) web services architecture. The administrator sends RESTful requests to the TOE over HTTPS (which is protected by TLS). The following references provide additional descriptive information: • [ECG] section 2.1.1.2, "Establishing Administrative Access." • [ST] sections 1.6.4.4, "Security Management," 7.2.6, "TLS Protocol," and 7.7, "Trusted Path/Channels." • [ICREST].

The evaluator determined that for each TSFI, the purpose, method of use and all parameters, are documented as described in the table above.

Assurance Activity AA-ADV_FSP.1-ADV-02

The evaluator shall check the interface documentation to ensure it identifies and describes the parameters for each TSFI that is identified as being security relevant.

This evaluation activity is supplemental for work units ADV_FSP.1-3.

Summary

The evaluator performed this examination in AA-ADV_FSP.1-ADV-01. The results of this analysis can be found in the same evaluation activity.

Assurance Activity AA-ADV_FSP.1-ADV-03

The evaluator shall examine the interface documentation to develop a mapping of the interfaces to SFRs.

This evaluation activity is supplemental for work units ADV_FSP.1-5.

Summary

The evaluator performed part of this examination in AA-ADV_FSP.1-ADV-01.

Additionally, the evaluator created the following table to provide information about SFRs that did not manifest themselves through an interface.

Table 8: SFRs not manifested through a TSFI

SFR component	Rationale
FAU_STG.1	Per the ST, audit records are stored locally in syslog files managed, and protected against unauthorized access, by using file permission bits in the underlying Linux host. The evaluator determined this is sufficient to support this SFR not being manifested through an interface.
FCS_CKM.4	Seeds and numbers from the key generation process are zeroized after the key has been generated. All session keys are zeroized when the session has ended. There is no interface into destruction of seeds and numbers; and session keys. Keys stored on the disk (i.e., SSH and TLS private keys) are zeroized by the administrator. There is an API the administrator can use from the tmsh (which is manifested through a TSFI). The evaluator determined this SFR is partially manifested through a TSFI. Since there is a portion of the SFR that is not manifested through a TSFI nor an interface, the evaluator decided to place this SFR in this section and explain how it is addressed by the TOE.
FCS_RBG_EXT.1	The TOE uses its entropy sources to fill the entropy pool. The entropy pool is used as a seed source for the DRNG. The evaluator determined this is sufficient to support this SFR not being manifested through an interface.
FPT_APW_EXT.1	Certain sensitive data is stored in the TOE's configuration files. This includes pre-shared, symmetric, private keys, and passwords. The TOE does not offer an interface to retrieve passwords, configuration files, or the contents of configuration files. The evaluator determined this is sufficient to support this SFR not being manifested through an interface.
FPT_SKP_EXT.1	This is addressed as part of FPT_APW_EXT.1. The evaluator determined this is sufficient to support this SFR not being manifested through an interface.
FPT_TST_EXT.1/PowerOn	The power-on self tests are performed internally (and automatically) by the TOE's cryptographic modules. The evaluator determined this is sufficient to support this SFR not being manifested through an interface.

2.2.3 Guidance documents (AGD)

2.2.3.1 Operational user guidance (AGD_OPE.1)

Assurance Activity AA-AGD_OPE.1-AGD-01

The evaluator shall ensure the Operational guidance documentation is distributed to Security Administrators and users (as appropriate) as part of the TOE, so that there is a reasonable guarantee that Security Administrators and users are aware of the existence and role of the documentation in establishing and maintaining the evaluated configuration.

Summary

The evaluator gained assurance that the guidance documentation will be distributed to administrators (users) as part of the TOE, and the administrators are aware of the existence of role of the documentation in establishing and maintaining the evaluated configuration, based on the following findings:

- Section 1.6.3.2 "Guidance Documentation" in [ST] lists the guidance documentation that are a part of the TOE.
- Section 1.1 "References" in [ECG] clearly lists all the guidance documents that are included in the TOE.
- As pointed out in section 1.1, the guidance documentation can be easily downloaded from F5 website.

Assurance Activity AA-AGD_OPE.1-AGD-02

The evaluator shall ensure that the Operational guidance is provided for every Operational Environment that the product supports as claimed in the Security Target and shall adequately address all platforms claimed for the TOE in the Security Target.

Summary

The developer provides [ECG] which serves as the main user guidance to prepare the TOE and its operational environment required by the evaluator configuration. In addition, the developer provided the following user guidance for the supported platforms listed in section 1.2 *TOE Identification* of [ST] :

- [PGi15000] Platform Guide: i15000 Series
- [PGi20000] Platform Guide: i2000/i4000 Series
- [PGi11000] Platform Guide: i5000/i7000/i10000/i11000 Series
- [PG2200] Platform Guide: VIPRION 2200 [2]
- [PG4400] Platform Guide: VIPRION 4400 Series
- [VCMPAMA] vCMP for Appliance Models: Administration
- [VCMPVMA] vCMP for VIPRION Systems: Administration
- [BIGIPKVM] BIG-IP VE in Linux KVM
- [BIGIPVMWare] BIG-IP VE in VMware ESXi
- [VEPLATFORMS] BIG-IP Virtual Edition Supported Platforms
- [K14810] K14810: Overview of BIG-IP VE license and throughput limits
- [K14946] K14946: Overview of BIG-IP VE image sizes
- [KVMSETUP] Linux KVM – BIG-IP VE Setup
- [KVMUG] Linux KVM – BIG-IP VE Users Guide
- [KVMCRYPTO] Linux KVM – Configure cryptographic offload for BIG-IP VE with Intel QAT
- [HyperVSETUP] Microsoft Hyper-V – BIG-IP VE Setup
- [HyperVUG] Microsoft Hyper-V – BIG-IP VE Users Guide
- [SWUPDATE] Update BIG-IP VE
- [VMwareSETUP] VMware ESXi – BIG-IP VE Setup

[2] Per F5 development, the VIPRION 2200 Platform Guide applies to the VIPRION C2400 model series.

- [VMwareUG] VMware ESXi – BIG-IP VE Users Guide

The evaluator determined that [ECG] applies to all the supported platforms.

Assurance Activity AA-AGD_OPE.1-AGD-03

The evaluator shall ensure that the Operational guidance contains instructions for configuring any cryptographic engine associated with the evaluated configuration of the TOE. 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 TOE.

Summary

In the evaluated configuration the TOE uses the default cryptographic engines described in [ST] . [ECG] section 2.1 contains the following statements:

> *The cryptographic operations in BIG-IP are configured at the protocol level, via the ccmode utility, and via instructions in this guide.*

Assurance Activity AA-AGD_OPE.1-AGD-04

The evaluator shall ensure the Operational guidance makes it clear to an administrator which security functionality and interfaces have been assessed and tested by the EAs.

Summary

Although [ECG] does not explicitly lists which security functionality and interfaces have been assessed and tested by the EAs, the evaluator was able to determine from the information provided in that document that all security functionality and interfaces claimed and described in [ST] were indeed assessed and tested by the EAs. The following table summarizes the evaluator findings.

Security functionality	Interfaces	Provided Guidance
Security audit	tmsh, GUI	[ECG] sections 2.3.8, 3.3, 4, 9, and 10. These sections also refer to additional guidance such as section 2.3.8 refers to [CLUSTERADM] , [CLUSTERADM] , [LTMMR] , and [TMOSI] .
Cryptographic support	tmsh, GUI, iControl, iControl REST	[ECG] sections 2.2.3, 2.2.6, 2.3.10, chapters 4 and 5
Identification & authentication	tmsh, GUI	[ECG] sections 2.2.7, 2.3.3, 2.3.4, 2.3.5, 3.1, 3.2 These sections also refer to additional guidance, section 2.3.4 refers to [K13092] , [K13454] , and [K42531434] .
Security function management	tmsh, GUI, iControl, iControl REST.	[ECG] chapters 2, 3, and 4. These chapters also refer to other user guides for additional guidance such as [TMSH-REFv12] , [TMSH-REFv17] for general security management; [USRADM] for user account management; [K15664] , [K14620] , [K15462] , [K14806] , [K14783] , and [K13302] for certificate management related guidance; and [K15497] for password policy.

Security functionality	Interfaces	Provided Guidance
Protection of the TSF	tmsh, iControl, iControl REST, GUI	[ECG] sections 2.1, 2.2.1, 2.2.2, 2.3.11, chapter 4. These sections and chapter also refer to other user guides for additional guidance such as section 2.2.2 refers to [TMSH-REFv12] and [TMSH-REFv17] for trusted update related guidance; section 2.3.11 refers to [TMSH-REFv12] and [TMSH-REFv17] and [ESSEN] for system time configuration related guidance.
TOE access	tmsh, GUI	[ECG] sections 2.2.4, 2.3.5, 3.2, chapter 4. These sections and chapter refer to other user guides for additional guidance such as section 2.2.4 refers to [K6068] ; section 2.3.5 refers to [TMSH-REFv12] and [TMSH-REFv17] .
Trusted path/channels	tmsh, GUI, iControl, iControl REST	[ECG] sections 2.2.3, 2.2.5, 2.3.8, 2.3.9, 2.3.10, chapters 4, 5, and 10. These sections and chapters refer to other user guides for additional guidance such as section 2.3.8 refers to [TMOSRA] ; section 2.3.9 refers to [K15664] , [K14620] , [K15462] , [K14806] , and [K14783] .

Additionally, section 1.2.3 of [ECG] explicitly lists the items, e.g., security functionality and interfaces, that are not supported in the evaluated configuration, i.e., not assessed and tested by the assurance activities such as remote server configuration, lmi shell, iRulesLX and iAppsLX. Furthermore, section 3.5 "Commands and APIs not Allowed in the Evaluated Configuration" discusses the tmsh commands and APIs that are disallowed in the evaluated configuration" and refers to sections 7 and 8 of [ECG] for the listing of disallowed tmsh commands and iControl APIs, respectively.

Assurance Activity AA-AGD_OPE.1-AGD-05

In addition, the evaluator shall ensure that the following requirements are also met.

a) The guidance documentation shall contain instructions for configuring any cryptographic engine associated with the evaluated configuration of the TOE. 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 TOE.

b) [TD0536] The documentation must describe the process for verifying updates to the TOE for each method selected for FPT_TUD_EXT.1.3 in the Security Target. The evaluator shall verify that this process includes the following steps:

1) Instructions for obtaining the update itself. This should include instructions for making the update accessible to the TOE (e.g., placement in a specific directory).

2) Instructions for initiating the update process, as well as discerning whether the process was successful or unsuccessful. This includes instructions that describe at least one method of validating the hash/digital signature.

c) The TOE will likely contain security functionality that does not fall in the scope of evaluation under this cPP. The guidance documentation shall make it clear to an administrator which security functionality is covered by the Evaluation Activities.

Summary

Regarding trusted update, the evaluator examined section 2.2.2.3 "Updating BIG-IP software after initial configuration" of [ECG] which states that the processing of updating the TOE (software) is the same as the initial installation (except the administrator does not need to verify the image) as described in section 2.2.2 "Re-install the BIG-IP software". According to section 2.2.2, the TOE

software available as an ISO download is digitally signed and verified as part of the ccmode command. Alternatively, the administrator can manually verify the ISO download by following the instructions provided in section 2.2.2. 2 "Verifying the product ISO using the digital signature". Section 2.2.2.3 "Updating BIG-IP software after initial configuration" of [ECG] also refers to [SWUPDATE] for updating BIG-IP VE . It states that validation is performed as described for the image file download in section 2.2.2.2 "Verifying the product ISO using the digital signature".

If the signature verification fails, the installation will fail. Either download the ISO again or contact F5 support.

Successful verification can be demonstrated by checking the installed software via the command `tmsh show sys software status` .

Assurance Activity AA-AGD_OPE.1-AGD-06

> *The evaluator shall examine the description of the extra instances of TOE components in the guidance documentation to confirm that they are consistent with those identified as allowed in the ST. This includes confirmation that the result of applying the guidance documentation to configure the extra component will leave the TOE in a state such that the claims for SFR support in each component are as described in the ST and therefore that all SFRs continue to be met when the extra components are present.*
>
> *The evaluator shall examine the secure communications described for the extra components to confirm that they are the same as described for the components in the minimum configuration (additional connections between allowed extra components and the components in the minimum configuration are allowed of course).*

Summary

Per section 1 *Introduction* of [ST] , the TOE is the entire network device/appliance. Additionally, section 2.1 *Preparing for BIG-IP Installation and Configuration* of [ECG] describes the assumptions on the TOE and its operational environment including what is allowed and not allowed in the evaluated configuration.

2.2.3.2 Preparative procedures (AGD_PRE.1)

Assurance Activity AA-AGD_PRE.1-AGD-01

> *The evaluator shall examine the Preparative procedures to ensure they include a description of how the Security Administrator verifies that the operational environment can fulfil its role to support the security functionality (including the requirements of the Security Objectives for the Operational Environment specified in the Security Target).*

Summary

The evaluator examined section 1 "Introduction" and section 2 "Installation and Configuration Procedures" of [ECG] , and determined they include a description of how the administrator verifies that the operational environment can fulfill its role to support the security functionality, based on the following information:

- Section 2.1 "Preparing for BIG-IP Installation and Configuration" of [ECG] describes what the administrators need to do in order to fulfill the security objectives of the operational environment. The evaluator found that [ECG] provides sufficient preparative procedures that can be performed by the administrator to verify the TOE and TOE environment are set up properly so that the environment can fulfill its role of supporting the security functionality of the TOE.

Assurance Activity AA-AGD_PRE.1-AGD-02

The evaluator shall examine the Preparative procedures to ensure they are provided for every Operational Environment that the product supports as claimed in the Security Target and shall adequately address all platforms claimed for the TOE in the Security Target.

Summary

The developer provides [ECG] which serves as the main user guidance to prepare the TOE and its operational environment required by the evaluated configuration. In addition, the developer provided the following user guidance for the supported virtual platforms listed section 1.2 *TOE Identification* of [ST] :

- [PGi15000] Platform Guide: i15000 Series
- [PGi20000] Platform Guide: i2000/i4000 Series
- [PGi11000] Platform Guide: i5000/i7000/i10000/i11000 Series
- [PG2200] Platform Guide: VIPRION 2200 [3]
- [PG4400] Platform Guide: VIPRION 4400 Series
- [VCMPAMA] vCMP for Appliance Models: Administration
- [VCMPVMA] vCMP for VIPRION Systems: Administration
- [BIGIPKVM] BIG-IP VE in Linux KVM
- [BIGIPVMWare] BIG-IP VE in VMware ESXi
- [VEPLATFORMS] BIG-IP Virtual Edition Supported Platforms
- [K14810] K14810: Overview of BIG-IP VE license and throughput limits
- [K14946] K14946: Overview of BIG-IP VE image sizes
- [KVMSETUP] Linux KVM – BIG-IP VE Setup
- [KVMUG] Linux KVM – BIG-IP VE Users Guide
- [KVMCRYPTO] Linux KVM – Configure cryptographic offload for BIG-IP VE with Intel QAT
- [HyperVSETUP] Microsoft Hyper-V – BIG-IP VE Setup
- [HyperVUG] Microsoft Hyper-V – BIG-IP VE Users Guide
- [SWUPDATE] Update BIG-IP VE
- [VMwareSETUP] VMware ESXi – BIG-IP VE Setup
- [VMwareUG] VMware ESXi – BIG-IP VE Users Guide

Assurance Activity AA-AGD_PRE.1-AGD-03

The evaluator shall examine the preparative procedures to ensure they include instructions to successfully install the TSF in each Operational Environment.

Summary

While performing other assurance activities, the evaluator determined that the preparative procedures include instructions to successfully install the TSF in the operational environment, which is described in section 2 "Installation and Configuration Procedures" of [ECG] and examined AGD_PRE.1-2.

This is supported by the evaluator's independent testing where the evaluator followed the provided guidance particularly [ECG] to prepare the TOE/TSF in the respective operational environment.

Assurance Activity AA-AGD_PRE.1-AGD-04

[3] Per F5 development, the VIPRION 2200 Platform Guide applies to the VIPRION C2400 model series.

The evaluator shall examine the preparative procedures to ensure they include instructions to manage the security of the TSF as a product and as a component of the larger operational environment.

Summary

The evaluator determined that the TOE is a network device, which is a product. Also, taking into account that the TOE is not a distributed TOE thus, when it is used in a larger operational environment, the TOE still acts as an individual product in the environment. Therefore, it does not require separate instructions for the TSF as a component of the larger operational environment.

Assurance Activity AA-AGD_PRE.1-AGD-05

In addition, the evaluator shall ensure that the following requirements are also met.

The preparative procedures must

a) include instructions to provide a protected administrative capability; and

b) identify TOE passwords that have default values associated with them and instructions shall be provided for how these can be changed.

Summary

Setting up the administrative capability is described in chapter 2 of [ECG] , particularly in the following sections:

- Section 2.1.1.2 *Establishing Administrative Access* describes the interfaces to administer the TOE which are: tmsh over SSH, Web GUI over HTTPS, and the programming interfaces iControl SOAP or iControl REST over TLS.
- Section 2.2.3.1 *Using SSH public-key authentication* provides instructions to set up SSH.
- Section 2.3.1 *ccmode command* provide instructions to execute the ccmode command which performs functions such as setting the required password policy, the allowed TLS ciphersuites, as well as auditing options.
- Section 2.3.4 *Login to the BIG-IP* describes how to log into the TOE using SSH (via tmsh) and the Web GUI (via HTTPS).
- Changing passwords is described in section 3.1 of [ECG] which provides step-by-step instruction how to enforce the password policy as well as how to change a password.

2.2.4 Tests (ATE)

2.2.4.1 Independent testing - conformance (ATE_IND.1)

Assurance Activity AA-ATE_IND.1-ATE-01

The evaluator tests the TOE in the minimum configuration as defined in the ST (and the guidance documentation).

If the description of the use of extra components in the ST and guidance documentation identifies any difference in the SFRs allocated to a component, or the scope of the SFRs involved (e.g. if different selections apply to different instances of the component) then the evaluator tests these additional SFR cases that were not included in the minimum configuration.

In addition the evaluator tests the following aspects for each extra component that is identified as allowed in the distributed TOE:

- *Communications: the evaluator follows the guidance documentation to confirm, by testing, that any additional connections introduced with the extra component and not present in the minimum configuration are consistent with the requirements stated in the ST (e.g. with regard to protocols and ciphersuites used). An example of such an additional connection would be if a single instance of the component is present in the minimum configuration*

and adding a duplicate component then introduces an extra communication between the two instances. Another example might be if the use of the additional components necessitated the use of a connection to an external authentication server instead of using locally stored credentials.

- *Audit: the evaluator confirms that the audit records from different instances of a component can be distinguished so that it is clear which instance generated the record.*
- *Management: if the extra component manages other components in the distributed TOE then the evaluator shall follow the guidance documentation to confirm that management via the extra component uses the same roles and role holders for administrators as for the component in the minimum configuration.*

Summary

The evaluator created [NDETP] which describes the test environment and each test case with sufficient detail to enable tests reproducibility. It also includes the test verdict and the test results.

The evaluator described the testing environment in [NDETP] 2 "Test setup and TOE preparation", i.e. versions of the TOEs to be tested, the setup of the test environment and the test equipment used during testing. Chapter 3 "Test case description" provides for each test case:

- Prerequisite
- Test procedure
- Expected outcome
- Test result
- Test verdict

The evaluator documented the detailed procedure for some tests and test evidence like WireShark captures and logs in [TestEvidence].

2.2.5 Vulnerability assessment (AVA)

2.2.5.1 Vulnerability Survey (AVA_VAN.1)

Assurance Activity AA-AVA_VAN.1-AVA-01

The calibration of test resources specified in paragraph 1418 of the [CEM] *applies to the tools listed in Section A.1.4 of the* [NDcPPv2.2-SD].

This evaluation activity is supplemental for work unit AVA_VAN.1-1.

Summary

The evaluator considered section 7.6 in [NDcPPv2.2e] and the modifications to work units in AVA_VAN.1 specified in section 5.6.1 of [NDcPPv2.2-SD] in performance of the CEM work units.

The evaluator used the following vulnerability databases for the pubic vulnerability search:

- Common Vulnerabilities and Exposures (CVE)
  https://cve.mitre.org/index.html
  The CVE database at MITRE is the largest and most comprehensive source of known vulnerabilities. Other publicly known databases searched by the evaluator were subsets at best, therefore the evaluator used the CVE database as the basis for the analysis.
- OpenSSL website
  https://www.openssl.org/news/vulnerabilities.html
  The evaluator only searched this site for OpenSSL vulnerabilities.
- OpenSSH website
  https://www.openssh.com/security.html

The evaluator only searched this site for OpenSSH vulnerabilities.

- The developer's website for security publications https://support.f5.com/csp/home

The TOE includes a number of third-party components specified by the developer in K51874520 BIG-IP third-party software matrix, https://support.f5.com/csp/article/K51874520 . Using this and other guidance, the evaluator created the following list of search terms:

- OpenSSL
- OpenSSH
- Apache
- BIND
- Curl
- JDBC
- OpenJDK (java)
- MarieDB
- Perl
- PHP
- PostgreSQL
- Python
- Net-SNMP
- NTP
- Node.js
- sSMTP
- syslog-ng
- Tomcat
- ZebOS

Section A.1.1 of [NDcPPv2.2-SD] requires the use of additional search keywords: router, switch, and TCP. These keywords greatly increase the number of responses but do not find any vulnerabilities not already identified in previous searches.

All searches were performed between 2022-08-16 and 2022-08-19. The evaluator also performed additional searches on all publicly accessible sites during the periods 2022-09-19 - 2022-09-22, 2022-10-07 - 2022-10-10, 2022-10-17 - 2022-10-19, 2022-11-01 - 2022-11-03 and 2022-12-08 - 2022-12-10 to verify that no new vulnerabilities had been published.

Note that the TOE already includes fixes for Spectre and Meltdown flaws, CVE-2017-5715, CVE-2017-5753, and CVE-2017-5754 as described in https://support.f5.com/csp/article/K91229003 , as well as a recent Bleichenbacher attack, CVE-2017-6168 as described in https://support.f5.com/csp/article/K21905460 .

The developer has provided the following information concerning mitigation of the recent Intel-based microarchitectural vulnerabilities like Zombieload.

When a site has a single administrative domain operating F5 Networks, Inc.’s BIG-IP, the only roles able to exploit these vulnerabilities are the administrators, so there would be no point to exploitation. Sites that are operating multiple administrative domains using the only supported way to do that with BIG-IP, vCMP (Virtual Clustered Multiprocessing), can fully mitigate the issues by ensuring that all untrusted vCMP guests are allocated more than one CPU core.

F5 Networks, Inc. has posted the following Security Advisory articles on the F5 Knowledge Center regarding the issues associated with the Intel processors along with their corresponding CVE number so that the F5 customers are aware of the issues. F5 will update the Knowledge Center articles as more information becomes available.

- K41283800: Intel-SA-00233 Microarchitectural Data Sampling Advisory - https://support.f5.com/csp/article/K41283800
- K52370164: Microarchitectural Store Buffer Data Sampling (MSBDS) CVE-2018-12126 - https://support.f5.com/csp/article/K52370164
- K97035296: Microarchitectural Load Port Data Sampling - Information Leak (MLPDS) CVE-2018-12127 - https://support.f5.com/csp/article/K97035296
- K80159635: Microarchitectural Fill Buffer Data Sampling (MFBDS) CVE-2018-12130 - https://support.f5.com/csp/article/K80159635
- K34303485: Microarchitectural Data Sampling Uncacheable Memory (MDSUM) CVE-2019-11091 - https://support.f5.com/csp/article/K34303485

No potential vulnerabilities were found to be applicable to the TOE, thus the evaluator identified no need for additional testing. The evaluator did perform a port scan of the TOE and found found no unexpected open ports, as expected.

The evaluator also performed fuzzy testing to generate flaw hypotheses. The following types of fuzzy testing was performed:

- the evaluator created mutated ICMPv4 and ICMPv6 packets carrying undefined "Type" field values (values of 44-252) and undefined "Code" values (values of 0-15)
- the evaluator created mutated IPv4 and IPv6 packets carrying undefined "Protocol" field values (values of 21-62, 66-68, 72-75, 80-254)
- the evaluator performed fuzzy testing on ICMPv4 and ICMPv6 Header fields: “sequence", "id", "code", "type", "checksum" one at a time
- the evaluator performed fuzzy testing on UDP Header fields: “source port" and"destination port" one at a time
- the evaluator performed fuzzy testing on TCP Header fields: “sequence number", "acknowledgement number", "source port", "destination port", "flags", "reserved", "window size", "data offset", "urgent pointer" one at a time

The evaluator created python scripts that utilized Scapy (version 2.4.0) in order to perform fuzzy testing. The evaluator did not detect any unexpected TOE behavior, only valid packets were processed by the TOE.

Assurance Activity AA-AVA_VAN.1-AVA-02

The evaluator shall examine the documentation outlined below provided by the vendor to confirm that it contains all required information. This documentation is in addition to the documentation already required to be supplied in response to the EAs listed previously.

[TD0547] The developer shall provide documentation identifying the list of software and hardware components that compose the TOE. Hardware components should identify at a minimum the processors used by the TOE. Software components include applications, the operating system and other major components that are independently identifiable and reusable (outside of the TOE), for example a web server, protocol or cryptographic libraries, (independently identifiable and reusable components are not limited to the list provided in the example). This additional documentation is merely a list of the name and version number of the components and will be used by the evaluators in formulating vulnerability hypotheses during their analysis.

Summary

The evaluation consists of the following models as provided in [ST] .:

- BIG-IP i4800
- BIG-IP i7800 vCMP guest
- BIG-IP VE running KVM Hypervisor on a Ubuntu 20.04 Linux installed on a Dell PowerEdge R630 with Intel Xeon processor E5-2600 v4.

Third-party software components, including cryptographic libraries, included in the TOE are found on the developer web site: https://support.f5.com/csp/article/K51874520 .

Assurance Activity AA-AVA_VAN.1-AVA-03

If the TOE is a distributed TOE then the developer shall provide:

a) documentation describing the allocation of requirements between distributed TOE components as in [NDcPPv2.2e], *section 3.4*

b) a mapping of the auditable events recorded by each distributed TOE component as in [NDcPPv2.2e], *section 6.3.3*

c) additional information in the Preparative Procedures as identified in the refinement of AGD_PRE.1 in additional information in the Preparative Procedures as identified in sections 3.4.1.2 and 3.5.1.2 of [NDcPPv2.2-SD].

Summary

The TOE is not a distributed TOE, therefore the evaluator determines this work unit to be not applicable.

Assurance Activity AA-AVA_VAN.1-AVA-04

The evaluator formulates hypotheses in accordance with process defined in Appendix A of [NDcPPv2.2-SD]. *The evaluator documents the flaw hypotheses generated for the TOE in the report in accordance with the guidelines in Appendix A.3. The evaluator shall perform vulnerability analysis in accordance with Appendix A.2. The results of the analysis shall be documented in the report according to Appendix A.3.*

Summary

The evaluator considered the four types of flaw hypotheses described in section A.1 of [NDcPPv2.2-SD] and determined that none provided any improvement to vulnerability analysis over the results of the CVE searches and penetration test.

A Appendixes

A.1 References

BIGIPKVM **BIG-IP VE in Linux KVM**
Date 2020-07-14
File name agd/BIG-IP VE in Linux KVM.pdf

BIGIPVMWare **BIG-IP VE in VMWare ESXi**
Date 2020-10-21
File name agd/BIG-IP VE in VMware ESXi.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

CCEVS-TD0527 **Updates to Certificate Revocation Testing (FIA_X509_EXT.1)**
Date 2020-07-01
Location https://www.niap-ccevs.org/Documents_and_Guidance/view_td.cfm?TD=0527

CCEVS-TD0536 **NIT Technical Decision for Update Verification Inconsistency**
Date 2020-07-13
Location https://www.niap-ccevs.org/Documents_and_Guidance/view_td.cfm?TD=0536

CCEVS-TD0537 **NIT Technical Decision for Incorrect reference to FCS_TLSC_EXT.2.3**
Date 2020-07-13
Location https://www.niap-ccevs.org/Documents_and_Guidance/view_td.cfm?TD=0537

CCEVS-TD0538 **NIT Technical Decision for Outdated link to allowed-with list**
Date 2020-07-13
Location https://www.niap-ccevs.org/Documents_and_Guidance/view_td.cfm?TD=0538

CCEVS-TD0547 **NIT Technical Decision for Clarification on developer disclosure of AVA_VAN**
Date 2020-10-15
Location https://www.niap-ccevs.org/Documents_and_Guidance/view_td.cfm?TD=0547

CCEVS-TD0555 **NIT Technical Decision for RFC Reference incorrect in TLSS Test**
Date 2020-11-06
Location https://www.niap-ccevs.org/Documents_and_Guidance/view_td.cfm?TD=0555

CCEVS-TD0556 **NIT Technical Decision for RFC 5077 question**
Date 2020-11-06
Location https://www.niap-ccevs.org/Documents_and_Guidance/view_td.cfm?TD=0556

CCEVS-TD0563 **NiT Technical Decision for Clarification of audit date information**
Date 2021-01-28
Location https://www.niap-ccevs.org/Documents_and_Guidance/view_td.cfm?TD=0563

CCEVS-TD0564 **NiT Technical Decision for Vulnerability Analysis Search Criteria**
Date 2021-01-28
Location https://www.niap-ccevs.org/Documents_and_Guidance/view_td.cfm?TD=0564

CCEVS-TD0569 **NIT Technical Decision for Session ID Usage Conflict in FCS_DTLSS_EXT.1.7**
Date 2021-01-28
Location https://www.niap-ccevs.org/Documents_and_Guidance/view_td.cfm?TD=0569

CCEVS-TD0570 **NiT Technical Decision for Clarification about FIA_AFL.1**
Date 2021-01-29
Location https://www.niap-ccevs.org/Documents_and_Guidance/view_td.cfm?TD=0570

CCEVS-TD0571 **NiT Technical Decision for Guidance on how to handle FIA_AFL.1**
Date 2021-01-29
Location https://www.niap-ccevs.org/Documents_and_Guidance/view_td.cfm?TD=0571

CCEVS-TD0572 **NiT Technical Decision for Restricting FTP_ITC.1 to only IP address identifiers**
Date 2021-01-29
Location https://www.niap-ccevs.org/Documents_and_Guidance/view_td.cfm?TD=0572

CCEVS-TD0591 **NIT Technical Decision for Virtual TOEs and hypervisors**
Date 2021-05-21
Location https://www.niap-ccevs.org/Documents_and_Guidance/view_td.cfm?TD=0591

CCEVS-TD0592	**NIT Technical Decision for Local Storage of Audit Records**	
	Date	2021-05-21
	Location	https://www.niap-ccevs.org/Documents_and_Guidance/view_td.cfm?TD=0592
CCEVS-TD0631	**NIT Technical Decision for Clarification of public key authentication for SSH Server**	
	Date	2022-03-21
	Location	https://www.niap-ccevs.org/Documents_and_Guidance/view_td.cfm?TD=0631
CCEVS-TD0632	**NIT Technical Decision for Consistency with Time Data for vNDs**	
	Date	2022-03-21
	Location	https://www.niap-ccevs.org/Documents_and_Guidance/view_td.cfm?TD=0632
CCEVS-TD0634	**NIT Technical Decision for Clarification required for testing IPv6**	
	Date	2022-03-21
	Location	https://www.niap-ccevs.org/Documents_and_Guidance/view_td.cfm?TD=0634
CCEVS-TD0635	**NIT Technical Decision for TLS Server and Key Agreement Parameters**	
	Date	2022-03-21
	Location	https://www.niap-ccevs.org/Documents_and_Guidance/view_td.cfm?TD=0635
CCEVS-TD0638	**TD0638: NIT Technical Decision for Key Pair Generation for Authentication**	
	Date	2022-08-05
	Location	https://www.niap-ccevs.org/Documents_and_Guidance/view_td.cfm?TD=0638
CCEVS-TD0670	**TD0670: NIT Technical Decision for Mutual and Non-Mutual Auth TLSC Testing**	
	Date	2022-09-16
	Location	https://www.niap-ccevs.org/Documents_and_Guidance/view_td.cfm?TD=0670
CEM	**Common Methodology for Information Technology Security Evaluation**	
	Version	3.1R5
	Date	April 2017
	Location	http://www.commoncriteriaportal.org/files/ccfiles/CEMV3.1R5.pdf
CLUSTERADM	**BIG-IP Device Service Clustering: Administration**	
	Version	MAN-0375-12
	Date received	2022-08-25
	File name	agd/BIG-IP Device Service Clustering Administration.pdf
CSEC-EP002	**Evaluation and Certification**	
	Version	34.0
	Date	2021-10-26
	Location	https://www.fmv.se/globalassets/csec/dokumentbibliotek/ep-002.pdf

CSEC-EP188 **Scheme Crypto Policy**
Version 12.0
Date 2021-10-26
Location https://www.fmv.se/globalassets/csec/dokumentbibliotek/ep-188.pdf

ECG **BIG-IP Common Criteria Evaluation Configuration Guide BIG-IP Release 16.1.3.1**
Version 6.13
Date 2022-11-07
File name agd/AGD v6.13.pdf

ESSEN **BIG-IP System: Essentials**
Version 16.0
Date received 2022-08-25
File name agd/BIG-IP System Essentials.pdf

GSG **BIG-IP Systems: Getting Started Guide**
Author(s) F5 Networks, Inc.
Version 10.1
Date 2010-02-04
File name agd/BIG-IP_Systems__Getting_Started_Guide.pdf

HyperVSETUP **Microsoft Hyper-V: BIG-IP VE Setup**
Date 2020-10-21
File name agd/Microsoft Hyper-V_ BIG-IP VE Setup.pdf

HyperVUG **Microsoft Hyper-V: BIG-IP VE User's Guide**
Date 2019-09-06
File name agd/Microsoft Hyper-V_ BIG-IP VE Users Guide.pdf

ICONTROL **iControl SDK 16.1.3.1**
Date received 2022-08-25
File name agd/iControl-16.1.3.1.zip

ICREST **iControl REST API User Guide**
Version 15.1.0
Date received 2022-08-25
File name agd/icontrol-rest-api-user-guide-15-1-0.pdf

K13092 **K13092: Overview of securing access to the BIG-IP system**
Date Aug 26, 2020
File name agd/Article_ K13092 - Overview of securing access to the BIG-IP system.pdf

K13302 **K13302: Configuring the BIG-IP system to use an SSL chain certificate (11.x - 16.x)**
Date Sep 17,2020
File name agd/Article_ K13302 - Configuring the BIG-IP system to use an SSL chain certificate (11.x - 16.x).pdf

K13454 **K13454: Configuring SSH public key authentication on BIG-IP systems (11.x - 16.x)**
Date Apr 20, 2020
File name agd/Article_ K13454 - Configuring SSH public key authentication on BIG-IP systems (11.x - 16.x).pdf

K14620 **K14620: Manage SSL certificates for BIG-IP systems using the Configuration utility**
Date Jul 15, 2020
File name agd/Article_ K14620 - Manage SSL certificates for BIG-IP systems using the Configuration utility.pdf

K14783 **K14783: Overview of the Client SSL profile (11.x - 17.x)**
Date Sep 28, 2020
File name agd/Article_ K14783 - Overview of the Client SSL profile (11.x - 17.x).pdf

K14806 **K14806: Overview of the Server SSL profile (11.x - 17.x)**
Date Jul 07, 2020
File name agd/Article_ K14806 - Overview of the Server SSL profile (11.x - 17.x).pdf

K14810 **K14810: Overview of BIG-IP VE license and throughput limits**
Date Jul 01, 2020
File name agd/Article_ K14810 - Overview of BIG-IP VE license and throughput limits.pdf

K14946 **K14946: Overview of BIG-IP VE image sizes**
Date Aug 24, 2020
File name agd/Article_ K14946 - Overview of BIG-IP VE image sizes.pdf

K15462 **K15462: Managing SSL certificates for BIG-IP systems using tmsh**
Date 2020-10-09
File name agd/Article_ K15462 - Managing SSL certificates for BIG-IP systems using tmsh.pdf

K15497 **K15497: Configuring a secure password policy for the BIG-IP system (11.x - 16.x)**
Date Sep 15, 2020
File name agd/Article_ K15497 - Configuring a secure password policy for the BIG-IP system (11.x - 16.x).pdf

K15664 **K15664: Overview of BIG-IP device certificates (11.x - 16.x)**
Date Apr 23, 2020
File name agd/Article_ K15664 - Overview of BIG-IP device certificates (11.x - 16.x).pdf

K42531434 **K42531434: Replacing the Configuration utility's self-signed device certificate with a CA-signed device certificate.**
Date Apr 23, 2020
File name agd/Article_ K42531434 - Replacing the Configuration utilitys self-signed device certificate with a CA-signed device certificate..pdf

K48615077 **K48615077: BIG-IP daemons (15.x - 16.x)**
Date 2020-11-05
File name agd/Article_ K48615077 - BIG-IP daemons (15.x - 16.x).pdf

K6068 **K6068: Configuring a pre-login or post-login message banner for the BIG-IP Enterprise Manager system**
Date Feb 05, 2018
File name agd/Article_ K6068 - Configuring a pre-login or post-login message banner for the BIG-IP or Enterprise Manager system.pdf

K80425458 **K80425458: Modifying the list of ciphers and MAC aand key exchange algorithms used by the SSH service on the BIG-IP or BIG-IQ systems**
Date May 21, 2020
File name agd/Article_ K80425458 - Modifying the list of ciphers and MAC aand key exchange algorithms used by the SSH service on the BIG-IP or BIG-IQ systems.pdf

K9908 **K9908: Configuring an automatic logout for idle sessions**
Date Sep 28, 2020
File name agd/Article_ K9908 - Configuring an automatic logout for idle sessions.pdf

KVMCRYPTO **KVM: Configure cryptographic offload for BIG-IP VE with Intel QAT**
Date 2020-10-21
File name agd/KVM_ Configure cryptographic offload for BIG-IP VE with Intel QAT.pdf

KVMSETUP **Linux KVM: BIG-IP VE Setup**
Date 2020-10-21
File name agd/Linux KVM_ BIG-IP VE Setup.pdf

KVMUG **KVM: BIG-IP VE Users Guide**
Date 2019-09-06
File name agd/KVM_ BIG-IP VE Users Guide.pdf

LTMMR **External Monitoring of BIG-IP Systems: Implementations**
Version MAN-0775-00
Date received 2022-08-25
File name agd/External Monitoring of BIG-IP Systems Implementations.pdf

NDcPPv2.2e **collaborative Protection Profile for Network Devices Version 2.2e**
Version 2.2e
Date 2020-03-23
Location https://www.niap-ccevs.org/MMO/PP/CPP_ND_V2.2E.pdf

NDcPPv2.2-SD **Supporting Document - Evaluation Activities for Network Device cPP**
Version 2.2
Date 2019-12-20
Location https://www.niap-ccevs.org/MMO/PP/CPP_ND_V2.2-SD.pdf

NDETP	**F5 BIG IP 16.1.3.1 NDcPP Evaluator Test Plan**	
	Author(s)	atsec information security AB
	Version	1.0
	Date	2022-11-10
	File name	ate/ND_Evaluator_Test_Plan.pdf
PG2200	**Platform Guide: VIPRION 2200**	
	Version	MAN-0493-02
	Date	July 20, 2017
	File name	agd/Platform_Guide__VIPRION_2200.pdf
PG4400	**Platform Guide: VIPRION 4400 Series**	
	Version	MAN-0311-09
	Date	August 17, 2017
	File name	agd/Platform_Guide__VIPRION_4400_Series.pdf
PGi11000	**Platform Guide: i5000/i7000/i10000/i11000 Series**	
	Version	MAN-0633-09
	Date	Apr 8, 2020
	File name	agd/platform-guide-i5000i7000i10000i11000-series.pdf
PGi15000	**Platform Guide: i15000 Series**	
	Version	MAN-0678-00
	Date	May 9, 2018
	File name	agd/platform-guide-i15000series.pdf
PGi20000	**Platform Guide: i2000/i4000 Series**	
	Version	MAN-0640-04
	Date	Dec 10, 2019
	File name	agd/Platform Guide i2000i4000 Series.pdf
SSLADM	**BIG-IP System: SSL Administration**	
	Version	MAN-0527-08
	Date received	2019-12-09
	File name	agd/BIG-IP System SSL Administration.pdf
ST	**F5 BIG-IP 16.1.3.1 including APM Security Target**	
	Version	6.7
	Date	2022-12-20
	File name	ase/F5 BIG-IP APM 16 v6.7.pdf
SWUPDATE	**Update BIG-IP VE**	
	Version	1.0
	Date received	2022-08-25
	File name	agd/Update BIG-IP VE.pdf
TestEvidence	**Tests output, logs and network captures**	
	Author(s)	atsec information security AB
	Date	2022-10-27
	File name	ate/TestEvidence.zip

TMOSI	**BIG-IP TMOS: Implementations**	
	Version	13.0
	Date received	2019-04-29
	File name	agd/BIG-IP_TMOS_Implementations.pdf
TMOSRA	**BIG-IP TMOS: Routing Administration**	
	Version	MAN-0412-13
	Date received	2022-08-25
	File name	agd/BIG-IP TMOS Routing Administration.pdf
TMSH-REFv12	**Traffic Management Shell (tmsh) Reference Guide**	
	Version	12.0
	Date	September 1, 2015
	File name	agd/bigip-tmsh-reference-12-0-0.pdf
TMSH-REFv17	**F5 TMSH Reference - 17.x**	
	Version	17.0
	Date received	2022-08-25
	File name	agd/tmsh_17.0.0.pdf
USRADM	**BIG-IP System: User Account Administration**	
	Version	MAN-0768-00
	Date received	2022-08-25
	File name	agd/BIG-IP System User Account Administration.pdf
VCMPAMA	**vCMP for Appliance Models: Administration**	
	Version	MAN-0491-08
	Date	Mar 5, 2019
	File name	agd/vCMP for Appliance Models Administration.pdf
VCMPVMA	**vCMP for VIPRION Systems: Administration**	
	Version	MAN-0376-13
	Date	Aug 8, 2018
	File name	agd/vcmp-for-viprion-systems-administration-14-0-0.pdf
VEPLATFORMS	**BIG-IP Virtual Edition Supported Platforms**	
	Date	2020-10-21
	File name	agd/BIG-IP VE Supported Platforms.pdf
VMwareSETUP	**VMware ESXi: BIG-IP VE Setup**	
	Date	2019-09-06
	File name	agd/VMware ESXi_ BIG-IP VE Setup.pdf
VMwareUG	**VMware ESXi: BIG-IP VE User's Guide**	
	Date	2020-10-21
	File name	agd/VMware ESXi_ BIG-IP VE Users Guide.pdf

A.2 Glossary

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

Authentication data
Information used to verify the claimed identity of a user.

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

Class
A grouping of CC families that share a common focus.

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

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

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

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

Element
An indivisible statement of security need.

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

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

Evaluation Assurance Level (EAL)
An assurance package, consisting of assurance requirements drawn from CC Part 3, representing a point on the CC predefined assurance scale.

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

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

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

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

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

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

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

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

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

Informal
Expressed in natural language.

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

Operation (on a component of the CC)
Modifying or repeating that component. Allowed operations on components are assignment, iteration, refinement and selection.

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

Operational environment
The environment in which the TOE is operated.

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

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

PP evaluation
Assessment of a PP against defined criteria.

Protection Profile (PP)
An implementation-independent statement of security needs for a TOE type.

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

Refinement
The addition of details to a component.

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

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

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

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

Security Function Policy (SFP)
A set of rules describing specific security behaviour enforced by the TSF and expressible as a set of SFRs.

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

Security Target (ST)
An implementation-dependent statement of security needs for a specific identified TOE.

Seed
Value used to initialize the internal state of an RNG.

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

Semiformal
Expressed in a restricted syntax language with defined semantics.

ST evaluation
Assessment of an ST against defined criteria.

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

Target of Evaluation (TOE)
A set of software, firmware and/or hardware possibly accompanied by guidance.

TOE evaluation
Assessment of a TOE against defined criteria.

TOE resource
Anything useable or consumable in the TOE.

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

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

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

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

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

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

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

User
See external entity

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