Assurance Activities Report
for
Axonius Cybersecurity Asset Management Platform v4.0-f

Version 1.0
28 February 2022

Evaluated By:

Leidos Inc.
https://www.leidos.com/civil/commercial-cyber/product-compliance
Common Criteria Testing Laboratory
6841 Benjamin Franklin Drive
Columbia, MD 21046

Prepared for:
National Information Assurance Partnership
Common Criteria Evaluation and Validation Scheme

The Developer of the TOE:
Axonius Federal Systems LLC
330 Madison Avenue, 39th Floor
New York, NY 10017

The TOE Evaluation was Sponsored by:
Axonius Federal Systems LLC
330 Madison Avenue, 39th Floor
New York, NY 10017

Evaluation Personnel:
Anthony J. Apted
Rutwij Kulkarni
Pascal Patin

Common Criteria Version:

- Common Criteria for Information Technology Security Evaluation Part 1: Introduction and general model, Version 3.1, Revision 5, April 2017.
- Common Criteria for Information Technology Security Evaluation Part 2: Security functional components, Version 3.1, Revision 5, April 2017.
- Common Criteria for Information Technology Security Evaluation Part 3: Security assurance components, Version 3.1, Revision 5, April 2017.

Common Evaluation Methodology Version:

- Common Methodology for Information Technology Security Evaluation, Evaluation Methodology, Version 3.1, Revision 5, April 2017.

Protection Profiles:

- Protection Profile for Application Software, Version 1.3, 1 March 2019
- Functional Package for Transport Layer Security (TLS), Version 1.1, February 12, 2019
- Extended Package for Secure Shell (SSH), Version 1.0, February 19, 2016

Revision History

Version	Date	Description
0.1	25 October 2021	Initial draft
0.2	23 November 2021	Updated for revised ST
1.0	28 February 2022	Final version for check-out

Contents

LIST OF TABLES

NO TABLE OF FIGURES ENTRIES FOUND.

1. Introduction

This document presents the results of performing assurance activities associated with the Axonius Cybersecurity Asset Management Platform v4.0-f evaluation. This report contains sections documenting the performance of evaluation activities associated with each of the Security Functional Requirements (SFRs) and Security Assurance Requirements (SARs) as specified in the following documents:

- *Protection Profile for Application Software*, Version 1.3, 1 March 2019 [PP_APP_v1.3]
- *Functional Package for Transport Layer Security (TLS)*, Version 1.1, February 12, 2019 [PKG_TLS_V1.1]
- *Extended Package for Secure Shell (SSH)*, Version 1.0, February 19, 2016 [PP_SSH_EP_v1.0]

Note that, in accordance with NIAP Policy Letter #5, all cryptography in the TOE for which NIST provides validation testing of FIPS-approved and NIST-recommended cryptographic algorithms and their individual components must be NIST validated. The CCTL will verify that the claimed NIST validation complies with the NIAP-approved PP requirements the TOE claims to satisfy. The CCTL verification of the NIST validation will constitute performance of the associated assurance activity. As such, Test assurance activities associated with functional requirements within the scope of Policy Letter #5 are performed by verification of the relevant CAVP certification and not through performance of any testing as specified in the claimed PP documents.

1.1 Technical Decisions

1.1.1 Technical Decisions against [PP_APP_v1.3]

This subsection lists the Technical Decisions that have been issued by NIAP against [PP_APP_v1.3], along with rationale as to their applicability or otherwise to this evaluation.

TD0416 – Correction to FCS_RBG_EXT.1 Test Activity

This TD has been applied to this evaluation.

TD0427 – Reliable Time Source

This TD has been applied to this evaluation.

TD0434 – Windows Desktop Applications Test

N/A – The TOE is not a Windows Desktop Application.

TD0435 – Alternative to SELinux for FPT_AEX_EXT.1.3

This TD has been applied to this evaluation.

TD0437 – Supported Configuration Mechanism

This TD has been applied to this evaluation.

TD0445 – User Modifiable File Definition

This TD has been applied to this evaluation.

TD0465 – Configuration Storage for .NET Apps

N/A – The TOE is not a Windows .NET application.

TD0495 – FIA_X509_EXT.1.2 Test Clarification

This TD has been applied to this evaluation.

TD0498 – Application Software PP Security Objectives and Requirements Rationale

This TD has been applied to this evaluation.

TD0510 – Obtaining random bytes for iOS/macOS

N/A – TD only affects evaluation for iOS and macOS applications and the TOE is a Linux application.

TD0515 – Use Android APK manifest in test

N/A – TD only affects evaluation for Android applications and the TOE is a Linux application.

TD0519 – Linux symbolic links and FMT_CFG_EXT.1

This TD has been applied to this evaluation.

TD0543 – FMT_MEC_EXT.1 evaluation activity update

N/A – TD only affects evaluation for Windows applications and the TOE is a Linux application.

TD0544 – Alternative testing methods for FPT_AEX_EXT.1.1

This TD has been applied to this evaluation.

TD0548 – Integrity for installation tests in AppSW PP 1.3

This TD has been applied to this evaluation.

TD0554 – iOS/iPadOS/Android AppSW Virus Scan

This TD has been applied to this evaluation.

TD0561 – Signature verification update

This TD has been applied to this evaluation.

TD0582 – PP-Configuration for Application Software and Virtual Private Network (VPN) Clients now allowed

N/A – the ST does not claim the PP-Modules for VPN Client or File Encryption.

TD0598 – Expanded AES Modes in FCS_COP for App PP

This TD has been applied to this evaluation.

TD0600 – Conformance claim sections updated to allow for MOD_VPNC_V2.3

N/A – the ST does not claim the PP-Module for VPN Client.

TD0601 – X.509 SFR Applicability in App PP

This TD has been applied to this evaluation.

1.1.2 Technical Decisions against [PKG_TLS_V1.1]

This subsection lists the Technical Decisions that have been issued by NIAP against [PKG_TLS_V1.1], along with rationale as to their applicability or otherwise to this evaluation.

TD0442 – Updated TLS Ciphersuites for TLS Package

This TD has been applied to this evaluation.

TD0469 – Modification of test activity for FCS_TLSS_EXT.1.1 test 4.1

This TD has been applied to this evaluation.

TD0499 – Testing with pinned certificates

This TD has been applied to this evaluation.

TD0513 – CA Certificate loading

This TD has been applied to this evaluation.

TD0588 – Session Resumption Support in TLS package

This TD has been applied to this evaluation.

1.1.3 Technical Decisions against [PP_SSH_EP_v1.0]

This subsection lists the Technical Decisions that have been issued by NIAP against [PP_SSH_EP_v1.0], along with rationale as to their applicability or otherwise to this evaluation.

TD0240 – FCS_COP.1.1(1) Platform provided crypto for encryption/decryption

This TD has been applied to this evaluation.

TD0331 – SSH Rekey Testig

This TD has been applied to this evaluation.

TD0332 – Support for RSA SHA2 host keys

This TD has been applied to this evaluation.

TD0420 – Conflict in FCS_SSHC_EXT.1.1 and FCS_SSHS_EXT.1.1

This TD has been applied to this evaluation.

TD0446 – Missing selections for SSH

This TD has been applied to this evaluation.

TD0598 – Expanded AES Modes in FCS_COP for App PP

This TD has been applied to this evaluation.

1.2 References

[ST] *Axonius Cybersecurity Asset Management Platform v4.0-f Security Target*, Version 1.0, 3 February 2022

[CCECG] *Axonius Cybersecurity Asset Management Platform v4.0-f Common Criteria Evaluated Configuration Guide (CCECG)*, Version 1.0, 14 January 2022.

2. Security Functional Requirement Assurance Activities

This section describes the assurance activities associated with the SFRs defined in the ST and the results of those activities as performed by the evaluation team. The assurance activities are derived from [PP_APP_v1.3], [PKG_TLS_V1.1], and [PP_SSH_EP_v1.0]. NIAP Technical Decisions have been applied and are identified as appropriate.

2.1 Cryptographic Support (FCS)

2.1.1 Cryptographic Asymmetric Key Generation (FCS_CKM.1(1))

2.1.1.1 TSS Activities

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.

Section 6.2 of [ST] ("Cryptographic Support") states the TOE generates asymmetric keys in support of trusted communications. Specifically, the TOE generates the following keys:

- ECC keys using NIST curves P-384 and P-521, in support of ECDHE key establishment schemes used for TLS communications when the TOE negotiates a TLS_ECDHE_* cipher suite, and to support SSH key exchange
- FFC keys using Diffie-Hellman Group 14 in support of SSH key exchange.

If the application invokes platform-provided functionality for asymmetric key generation, then the evaluator shall examine the TSS to verify that it describes how the key generation functionality is invoked.

The statement of FCS_CKM.1(1) in Section 5.2.1.1 of [ST] ("FCS_CKM.1(1) Cryptographic Asymmetric Key Generation") specifies the TOE implements key generation functionality. As such, this activity is not applicable.

2.1.1.2 Guidance Activities

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 uses defined in this PP.

Section "Setting Up Secure Communications in Axonius Cybersecurity Asset Management Platform v4.0-f" of [CCECG] states no administrator configuration is required to configure the TOE to use the selected key generation schemes and key sizes.

2.1.1.3 Test Activities

If the application implements asymmetric key generation, then the following test activities shall be carried out. **Key Generation for Elliptic Curve Cryptography (ECC)** Performed in accordance with NIAP Policy Letter #5.

Section 6.2 of [ST] ("Cryptographic Support"), Table 5 ("Cryptographic Functions") identifies the CAVP certifications verifying validation for ECC key generation, as follows.

Algorithm	Tested Capabilities	Certificates
ECC schemes using “NIST curves that meet the following: FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Appendix B.4	Curve: P-256, P-384, P-521 Secret Generation Mode: Extra Bits, Testing Candidates Prerequisites: DRBG (C1826)	CAVP #C1826 (ECDSA KeyGen (FIPS186-4))

If the application implements asymmetric key generation, then the following test activities shall be carried out.

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

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

Refer to Test Activities for FCS_CKM.2.

2.1.2 Cryptographic Symmetric Key Generation (FCS_CKM.1(2))

2.1.2.1 TSS Activities

The evaluator shall review the TSS to determine that it describes how the functionality described by FCS_RBG_EXT.1 is invoked.

Section 6.2 of [ST] (“Cryptographic Support”) states the TOE generates 256 bit symmetric keys using the CTR_DRBG (AES) algorithm. The algorithm implementation is provided by version 2.0.16 of the OpenSSL FIPS Object Module, which is wrapped with OpenSSL 1.0.2. The CTR_DRBG (AES) functionality is invoked via the Python Requests library, using the “ssl” library.

If the application is relying on random bit generation from the host platform, the evaluator shall verify the TSS includes the name/manufacturer of the external RBG and describes the function call and parameters used when calling the external DRBG function. If different external RBGs are used for different platforms, the evaluator shall verify the TSS identifies each RBG for each platform. Also, the evaluator shall verify the TSS includes a short description of the vendor's assumption for the amount of entropy seeding the external DRBG. The evaluator uses the description of the RBG functionality in FCS_RBG_EXT or documentation available for the operational environment to determine that the key size being requested is identical to the key size and mode to be used for the encryption/decryption of the user data.

The application does not rely on random bit generation from the host platform, so this activity is not applicable.

2.1.2.2 Guidance Activities

None.

2.1.2.3 Test Activities

None.

2.1.3 Password Conditioning (FCS_CKM.1(3))

2.1.3.1 TSS Activities

Support for PBKDF: The evaluator shall examine the password hierarchy TSS to ensure that the formation of all password based derived keys is described and that the key sizes match that described by the ST author. The evaluator shall check that the TSS describes the method by which the password/passphrase is first encoded and then fed to the SHA algorithm. The settings for the algorithm (padding, blocking, etc.) shall be described, and the evaluator shall verify that these are supported by the selections in this component as well as the selections concerning the hash function itself. The evaluator shall verify that the TSS contains a description of how the output of the hash function is used to form the submask that will be input into the function. For the NIST SP 800-132-based conditioning of the password/passphrase, the required evaluation activities will be performed when doing the evaluation activities for the appropriate requirements (FCS_COP.1.1(4)). No explicit testing of the formation of the submask from the input password is required. FCS_CKM.1.1(3): The ST author shall provide a description in the TSS regarding the salt generation. The evaluator shall confirm that the salt is generated using an RBG described in FCS_RBG_EXT.1.

Section 6.2 of [ST] ("Cryptographic Support") states the TOE uses PBKDF2 (as implemented in the Python hashlib.pbkdf2_hmac function) for the following purposes:

- Derive a key to set up LUKS volume encryption from a password supplied by the user
- Protect Web GUI password credentials.

Section 6.2 of [ST] states the TOE uses HMAC-SHA-512 as the pseudorandom function with 100,000 iterations. It generates random salts using the CTR_DRBG (AES) DRBG provided by OpenSSL, which it invokes using the Python secrets.token_hex function. The Python hashlib.pbkdf2_hmac function accepts the password from which it derives a key in the form of a byte buffer. The evaluator confirmed the settings described in the TSS are supported by the selections made in FCS_CKM.1(3) and FCS_COP.1(4).

2.1.3.2 Guidance Activities

None.

2.1.3.3 Test Activities

None.

2.1.4 Cryptographic Key Establishment (FCS_CKM.2)

2.1.4.1 TSS Activities

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.

Section 6.2 of [ST] ("Cryptographic Support") states the TOE supports the following key establishment schemes:

- ECDHE key establishment schemes used for TLS communications when the TOE negotiates a TLS_ECDHE_* cipher suite and in support of SSH key exchange
- Key establishment scheme using Diffie-Hellman Group 14 in support of SSH key exchange.

2.1.4.2 Guidance Activities

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

Section “Setting Up Secure Communications in Axonius Cybersecurity Asset Management Platform v4.0-f” of [CCECG] states no administrator configuration is required to configure the TOE to use the selected key establishment schemes.

2.1.4.3 Test Activities

SP800-56A Key Establishment Schemes Performed in accordance with NIAP Policy Letter #5.

Section 6.2 of [ST] (“Cryptographic Support”), Table 5 (“Cryptographic Functions”) identifies the CAVP certifications verifying validation for ECDSA-based key establishment, as follows.

Algorithm	Tested Capabilities	Certificates
ECC schemes that meet the following: NIST Special Publication 800-56A, “Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography”.	Function: Domain Parameter Generation, Domain Parameter Validation, Full Public Key Validation, Key Pair Generation, Partial Public Key Validation, Public Key Regeneration Scheme: Ephemeral Unified: KAS Role: Initiator, Responder Shared Secret Computation: Parameter Set: EC: Hash Algorithm: SHA2-256 Curve: P-256 ED: Hash Algorithm: SHA2-384 Curve: P-384 EE: Hash Algorithm: SHA2-512 Curve: P-521	CAVP #C1826 (KAS-ECC Component)

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

Refer to Test Activities associated with FCS_SSHC_EXT.1.6.

2.1.5 Cryptographic Key Generation Services (FCS_CKM_EXT.1)

2.1.5.1 TSS Activities

The evaluator shall inspect the application and its developer documentation to determine if the application needs asymmetric key generation services. If not, the evaluator shall verify the ***generate no asymmetric cryptographic keys*** selection is present in the ST. Otherwise, the evaluation activities shall be performed as stated in the selection-based requirements.

The evaluator examined the application and its associated developer documentation and determined the TOE requires asymmetric key generation services, since it uses cryptographic protocols for communication with external IT entities. Section 5.2.1.5 of [ST] ("FCS_CKM_EXT.1 Cryptographic Key Generation Services") specifies the TOE implements asymmetric key generation. The ST includes the appropriate selection-based requirements.

2.1.5.2 Guidance Activities

None.

2.1.5.3 Test Activities

None.

2.1.6 Cryptographic Operation – Encryption/Decryption (FCS_COP.1(1))

2.1.6.1 TSS Activities

None.

2.1.6.2 Guidance Activities

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

Section "Setting Up Secure Communications in Axonius Cybersecurity Asset Management Platform v4.0-f" of [CCECG] states no administrator configuration is required to configure the TOE to use the required encryption modes and key sizes.

2.1.6.3 Test Activities

Performed in accordance with NIAP Policy Letter #5.

Section 6.2 of [ST] ("Cryptographic Support"), Table 5 ("Cryptographic Functions") identifies the CAVP certifications verifying AES encryption and decryption, as follows.

Algorithm	Tested Capabilities	Certificates
AES-CBC as defined in NIST SP 800-38A	AES-CBC: Direction: Decrypt, Encrypt Key Lengths: 128, 256 (bits)	CAVP #C1826 (AES-CBC)

Algorithm	Tested Capabilities	Certificates
AES-GCM as defined in NIST SP 800-38D	AES-GCM: Direction: Decrypt, Encrypt IV Generation: Internal Key Lengths: 128, 256 (bits)	CAVP #C1826 (AES-GCM)

2.1.7 Cryptographic Operation – Encryption/Decryption (FCS_COP.1(1)/SSH)

2.1.7.1 TSS Activities

The evaluator shall review the TSF of the base PP to verify consistency with the functionality that was claimed by the base PP to ensure that applicable dependencies are met.

The evaluator verified that the claims are consistent with the base PP and the applicable dependencies are met.

Modified in accordance with TD0240.

If perform encryption/decryption services is chosen, the evaluator shall verify that the TSS describes the counter mechanism including rationale that the counter values provided are unique.

Section 6.2 of [ST] (“Cryptographic Support”) states the TOE ensures the uniqueness of counter values by using counter blocks that do not repeat within a given message and by ensuring that initial counter blocks are chosen such that counters are unique across all messages that are encrypted under a given key.

2.1.7.2 Guidance Activities

None.

2.1.7.3 Test Activities

Performed in accordance with NIAP Policy Letter #5.

Section 6.2 of [ST] (“Cryptographic Support”), Table 5 (“Cryptographic Functions”) identifies the CAVP certifications verifying AES encryption and decryption, as follows.

Algorithm	Tested Capabilities	Certificates
AES-CTR as defined in NIST SP 800-38A	AES-CTR: Direction: Encrypt Key Lengths: 128, 256 (bits)	CAVP #C1826 (AES-CTR)

If "invoke platform-provided" is selected, the evaluator confirms that SSH connections are only successful if appropriate algorithms and appropriate key sizes are configured. To do this, for each listening SSH socket connection on the TOE, the evaluator configures an SSH client to connect with an invalid cryptographic algorithm and key-size. The evaluator observes that the connection fails. Likewise, for initiated connection, the evaluator configures a listening SSH socket on the remote server that accepts only invalid cryptographic algorithms and keys. The evaluator observes that the connection fails.

The ST selects “perform”. Therefore, this activity is not applicable to the TOE.

2.1.8 Cryptographic Operation – Hashing (FCS_COP.1(2))

2.1.8.1 TSS Activities

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

Section 6.2 of [ST] ("Cryptographic Support") states the TOE uses the hash function in support of TLS communications and to support the HMAC functions used in PBKDF2 and SSH data integrity MAC functions.

2.1.8.2 Guidance Activities

None.

2.1.8.3 Test Activities

Performed in accordance with NIAP Policy Letter #5.

Section 6.2 of [ST] ("Cryptographic Support"), Table 5 ("Cryptographic Functions") identifies the CAVP certifications verifying cryptographic hashing, as follows.

Algorithm	Tested Capabilities	Certificates
SHS as defined in FIPS Pub 180-4	SHA-1	CAVP #C1826 (SHA-1)
	SHA-256	CAVP #C1826 (SHA2-256)
	SHA-384	CAVP #C1826 (SHA2-384)
	SHA-512	CAVP #C1826 (SHA2-512)

2.1.9 Cryptographic Operation – Signing (FCS_COP.1(3))

2.1.9.1 TSS Activities

None.

2.1.9.2 Guidance Activities

None.

2.1.9.3 Test Activities

Performed in accordance with NIAP Policy Letter #5.

Section 6.2 of [ST] ("Cryptographic Support"), Table 5 ("Cryptographic Functions") identifies the CAVP certifications verifying digital signature generation and verification services, as follows.

Algorithm	Tested Capabilities	Certificates
RSA schemes using cryptographic key sizes of 2048 bits that meet the following: FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Section 4	**RSA SigGen (186-4)** Signature Type: PKCS 1.5 Modulo: 2048 Hash Algorithm: SHA2-256 Hash Algorithm: SHA2-384 Hash Algorithm: SHA2-512	CAVP #C1826 (RSA SigGen (FIPS186-4))
	RSA SigVer (186-4) Signature Type: PKCS 1.5 Modulo: 2048 Hash Algorithm: SHA2-256 Hash Algorithm: SHA2-384 Hash Algorithm: SHA2-512	CAVP #C1826 (RSA SigVer (FIPS186-4))
ECDSA schemes using NIST curves P-256, P-384, and P-521, that meet the following: FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Section 5	**ECDSA SigGen (186-4)** Capabilities: Curve: P-256 Hash Algorithm: SHA2-256, SHA2-384, SHA2-512 Curve: P-384 Hash Algorithm: SHA2-256, SHA2-384, SHA2-512 Curve: P-521 Hash Algorithm: SHA2-256, SHA2-384, SHA2-512	CAVP #C1826 (ECDSA SigGen (FIPS186-4))
	ECDSA SigGen (186-4) Capabilities: Curve: P-256 Hash Algorithm: SHA-1, SHA2-256, SHA2-384, SHA2-512 Curve: P-384 Hash Algorithm: SHA-1, SHA2-256, SHA2-384, SHA2-512 Curve: P-521 Hash Algorithm: SHA-1, SHA2-256, SHA2-384, SHA2-512	CAVP #C1826 (ECDSA SigVer (FIPS186-4))

2.1.10 Cryptographic Operation – Keyed-Hash Message Authentication (FCS_COP.1(4))

2.1.10.1 TSS Activities

None.

2.1.10.2 Guidance Activities

None.

2.1.10.3 Test Activities

Performed in accordance with NIAP Policy Letter #5.

Section 6.2 of [ST] (“Cryptographic Support”), Table 5 (“Cryptographic Functions”) identifies the CAVP certifications verifying keyed-hash message authentication services, as follows.

Algorithm	Tested Capabilities	Certificates
HMAC that meets : FIPS Pub 198-1, "The Keyed-Hash Message Authentication Code”, and FIPS Pub 180-4, “Secure Hash Standard”	HMAC-SHA-1 Key sizes < block size Key sizes > block size Key size = block size HMAC-SHA2-256 Key sizes < block size Key sizes > block size Key size = block size HMAC-SHA2-384 Key sizes < block size Key sizes > block size Key size = block size HMAC-SHA2-512 Key sizes < block size Key sizes > block size Key size = block size	CAVP #C1826: HMAC-SHA-1 HMAC-SHA2-256 HMAC-SHA2-384 HMAC-SHA2-512

2.1.11 HTTPS Protocol (FCS_HTTPS_EXT.1/Client)

This is as specified in TD0601.

2.1.11.1 FCS_HTTPS_EXT.1.1/Client

2.1.11.1.1 TSS Activities

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

Section 6.2 of [ST] (“Cryptographic Support”) states the TOE’s implementation of HTTPS conforms to RFC 2818 by using a dedicated server port for its HTTP over TLS traffic and providing the server's identity in the server's Certificate message. The TOE uses HTTP over TLS client functionality for communications between the TOE and adapter sources in the operational environment. The TLS client will not establish a trusted channel if the server certificate is invalid. There is no administrative override. This communication does not use mutually authenticated TLS.

2.1.11.1.2 Guidance Activities

None.

2.1.11.1.3 Test Activities

The evaluator shall attempt to establish an HTTPS connection with a webserver, observe the traffic with a packet analyzer, and verify that the connection succeeds and that the traffic is identified as TLS or HTTPS.

This test was performed in conjunction with FCS_TLSC_EXT.1.1 Test 1. The connections in that test demonstrated the TOE's ability to connect to a web server and verified that all traffic was encrypted with TLS.

2.1.11.2 FCS_HTTPS_EXT.1.2/Client

2.1.11.2.1 TSS Activities

None.

2.1.11.2.2 Guidance Activities

None.

2.1.11.2.3 Test Activities

Other tests are performed in conjunction with the TLS package.

2.1.11.3 FCS_HTTPS_EXT.1.3/Client

2.1.11.3.1 TSS Activities

None.

2.1.11.3.2 Guidance Activities

None.

2.1.11.3.3 Test Activities

Certificate validity shall be tested in accordance with testing performed for FIA_X509_EXT.1, and the evaluator shall perform the following test:

- **Test 1**: The evaluator shall demonstrate that using a certificate without a valid certification path results in the selected action in the SFR. If "notify the user" is selected in the SFR, then the evaluator shall also determine that the user is notified of the certificate validation failure. Using the administrative guidance, the evaluator shall then load a certificate or certificates to the Trust Anchor Database needed to validate the certificate to be used in the function, and demonstrate that the function succeeds. The evaluator then shall delete one of the certificates, and show that again, using a certificate without a valid certification path results in the selected action in the SFR, and if "notify the user" was selected in the SFR, the user is notified of the validation failure.

This test was performed in conjunction with FIA_X509_EXT.1.1.1 Test 1. The two final parts of that test required the evaluator to show that the TOE would not open a connection to a TLS server whose certificate did not have a valid certification path. The evaluator also loaded a certificate into the TOE trust store showing that the connection would succeed with a valid certification path.

2.1.12 HTTPS Protocol (FCS_HTTPS_EXT.1/Server)

This is as specified in TD0601.

2.1.12.1 FCS_HTTPS_EXT.1.1/Server

2.1.12.1.1 TSS Activities

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

Section 6.2 of [ST] ("Cryptographic Support") states the TOE's implementation of HTTPS conforms to RFC 2818 by using a dedicated server port for its HTTP over TLS traffic and providing the server's identity in the server's Certificate message.

2.1.12.1.2 Guidance Activities

None.

2.1.12.1.3 Test Activities

The evaluator shall attempt to establish an HTTPS connection to the TOE using a client, observe the traffic with a packet analyzer, and verify that the connection succeeds and that the traffic is identified as TLS or HTTPS.

This test was performed in conjunction with FCS_TLSS_EXT.1.1. The connections in that test demonstrated the TOE's ability to connect to a web client and verified that all traffic was encrypted with TLS.

2.1.12.2 FCS_HTTPS_EXT.1.2/Server

2.1.12.2.1 TSS Activities

None.

2.1.12.2.2 Guidance Activities

None.

2.1.12.2.3 Test Activities

Other tests are performed in conjunction with the TLS package.

2.1.13 Random Bit Generation Services (FCS_RBG_EXT.1)

2.1.13.1 TSS Activities

If ***use no DRBG functionality*** is selected, the evaluator shall inspect the application and its developer documentation and verify that the application needs no random bit generation services.

If ***implement DRBG functionality*** is selected, the evaluator shall ensure that additional FCS_RBG_EXT.2 elements are included in the ST.

If ***invoke platform-provided DRBG functionality*** is selected, the evaluator performs the following activities. The evaluator shall examine the TSS to confirm that it identifies all functions (as described by the SFRs included in the ST) that obtain random numbers from the platform RBG. The evaluator shall determine that for each of these functions, the TSS states which platform interface (API) is used to obtain the random numbers. The evaluator shall confirm that each of these interfaces corresponds to the acceptable interfaces listed for each platform below.

It should be noted that there is no expectation that the evaluators attempt to confirm that the APIs are being used "correctly" for the functions identified in the TSS; the activity is to list the used APIs and then do an existence check via decompilation.

In FCS_RBG_EXT.1, the ST author has selected ***implement DRBG functionality***. The evaluator confirmed the ST includes FCS_RBG_EXT.2.

2.1.13.2 Guidance Activities

None defined.

2.1.13.3 Test Activities

Modified in accordance with TD0416.

If ***invoke platform-provided DRBG functionality*** is selected, the following tests shall be performed:

The evaluator shall decompile the application binary using a decompiler suitable for the application (TOE). The evaluator shall search the output of the decompiler to determine that, for each API listed in the TSS, that API appears in the output. If the representation of the API does not correspond directly to the strings in the following list, the evaluator shall provide a mapping from the decompiled text to its corresponding API, with a description of why the API text does not directly correspond to the decompiled text and justification that the decompiled text corresponds to the associated API.

The following are the per-platform list of acceptable APIs:

For Linux: The evaluator shall verify that the application collects random from `/dev/random` or `/dev/urandom`.

In FCS_RBG_EXT.1, the ST author has selected ***implement DRBG functionality***.

2.1.14 Random Bit Generation from Application (FCS_RBG_EXT.2)

2.1.14.1 FCS_RBG_EXT.2.1

2.1.14.1.1 TSS Activities

None.

2.1.14.1.2 Guidance Activities

None.

2.1.14.1.3 Test Activities

Performed in accordance with NIAP Policy Letter #5.

Section 6.2 of [ST] ("Cryptographic Support"), Table 5 ("Cryptographic Functions") identifies the CAVP certifications verifying deterministic random bit generation services, as follows.

Algorithm	Tested Capabilities	Certificates
CTR_DRBG in accordance with NIST Special Publication 800-90A	Counter DRBG Mode: AES-256	CAVP #C1826: Counter DRBG

2.1.14.2 FCS_RBG_EXT.2.2

2.1.14.2.1 TSS Activities

Documentation shall be produced – and the evaluator shall perform the activities – in accordance with Appendix D - Entropy Documentation and Assessment and the Clarification to the Entropy Documentation and Assessment Annex.

The vendor provided documentation describing the entropy sources used by the TOE.

2.1.14.2.2 Guidance Activities

None.

2.1.14.2.3 Test Activities

In the future, specific statistical testing (in line with NIST SP 800-90B) will be required to verify the entropy estimates.

The proprietary Entropy Analysis Report (EAR) describes how the TSF extracts random data from software-based sources to ensure that an amount of entropy that is at least equal to the strength of the generated keys is present (i.e., at least 256 bits when the largest supported keys are generated) when seeding the DRBG for key generation purposes. The TOE relies on entropy sources provided by its platform. The TOE collects entropy from the non-blocking pool via Python calls that invoke the Linux `getrandom()` system call.

2.1.15 SSH Protocol (FCS_SSH_EXT.1)

2.1.15.1 TSS Activities

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

This is covered in the Evaluation Activities for FCS_SSHC_EXT.1 in Section 2.1.16.

2.1.15.2 Guidance Activities

None.

2.1.15.3 Test Activities

None.

2.1.16 SSH Protocol – Client (FCS_SSHC_EXT.1)

2.1.16.1 FCS_SSHC_EXT.1.1

2.1.16.1.1 TSS Activities

Modified in accordance with TD0420.

The evaluator will check to ensure that the TSS contains a description of the public key algorithms that are acceptable for use for authentication, that this list conforms to FCS_SSHC_EXT.1.4, and ensure that password-based authentication methods ~~are also allowed~~, if supported, are described.

Section 6.2 of [ST] (“Cryptographic Support”) states the TOE supports public key and password authentication methods as described in RFC 4252. Section 6.2 of [ST] further identifies the TOE supports ecdsa-sha2-nistp256 and ecdsa-sha2-nistp384 as its public key algorithms. This is consistent with the selections made in FCS_SSHC_EXT.1.4.

2.1.16.1.2 Guidance Activities

None.

2.1.16.1.3 Test Activities

Test 1: The evaluator will, for each public key algorithm supported, show that the TOE supports the use of that public key algorithm to authenticate a user connection to an SSH server. Any configuration activities required to support this test shall be performed according to instructions in the guidance documentation.

The evaluator configured the TOE in accordance with the guidance documentation and attempted to establish a connection to an external SSH server using each of the supported public key algorithms in turn. For each supported algorithm, the TOE was able to authenticate itself to the SSH server and establish the SSH connection.

Modified in accordance with TD0420.

Test 2 [conditional]: Using the guidance documentation, the evaluator will configure the TOE to perform password-based authentication to an SSH server, and demonstrate that a user can be successfully authenticated by the TOE to an SSH server using a password as an authenticator.

The evaluator configured the TOE in accordance with the guidance documentation to perform password-based authentication and attempted to establish a connection to an external SSH server. The TOE was able to authenticate the user to the SSH server using the password as an authenticator.

2.1.16.2 FCS_SSHC_EXT.1.2

2.1.16.2.1 TSS Activities

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

Section 6.2 of [ST] states the TOE’s SSH client keeps track of SSH packet sizes and ensures packets greater than 32 kilobytes in an SSH transport connection are dropped.

2.1.16.2.2 Guidance Activities

None.

2.1.16.2.3 Test Activities

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

The evaluator established a connection from the TOE to the SSH server and send attempted to send a packet larger than 32 kilobytes from the SSH server to the TOE. The evaluator observed the TOE dropped the large packet.

2.1.16.3 FCS_SSHC_EXT.1.3

2.1.16.3.1 TSS Activities

The evaluator will 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 will check the TSS to ensure that the encryption algorithms specified are identical to those listed for this component.

Section 6.2 of [ST] states the TOE's SSH transport implementation uses the aes128-ctr, aes256-ctr, aes128-cbc, and aes256-cbc encryption algorithms. This list is identical to the algorithms specified in FCS_SSHC_EXT.1.3. The TSS does not specify any optional SSH characteristics.

2.1.16.3.2 Guidance Activities

The evaluator will 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).

Section "Setting Up Secure Communications in Axonius Cybersecurity Asset Management Platform v4.0-f" of [CCECG] states no administrator configuration is required to configure the TOE so that SSH conforms to the description in the TSS. It states the SSH client implements only the following encryption algorithms: aes128-ctr; aes256-ctr; aes128-cbc; and aes256-cbc.

2.1.16.3.3 Test Activities

Test 1: The evaluator will establish an SSH connection using each of the encryption algorithms 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.

The evaluator attempted to establish a connection from the TOE to the SSH server using each of the supported encryption algorithms in turn. The evaluator observed through examining packet captures the successful negotiation of each supported encryption algorithm.

Test 2: The evaluator will configure an SSH server to only allow the 3des-cbc encryption algorithm and no other encryption algorithms. The evaluator will attempt to establish an SSH connection from the TOE to the SSH server and observe that the connection is rejected.

The evaluator configured the SSH server to accept only 3des-cbc as the encryption algorithm and attempted to establish a connection from the TOE to the SSH server. The evaluator observed the TOE rejected the connection.

2.1.16.4 FCS_SSHC_EXT.1.4

2.1.16.4.1 TSS Activities

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

Section 6.2 of [ST] states the TOE's SSH transport implementation uses the ecdsa-sha2-nistp256 and ecdsa-sha2-nistp384 public key algorithms. This list is identical to the algorithms specified in FCS_SSHC_EXT.1.4. The TSS does not specify any optional SSH characteristics.

2.1.16.4.2 Guidance Activities

The evaluator will 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).

Section "Setting Up Secure Communications in Axonius Cybersecurity Asset Management Platform v4.0-f" of [CCECG] states no administrator configuration is required to configure the TOE so that SSH conforms to the description in the TSS. It states the SSH client implements only the following public key algorithms: ecdsa-sha2-nistp256; and ecdsa-sha2-nistp384.

2.1.16.4.3 Test Activities

Test 1: The evaluator will establish a SSH connection using each of the public key algorithms specified by the requirement to authenticate an SSH server to the TOE. It is sufficient to observe (on the wire) the successful negotiation of the algorithm to satisfy the intent of the test.

This test was performed as part of the testing for FCS_SSHC_EXT.1.1.

Test 2: The evaluator will configure an SSH server to only allow the ssh-dsa public key algorithm and no other public key algorithms. The evaluator will attempt to establish an SSH connection from the TOE to the SSH server and observe that the connection is rejected.

The evaluator configured the SSH server to accept only ssh-dsa as the public key algorithm and attempted to establish a connection from the TOE to the SSH server. The evaluator observed the TOE rejected the connection.

2.1.16.5 FCS_SSHC_EXT.1.5

2.1.16.5.1 TSS Activities

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

Section 6.2 of [ST] states the TOE's SSH transport implementation uses hmac-sha1, hmac-sha1-96, hmac-sha2-256, and hmac-sha2-512 as its data integrity MAC algorithms. This list is identical to the algorithms specified in FCS_SSHC_EXT.1.5.

2.1.16.5.2 Guidance Activities

The evaluator will also check the guidance documentation to ensure that it contains instructions to the 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).

Section "Setting Up Secure Communications in Axonius Cybersecurity Asset Management Platform v4.0-f" of [CCECG] states no administrator configuration is required to configure the TOE so that SSH conforms to the description in the TSS. It states the SSH client implements only the following data integrity MAC algorithms: hmac-sha1; hmac-sha1-96; hmac-sha2-256; and hmac-sha2-512. It explicitly states the "none" MAC algorithm is not allowed.

2.1.16.5.3 Test Activities

Modified in accordance with TD0446.

Test 1: The evaluator will establish a SSH connection using each of the integrity 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.

The evaluator attempted to establish a connection from the TOE to the SSH server using each of the supported data integrity algorithms in turn. The evaluator observed through examining packet captures the successful negotiation of each supported data integrity algorithm.

Test 2: The evaluator will configure an SSH server to only allow the "none" MAC algorithm. The evaluator will attempt to connect from the TOE to the SSH server 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.

The evaluator configured the SSH server to allow only the "none" MAC algorithm and attempted to establish a connection from the TOE to the SSH server. The evaluator observed the TOE rejected the connection.

Test 3: The evaluator will configure an SSH server to only allow the hmac-md5 MAC algorithm. The evaluator will attempt to connect from the TOE to the SSH server and observe that the attempt fails.

The evaluator configured the SSH server to allow only the hmac-md5 MAC algorithm and attempted to establish a connection from the TOE to the SSH server. The evaluator observed the TOE rejected the connection.

2.1.16.6 FCS_SSHC_EXT.1.6

2.1.16.6.1 TSS Activities

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

Section 6.2 of [ST] states the TOE's SSH transport implementation uses the diffiehellman-group14-sha1, ecdh-sha2-nistp256, ecdh-sha2-nistp384, and ecdh-sha2-nistp521 key exchange methods. This list is identical to the methods specified in FCS_SSHC_EXT.1.6.

2.1.16.6.2 Guidance Activities

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

Section "Setting Up Secure Communications in Axonius Cybersecurity Asset Management Platform v4.0-f" of [CCECG] states no administrator configuration is required to configure the TOE so that SSH conforms to the description in the TSS. It states the SSH client implements only the following key exchange methods: diffiehellman-group14-sha1; ecdh-sha2-nistp256; ecdh-sha2-nistp384; and ecdh-sha2-nistp521.

2.1.16.6.3 Test Activities

Test 1: The evaluator will configure an SSH server to permit all allowed key exchange methods. The evaluator will attempt to connect from the TOE to the SSH server using each allowed key exchange method, and observe that each attempt succeeds.

The evaluator attempted to establish a connection from the TOE to the SSH server using each of the supported key exchange methods in turn. The evaluator observed a successful connection attempt using each supported key exchange method.

2.1.16.7 FCS_SSHC_EXT.1.7

2.1.16.7.1 TSS Activities

None.

2.1.16.7.2 Guidance Activities

None.

2.1.16.7.3 Test Activities

Modified in accordance with TD0331.

Test 1: The evaluator will configure the TOE to create a log entry when a rekey occurs. The evaluator will connect to the TOE with an SSH client and cause a rekey to occur according to the selection(s) in the ST, and subsequently the evaluator uses available methods and tools to verify that rekeying occurs. This could be done, e.g., by checking that a corresponding audit event has been generated by the TOE, if the TOE supports auditing of rekey events.

Due to the nature of how the TOE uses SSH as a client it should never hit a rekey threshold. The TOE is an asset management platform which opens SSH connections in order to collect information from other products. A connection should not last more than a few seconds and will only transmit a few kilobytes of data.

Sample SSH connections were provided to the validation team along with an example of the type of data that is transmitted over SSHC. Based on this it was determined that this requirement is met indirectly since there is no realistic way for the TOE to hit a reasonable rekey threshold.

2.1.16.8 FCS_SSHC_EXT.1.8

2.1.16.8.1 TSS Activities

None.

2.1.16.8.2 Guidance Activities

None.

2.1.16.8.3 Test Activities

Test 1: The evaluator will delete all entries in the TOE's list of recognized SSH server host keys and, if selected, all entries in the TOE's list of trusted certification authorities. The evaluator will initiate a connection from the TOE to an SSH server. The evaluator shall ensure that the TOE either rejects the connection or displays the SSH server's public key (either the key bytes themselves or a hash of the key using any allowed hash algorithm) and prompts the user to accept or deny the key before continuing the connection.

The evaluator deleted the SSH server host key fingerprint from the TOE's list of recognized host keys and then attempted to initiate a connection from the TOE to the SSH server. The evaluator observed the TOE rejected the attempt to establish an SSH connection.

Test 2: The evaluator will add an entry associating a host name with a public key into the TOE's local database. The evaluator will replace, on the corresponding SSH server, the server's host key with a different host key. The evaluator will initiate a connection from the TOE to the SSH server using password-based authentication, shall ensure that the TOE rejects the connection, and shall ensure that the password was not transmitted to the SSH server (for example, by instrumenting the SSH server with a debugging capability to output received passwords).

The evaluator added an entry associating a host name into the TOE's local database and then replaced the server's host key on the corresponding SSH server with a different host key. The evaluator attempted to connect the TOE to the SSH server using password-based authentication and observed the TOE rejected the connection attempt. The evaluator examined the SSH server's logs and confirmed the password was not transmitted.

2.1.17 Storage of Credentials (FCS_STO_EXT.1)

2.1.17.1 TSS Activities

The evaluator shall check the TSS to ensure that it lists all persistent credentials (secret keys, PKI private keys, or passwords) needed to meet the requirements in the ST. For each of these items, the evaluator shall confirm that the TSS lists for what purpose it is used, and how it is stored.

Section 6.2 of [ST] ("Cryptographic Support") states the TOE stores credentials for adapters in the MongoDB encrypted using 256 bit AES in CBC mode. It also states the TOE uses the PBKDF2 algorithm to protect Web GUI password credentials and private keys.

2.1.17.2 Guidance Activities

None defined.

2.1.17.3 Test Activities

For all credentials for which the application implements functionality, the evaluator shall verify credentials are encrypted according to FCS_COP.1(1) or conditioned according to FCS_CKM.1.1(1) and FCS_CKM.1(3).

The evaluator confirmed the ST includes FCS_COP.1(1), which specifies the encryption functionality used to protect adapter credentials, and FCS_CKM.1(3), which specifies the PBKDF2 algorithm used to protect Web GUI password credentials and private keys.

2.1.18 TLS Protocol (FCS_TLS_EXT.1)

2.1.18.1 TSS Activities

None.

2.1.18.2 Guidance Activities

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

In Section 5.2.1.18 of [ST] ("FCS_TLS_EXT.1 TLS Protocol (TLS EP)"), "TLS as a client" and "TLS as a server" are selected in FCS_TLS_EXT.1.1. The ST includes FCS_TLSS_EXT.1 and FCS_TLSC_EXT.1 from the selection-based requirements in [PKG_TLS_V1.1], consistent with the selections made in FCS_TLS_EXT.1.1.

2.1.18.3 Test Activities

None.

2.1.19 TLS Client Protocol (FCS_TLSC_EXT.1) [FPTLS]

2.1.19.1 FCS_TLSC_EXT.1.1

2.1.19.1.1 TSS Activities

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

Section 6.2 of [ST] ("Cryptographic Support") states the TOE's TLS client implementation supports the following TLS cipher suites in the TOE's evaluated configuration:

- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289.

This list comprises exactly the list of supported cipher suites specified in the SFR.

2.1.19.1.2 Guidance Activities

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

Section "Setting Up Secure Communications in Axonius Cybersecurity Asset Management Platform v4.0-f" of [CCECG] states no administrator configuration is required to configure the TOE so that TLS conforms to the description in the TSS.

2.1.19.1.3 Test Activities

The evaluator shall also perform the following tests:

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

The evaluator established a connection from the TOE to a TLS test server using each of the cipher suites claimed by the TOE. A wire capture was made of each connection to verify that the connection was completed using the claimed cipher suite.

Test 2: The goal of the following test is to verify that the TOE accepts only certificates with appropriate values in the extendedKeyUsage extension, and implicitly that the TOE correctly parses the extendedKeyUsage extension as part of X.509v3 server certificate validation.

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

The evaluator attempted a TLS connection from the TOE to a TLS test server with a certificate that did not contain the Server Authentication purpose in the extendedKeyUsage extension and observed the attempt failed. This connection attempt was then repeated to a server whose certificate did have the Server Authentication purpose in the extendedKeyUsage extension and observed to succeed. Wire captures were made of both connections to verify that the first was not successful while the second was.

Test 3: The evaluator shall send a server certificate in the TLS connection that does not match the server-selected cipher suite (for example, send a ECDSA certificate while using the TLS_RSA_WITH_AES_128_CBC_SHA cipher suite or send a RSA certificate while using one of the ECDSA cipher suites.) The evaluator shall verify that the product disconnects after receiving the server's Certificate handshake message.

The evaluator attempted to open a connection from the TOE to a TLS test server. The TLS test server selected a ciphersuite which uses an ECDSA cipher but then transmitted an RSA certificate. After receiving this certificate the TOE rejected the connection with a Bad Certificate error.

Test 4: The evaluator shall configure the server to select the TLS_NULL_WITH_NULL_NULL cipher suite and verify that the client denies the connection.

The evaluator attempted to establish a connection from the TOE to a TLS test server which used the TLS_NULL_WITH_NULL_NULL cipher suite. The TOE terminated the connection attempt after receiving the server's Server Hello packet.

Test 5: The evaluator shall perform the following modifications to the traffic:

Test 5.1: Change the TLS version selected by the server in the Server Hello to an undefined TLS version (for example 1.5 represented by the two bytes 03 06) and verify that the client rejects the connection.

The evaluator attempted to establish a connection from the TOE to a TLS test server which had a TLS version of 03 06 in its Server Hello packet. The TOE terminated the connection attempt after receiving the Server Hello packet.

Test 5.2: Change the TLS version selected by the server in the Server Hello to the most recent unsupported TLS version (for example 1.1 represented by the two bytes 03 02) and verify that the client rejects the connection.

The evaluator attempted to establish a connection from the TOE to a TLS test server which had a TLS version of 03 02 (for TLS 1.1) in its Server Hello packet. The TOE terminated the connection attempt after receiving the Server Hello packet.

Test 5.3: [conditional] If DHE or ECDHE cipher suites are supported, modify at least one byte in the server's nonce in the Server Hello handshake message, and verify that the client does not complete the handshake and no application data flows.

The evaluator attempted to establish a connection from the TOE to a TLS test server which had a byte modified in the server's nonce in the Server Hello packet. The TOE terminated the connection attempt after receiving the Server Hello packet.

Test 5.4: Modify the server's selected cipher suite in the Server Hello handshake message to be a cipher suite not presented in the Client Hello handshake message. The evaluator shall verify that the client does not complete the handshake and no application data flows.

The evaluator attempted to establish a connection from the TOE to a TLS test server which had the selected cipher suite the Server Hello packet modified to be one not presented in the Client Hello message. The TOE terminated the connection attempt after receiving the Server Hello packet.

Test 5.5: [conditional] If DHE or ECDHE cipher suites are supported, modify the signature block in the server's Key Exchange handshake message, and verify that the client does not complete the handshake and no application data flows. This test does not apply to cipher suites using RSA key exchange. If a TOE only supports RSA key exchange in conjunction with TLS, then this test shall be omitted.

The evaluator attempted to establish a connection from the TOE to a TLS test server which had the signature block in the server's Key Exchange message modified. The TOE did not complete the handshake and the connection was not established.

Test 5.6: Modify a byte in the Server Finished handshake message, and verify that the client does not complete the handshake and no application data flows.

The evaluator attempted to establish a connection from the TOE to a TLS test server which had a byte modified in the Server Finished message. The TOE terminated the connection with an "Encrypted Alert" message.

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

The evaluator attempted to establish a connection from the TOE to a TLS test server which sent a message with random bytes following the Change Cipher Spec message. The TOE terminated the connection with an "Encrypted Alert" message.

2.1.19.2 FCS_TLSC_EXT.1.2

2.1.19.2.1 TSS Activities

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

Section 6.2 of [ST] ("Cryptographic Support") states the TOE uses TLS client functionality for communications between the TOE and adapter sources in the operational environment. The TOE validates the reference identifier of a presented server certificate as part of certificate validation in the establishment of TLS connectivity. This is done through validation of the Common Name (CN) and Subject Alternative Name (SAN) certificate fields. The SAN field is expected to contain the Fully Qualified Domain Name (FQDN) of the system to which the TOE is attempting to connect (i.e., the adapter source). The reference identifier is established by configuration. The TOE supports IP addresses and wildcards, but does not support certificate pinning.

2.1.19.2.2 Guidance Activities

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

Section 6.4 of [ST] ("Identification and Authentication") states the TOE uses X.509 certificates when the TOE is providing its own TLS server certificate to a TLS client and for authenticating the asset's TLS server certificate presented to it.

The guidance provided by [CCECG] includes instructions to the administrator for setting the reference identifier to be used for TLS certificate validation. Refer to "Configuring the Adapter Connection", which provides instructions for configuring the reference identifier when configuring an adapter that the TOE will communicate with over TLS/HTTPS (where the TOE acts as the TLS client).

2.1.19.2.3 Test Activities

Modified in accordance with TD0499.

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

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

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

The evaluator configured the TLS server to present a certificate with a CN not matching the configured reference identifier and without a SAN extension. The evaluator confirmed the connection attempt failed.

Test 2: The evaluator shall present a server certificate that contains a CN that matches the reference identifier, contains the SAN extension, but does not contain an identifier in the SAN that matches the reference identifier. The evaluator shall verify that the connection fails. The evaluator shall repeat this test for each supported SAN type.

The evaluator configured the TLS server to present a certificate with a CN matching the configured reference identifier and with a SAN extension not matching the configured reference identifier. The evaluator confirmed the connection attempt failed. This test was performed twice, the first time using a bad IP address, and the second time using a bad FQDN.

Test 3: [conditional] If the TOE does not mandate the presence of the SAN extension, the evaluator shall present a server certificate that contains a CN that matches the reference identifier and does not contain the SAN extension. The evaluator shall verify that the connection succeeds. If the TOE does mandate the presence of the SAN extension, this Test shall be omitted.

The evaluator configured the TLS server to present a certificate with a CN matching the configured reference identifier and without a SAN extension. The evaluator confirmed the connection attempt succeeded.

Test 4: The evaluator shall present a server certificate that contains a CN that does not match the reference identifier but does contain an identifier in the SAN that matches. The evaluator shall verify that the connection succeeds.

The evaluator configured the TLS server to present a certificate with a CN not matching the configured reference identifier but with a SAN extension matching the configured reference identifier. The evaluator confirmed the connection attempt succeeded.

Test 5: The evaluator shall perform the following wildcard tests with each supported type of reference identifier. The support for wildcards is intended to be optional. If wildcards are supported, the first, second, and third tests below shall be executed. If wildcards are not supported, then the fourth test below shall be executed.

Test 5.1: [conditional]: If wildcards are supported, the evaluator shall present a server certificate containing a wildcard that is not in the left-most label of the presented identifier (e.g. foo.*.example.com) and verify that the connection fails.

The evaluator configured the TLS server to present a certificate with a CN with a wildcard that was in the second position from the left (test.*.leidos.ate). The evaluator confirmed the connection attempt failed. The evaluator then repeated this test, but with the identifier in the SAN extension rather than the CN. The evaluator confirmed the connection attempt again failed.

Test 5.2: [conditional]: If wildcards are supported, the evaluator shall present a server certificate containing a wildcard in the left-most label but not preceding the public suffix (e.g. *.example.com). The evaluator shall configure the reference identifier with a single left-most label (e.g. foo.example.com) and verify that the connection succeeds. The evaluator shall configure the reference identifier without a left-most label as in the certificate (e.g. example.com) and verify that the connection fails. The evaluator shall configure the reference identifier with two left-most labels (e.g. bar.foo.example.come) and verify that the connection fails.

The evaluator configured the TLS server to present a certificate with a CN with a wildcard that was in the left-most position (*.leidos.ate). The evaluator first configured the reference identifier on the TOE with a single left-most label (tlss.leidos.ate). The subsequent connection attempt succeeded. The evaluator

repeated this test with the identifier in the SAN extension rather than the CN. The evaluator confirmed the connection attempt again succeeded.

The evaluator then configured the TOE with a reference identifier without a left-most label (leidos.ate). The subsequent connection attempts, with the identifier first in the CN and then in the SAN extension, both failed. Finally, the evaluator configured the reference identifier with two left-most labels (test.tlss.leidos.ate). The subsequent connection attempts, with the identifier first in the CN and then in the SAN extension, both failed.

Test 5.3: [conditional]: If wildcards are supported, the evaluator shall present a server certificate containing a wildcard in the left-most label immediately preceding the public suffix (e.g. *.com). The evaluator shall configure the reference identifier with a single left-most label (e.g. foo.com) and verify that the connection fails. The evaluator shall configure the reference identifier with two left-most labels (e.g. bar.foo.com) and verify that the connection fails.

The evaluator configured the TLS server to present a certificate with a wildcard in the left-most position immediately preceding the public suffix (*.ate). The evaluator first configured the reference identifier on the TOE with a single left-most label (leidos.ate). The subsequent connection attempts, with the identifier first in the CN and then in the SAN extension, both failed. The evaluator then configured the TOE with a reference identifier with two left-most labels (tlss.leidos.ate). The subsequent connection attempts, with the identifier first in the CN and then in the SAN extension, both failed.

Test 5.4: conditional]: If wildcards are not supported, the evaluator shall present a server certificate containing a wildcard in the left-most label (e.g.*.example.com). The evaluator shall configure the reference identifier with a single left-most label (e.g. foo.example.com) and verify that the connection fails.

The TOE supports wildcards and therefore this test is not applicable.

Test 6: conditional] If URI or Service name reference identifiers are supported, the evaluator shall configure the DNS name and the service identifier. The evaluator shall present a server certificate containing the correct DNS name and service identifier in the URIName or SRVName fields of the SAN and verify that the connection succeeds. The evaluator shall repeat this test with the wrong service identifier (but correct DNS name) and verify that the connection fails.

The TOE does not support URI or service name reference identifiers and therefore this test is not applicable.

Test 7: [conditional] If pinned certificates are supported the evaluator shall present a certificate that does not match the pinned certificate and verify that the connection fails.

The TOE does not support pinned certificates and therefore this test is not applicable.

2.1.19.3 FCS_TLSC_EXT.1.3

2.1.19.3.1 TSS Activities

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

In FCS_TLSC_EXT.1.3, the ST selects “with no exceptions”.

2.1.19.3.2 Guidance Activities

None.

2.1.19.3.3 Test Activities

Modified in accordance with TD0513.

The evaluator shall demonstrate that using an invalid certificate (unless excepted) results in the function failing as follows, unless excepted:

Test 1a: The evaluator shall demonstrate that a server using a certificate with a valid certification path successfully connects.

The evaluator attempted a connection from the TOE to a TLS server using a certificate with a valid certification path. The connection attempt succeeded.

Test 1b: The evaluator shall modify the certificate chain used by the server in test 1a to be invalid and demonstrate that a server using a certificate without a valid certification path to a trust store element of the TOE results in an authentication failure.

The evaluator removed an Intermediate CA certificate from the certificate chain presented by the TLS server, and attempted a connection from the TOE. The evaluator observed this connection attempt failed.

Test 1c [conditional]: If the TOE trust store can be managed, the evaluator shall modify the trust store element used in Test 1a to be untrusted and demonstrate that a connection attempt from the same server used in Test 1a results in an authentication failure.

The evaluator modified the trust store on the TOE to remove trust from a certificate in the TLS server’s certificate chain and attempted a connection from the TOE. The evaluator observed this connection attempt failed.

Test 2: The evaluator shall demonstrate that a server using a certificate which has been revoked results in an authentication failure.

This test is performed in conjunction with FIA_X509_EXT.1.1 Test 3.

Test 3: The evaluator shall demonstrate that a server using a certificate which has passed its expiration date results in an authentication failure.

The evaluator attempted a connection from the TOE to a TLS server using an expired certificate. The connection attempt failed.

Test 4: The evaluator shall demonstrate that a server using a certificate which does not have a valid identifier results in an authentication failure.

This test is performed in conjunction with FCS_TLSC_EXT.1.2.

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

2.1.20.1 TSS Activities

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

Section 6.2 of [ST] (“Cryptographic Support”) states the TSF presents secp384r1 as the supported value in the Supported Elliptic Curves (Supported Groups) extension.

2.1.20.2 Guidance Activities

None.

2.1.20.3 Test Activities

The evaluator shall also perform the following test:

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

The evaluator attempted to open a connection from the TOE to a TLS test server configured to use the secp384r1 curve. A connection was successfully established.

2.1.21 TLS Server Protocol (FCS_TLSS_EXT.1)

2.1.21.1 FCS_TLSS_EXT.1.1

2.1.21.1.1 TSS Activities

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

Section 6.2 of [ST] (“Cryptographic Support”) states the TOE’s TLS server implementation supports the following TLS cipher suites in the TOE’s evaluated configuration:

- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289.

This list comprises exactly the list of supported cipher suites specified in the SFR.

2.1.21.1.2 Guidance Activities

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

Section “Setting Up Secure Communications in Axonius Cybersecurity Asset Management Platform v4.0-f” of [CCECG] states no administrator action is required to configure the cryptographic module or TLS 1.2 for client and server communications.

2.1.21.1.3 Test Activities

The evaluator shall also perform the following tests:

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

The evaluator established a connection from a TLS test client to the TOE using each of the cipher suites claimed by the TOE. A wire capture was made of each connection to verify that the connection was completed using the claimed cipher suite.

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

The evaluator attempted to establish a connection to the TOE with a TLS test client configured to use the TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256 cipher. The TOE terminated the connection attempt after receiving the Client Hello message. A second connection attempt was made with the TLS_NULL_WITH_NULL_NULL cipher. The TOE again terminated the connection attempt after receiving the Client Hello message.

Test 3: If RSA key exchange is used in one of the selected ciphersuites, the evaluator shall use a client to send a properly constructed Key Exchange message with a modified EncryptedPreMasterSecret field during the TLS handshake. The evaluator shall verify that the handshake is not completed successfully and no application data flows.

The TOE does not support RSA key exchange and therefore the test is not applicable.

Modified in accordance with TD0469.

Test 4: The evaluator shall perform the following modifications to the traffic:

~~**Test 4.1:** Change the TLS version proposed by the client in the Client Hello to a non-supported TLS version (for example 1.3 represented by the two bytes 03 04) and verify that the server rejects the connection.~~

This test was removed by TD0469.

Test 4.2: Modify a byte in the data of the client's Finished handshake message, and verify that the server rejects the connection and does not send any application data.

The evaluator attempted to establish a connection to the TOE from a TLS test client that sent a modified Client Finished handshake message. The TOE terminated the connection after receiving the Client Finished message from the test client.

Modified in accordance with TD0588.

Test 4.3: 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): ~~Generate a Fatal Alert by sending a Finished message from the client before the client sends a ChangeCipherSpec message, and then send a Client Hello with the session identifier from the previous incomplete session, and verify that the server does not resume the session.~~

The evaluator attempted to establish a connection to the TOE from a TLS test client that sent the Finished Message before the ChangeCipherSpec message. The evaluator then attempted reconnecting to the TOE by sending the Client Hello message with the session identifier from the previous incomplete session. The TOE denied the connection.

Test 4.3i [conditional]: If the TOE does not support session resumption based on session IDs according to RFC4346 (TLS1.1) or RFC5246 (TLS1.2) or session tickets according to RFC5077, the evaluator shall perform the following test:

a) The evaluator shall send a Client Hello with a zero-length session identifier and with a SessionTicket extension containing a zero-length ticket.

b) The evaluator shall verify the server does not send a NewSessionTicket handshake message (at any point in the handshake).

c) The evaluator shall verify the Server Hello message contains a zero-length session identifier or passes the following steps:

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

d) The evaluator shall complete the TLS handshake and capture the SessionID from the ServerHello.

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

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

The TOE supports session resumption so this test is not applicable.

Test 4.3ii [conditional]: If the TOE supports session resumption using session IDs according to RFC4346 (TLS1.1) or RFC5246 (TLS1.2), the evaluator shall carry out the following steps (note that for each of these tests, it is not necessary to perform the test case for each supported version of TLS):

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

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

The evaluator verified that the TOE will successfully resume a session using session IDs. Additionally, it was verified that using a session ID from an interrupted TLS handshake results in the connection attempt being terminated.

Test 4.3iii [conditional]: If the TOE supports session tickets according to RFC5077, the evaluator shall carry out the following steps (note that for each of these tests, it is not necessary to perform the test case for each supported version of TLS):

a) The evaluator shall permit a successful TLS handshake to occur in which a session ticket is exchanged with the non-TOE client. The evaluator shall then attempt to correctly reuse the previous session by sending the session ticket in the ClientHello. The evaluator shall confirm that the TOE responds with a ServerHello with an empty SessionTicket extension, NewSessionTicket, ChangeCipherSpec and Finished messages (as seen in figure 2 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.

The TOE does not support session tickets according to RFC 5077, so this test is not applicable.

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

The evaluator attempted to establish a connection to the TOE from a TLS test client that was configured to send garbled data after the client sent its ChangeCipherSpec message. The TOE terminated the connection after receiving the garbled data from the test client.

2.1.21.2 FCS_TLSS_EXT.1.2

2.1.21.2.1 TSS Activities

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

Section 6.2 of [ST] (“Cryptographic Support”) states the TOE uses TLS 1.2 for client and server communications and in cases where the TOE acts as a TLS server, rejects all other TLS versions and all SSL versions.

2.1.21.2.2 Guidance Activities

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

Section “Setting Up Secure Communications in Axonius Cybersecurity Asset Management Platform v4.0-f” of [CCECG] states no administrator action is required to configure the cryptographic module or TLS 1.2 for client and server communications.

2.1.21.2.3 Test Activities

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

The evaluator used a TLS test client to attempt to open SSL 2.0 and 3.0 and TLS 1.0 and 1.1 connections to the TOE. All of these connection attempts were rejected by the TOE after it received the Client Hello messages.

2.1.21.3 FCS_TLSS_EXT.1.3

2.1.21.3.1 TSS Activities

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

Section 6.2 of [ST] (“Cryptographic Support”) states all supported cipher suites use elliptic curves as the method of key establishment. The TSF uses secp384r1 and sep521r1 as the key agreement parameters in its Key Exchange message.

2.1.21.3.2 Guidance Activities

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

Section “Setting Up Secure Communications in Axonius Cybersecurity Asset Management Platform v4.0-f” of [CCECG] states no administrator action is required to configure the cryptographic module or TLS 1.2 for client and server communications.

2.1.21.3.3 Test Activities

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

Test 1: [conditional] If RSA-based key establishment is selected, the evaluator shall attempt a connection using RSA-based key establishment with a supported size. The evaluator shall verify that the size used matches that which is configured. The evaluator shall repeat this test for each supported size of RSA-based key establishment.

The TOE does not use RSA-based key establishment schemes, so this test is not applicable.

Test 2: conditional] If finite-field (i.e. non-EC) Diffie-Hellman ciphers are selected, the evaluator shall attempt a connection using a Diffie-Hellman key exchange with a supported parameter size or supported group. The evaluator shall verify that the key agreement parameters in the Key Exchange message are the ones configured. The evaluator shall repeat this test for each supported parameter size or group.

The TOE does not use finite field-based key establishment schemes, so this test is not applicable.

Test 3: [conditional] If ECDHE ciphers are selected, the evaluator shall attempt a connection using an ECDHE ciphersuite with a supported curve. The evaluator shall verify that the key agreement parameters in the Key Exchange message are the ones configured. The evaluator shall repeat this test for each supported elliptic curve.

The evaluator used a TLS test client to establish a TLS connection to the TOE using each of the curves claimed by the TOE. In each case, the connection attempt succeeded.

2.2 User Data Protection (FDP)

2.2.1 Encryption of Sensitive Application Data (FDP_DAR_EXT.1)

2.2.1.1 TSS Activities

The evaluator shall examine the TSS to ensure that it describes the sensitive data processed by the application. The evaluator shall then ensure that the following activities cover all of the sensitive data identified in the TSS.

Section 6.3 of [ST] ("User Data Protection"), Table 6 ("Sensitive Data") lists the data considered to be sensitive by the TOE, any communication ("Exchange") of that data, and how the data is protected at rest and in transit. The GUI Credentials, TLS Private Key, and MongoDB encryption key are all protected in accordance with FCS_STO_EXT.1 and additionally leverage platform-provided encryption (LUKS) when stored in non-volatile memory. Adapter credentials are stored in accordance with FCS_STO_EXT.1 using 256 bit AES in CBC mode.

If ***not store any sensitive data*** is selected, the evaluator shall inspect the TSS to ensure that it describes how sensitive data cannot be written to non-volatile memory. The evaluator shall also ensure that this is consistent with the filesystem test below.

The ST does not select "not store any sensitive data", so this activity is not applicable.

2.2.1.2 Guidance Activities

None defined.

2.2.1.3 Test Activities

Evaluation activities (after the identification of the sensitive data) are to be performed on all sensitive data listed that are not covered by FCS_STO_EXT.1.

The evaluator shall inventory the filesystem locations where the application may write data. The evaluator shall run the application and attempt to store sensitive data. The evaluator shall then inspect those areas of the filesystem to note where data was stored (if any), and determine whether it has been encrypted.

If ***leverage platform-provided functionality*** is selected, the evaluation activities will be performed as stated in the following requirements, which vary on a per-platform basis:

For Linux: The Linux platform currently does not provide data-at-rest encryption services which depend upon invocation by application developers. The evaluator shall verify that the Operational User Guidance makes the need to activate platform encryption clear to the end user.

Section 5.2.2.1 of [ST] ("FDP_DAR_EXT.1 Encryption of Sensitive Application Data") specifies FDP_DAR_EXT.1 and selects "leverage platform-provided functionality to encrypt sensitive data" and "protect sensitive data in accordance with FCS_STO_EXT.1". The guidance provided by [CCECG] makes clear to the administrator the need to activate platform encryption in the evaluated configuration. Refer to "Software Download and Installation", subsection "Installation".

2.2.2 Access to Platform Resources (FDP_DEC_EXT.1)

2.2.2.1 FDP_DEC_EXT.1.1

2.2.2.1.1 TSS Activities

None defined.

2.2.2.1.2 Guidance Activities

The evaluator shall perform the platform-specific actions below and inspect user documentation to determine the application's access to hardware resources. The evaluator shall ensure that this is consistent with the selections indicated. The evaluator shall review documentation provided by the application developer and for each resource which it accesses, identify the justification as to why access is required.

The statement of FDP_DEC_EXT.1.1 in Section 5.2.2.2 of [ST] ("FDP_DEC_EXT.1 Access to Platform Resources") selects "network connectivity" as the only platform hardware resource the TOE requires to access.

Section "Initial Configuration", subsection "Connecting adapters", of [CCECG] states the only platform-provided hardware resource the TOE accesses is network connectivity, in order to be able to communicate with adapters on IT entities in its operational environment.

2.2.2.1.3 Test Activities

For Linux: The evaluator shall verify that either the application software or its documentation provides a list of the hardware resources it accesses.

The evaluator confirmed the TOE documentation provides a list of hardware resources the TOE accesses, comprising only network connectivity.

2.2.2.2 FDP_DEC_EXT.1.2

2.2.2.2.1 TSS Activities

None defined.

2.2.2.2.2 Guidance Activities

The evaluator shall perform the platform-specific actions below and inspect user documentation to determine the application's access to sensitive information repositories. The evaluator shall ensure that this is consistent with the selections indicated. The evaluator shall review documentation provided by the application developer and for each sensitive information repository which it accesses, identify the justification as to why access is required.

The statement of FDP_DEC_EXT.1.2 in Section 5.2.2.2 of [ST] ("FDP_DEC_EXT.1 Access to Platform Resources") selects "no sensitive information repositories".

Section "Initial Configuration", subsection "Connecting adapters", of [CCECG] states the TOE does not access any sensitive information repositories.

2.2.2.2.3 Test Activities

For Linux: The evaluator shall verify that either the application software or its documentation provides a list of sensitive information repositories it accesses.

The evaluator confirmed the TOE documentation makes clear the TOE does not access any sensitive information repositories.

2.2.3 Network Communications (FDP_NET_EXT.1)

2.2.3.1 TSS Activities

None defined.

2.2.3.2 Guidance Activities

None defined.

2.2.3.3 Test Activities

The evaluator shall perform the following tests:

Test 1: The evaluator shall run the application. While the application is running, the evaluator shall sniff network traffic ignoring all non-application associated traffic and verify that any network communications witnessed are documented in the TSS or are user-initiated.

The evaluator ran a wire capture on the TOE's platform while the TOE was running. The evaluator confirmed the observed traffic was either user-initiated or comprised TOE-initiated connections to external TLS and SSH servers.

Test 2: The evaluator shall run the application. After the application initializes, the evaluator shall run network port scans to verify that any ports opened by the application have been captured in the ST for the third selection and its assignment. This includes connection-based protocols (e.g. TCP, DCCP) as well as connectionless protocols (e.g. UDP).

An nmap port scan was performed on the platform hosting the TOE prior to initializing the TOE. The TOE was then initialized and the pot scan repeated. The evaluator determined the TOE did not open any new ports (either TCP or UDP).

2.3 Identification and Authentication (FIA)

2.3.1 X.509 Certificate Validation (FIA_X509_EXT.1)

2.3.1.1 FIA_X509_EXT.1.1

2.3.1.1.1 TSS Activities

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

Section 6.4 of [ST] ("Identification and Authentication") states the check of validity of certificates takes place when establishing trusted communications. The TOE performs certificate and certificate path validation in accordance with RFC 5280.

2.3.1.1.2 Guidance Activities

None defined.

2.3.1.1.3 Test Activities

Modified in accordance with TD0601.

The tests described must be performed in conjunction with the other certificate services evaluation activities, including the functions in FIA_X509_EXT.2.1. The tests for the extendedKeyUsage rules are performed in conjunction with the uses that require those rules. If the application supports chains of length four or greater, the evaluator shall create a chain of at least four certificates: the node certificate to be tested, two Intermediate CAs, and the self-signed Root CA. If the application supports a maximum trust depth of two, then a chain with no Intermediate CA should instead be created.

Test 1: The evaluator shall demonstrate that validating a certificate without a valid certification path results in the function failing, for each of the following reasons, in turn:

- by establishing a certificate path in which one of the issuing certificates is not a CA certificate,
- by omitting the basicConstraints field in one of the issuing certificates,
- by setting the basicConstraints field in an issuing certificate to have CA=False,
- by omitting the CA signing bit of the key usage field in an issuing certificate, and
- by setting the path length field of a valid CA field to a value strictly less than the certificate path.

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

The evaluator attempted to validate a certificate that did not have a valid certificate path, for each of the listed failure cases. The evaluator confirmed the tested function failed in each of these cases. The evaluator then established a valid certificate path and confirmed the TOE as able to establish a connection to an external TLS server. The evaluator then removed the root CA certificate and attempted to connect to the TLS server. The evaluator verified the connection was unsuccessful.

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

The evaluator attempted to connect the TOE to a TLS server that was using an expired X.509 certificate. The evaluator verified the TOE rejected the connection attempt.

Test 3: The evaluator shall test that the TOE can properly handle revoked certificates-–conditional on whether CRL, OCSP, OCSP Stapling or OCSP Multi-stapling is selected; if multiple methods are selected, then the following tests shall be performed for each method:

The evaluator shall test revocation of the node certificate.

The evaluator shall also test revocation of an intermediate CA certificate (i.e. the intermediate CA certificate should be revoked by the root CA), if intermediate CA certificates are supported. If OCSP Stapling per RFC6066 is the only supported revocation method, this test is omitted.

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.

The evaluator verified that the TOE could use OCSP to check the revocation status of both a leaf certificate and an intermediate CA, and that it would reject a connection where either of those certificates was revoked.

Test 4: If any OCSP option is selected, the evaluator shall configure the OCSP server or use a man-in-the-middle tool to present a certificate that does not have the OCSP signing purpose and verify that validation of the OCSP response fails. If CRL is selected, the evaluator shall configure the CA to sign a CRL with a certificate that does not have the cRLsign key usage bit set, and verify that validation of the CRL fails.

The evaluator verified that the TOE would reject an OCSP response from a responder that lacked the OCSP signing purpose in its certificate.

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.)

The evaluator attempted to open a TLS connection from the TOE to a TLS test server. The test server's certificate had one of its first eight bytes modified. The TOE terminated the connection attempt after receiving the server certificate.

Test 6: The evaluator shall modify any byte in the last byte of the certificate and demonstrate that the certificate fails to validate. (The signature on the certificate will not validate.)

The evaluator attempted to open a TLS connection from the TOE to a TLS test server. The test server's certificate had its last byte modified. The TOE terminated the connection attempt after receiving the server certificate.

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

The evaluator attempted to open a TLS connection from the TOE to a TLS test server. The test server's certificate had its public key modified. The TOE terminated the connection attempt after receiving the server certificate.

Test 8a: (Conditional on support for EC certificates as indicated in FCS_COP.1(3)). The evaluator shall establish a valid, trusted certificate chain consisting of an EC leaf certificate, an EC Intermediate CA certificate not designated as a trust anchor, and an EC certificate designated as a trusted anchor, where the elliptic curve parameters are specified as a named curve. The evaluator shall confirm that the TOE validates the certificate chain.

The TOE does not support EC certificates and therefore this test is not applicable.

Test 8b: (Conditional on support for EC certificates as indicated in FCS_COP.1(3)). The evaluator shall replace the intermediate certificate in the certificate chain for Test 8a with a modified certificate, where the modified intermediate CA has a public key information field where the EC parameters uses an explicit format version of the Elliptic Curve parameters in the public key information field of the intermediate CA certificate from Test 8a, and the modified Intermediate CA certificate is signed by the trusted EC root CA, but having no other changes. The evaluator shall confirm the TOE treats the certificate as invalid.

The TOE does not support EC certificates and therefore this test is not applicable.

2.3.1.2 FIA_X509_EXT.1.2

2.3.1.2.1 TSS Activities

None defined.

2.3.1.2.2 Guidance Activities

None defined.

2.3.1.2.3 Test Activities

Modified in accordance with TD0495

The tests described must be performed in conjunction with the other certificate services assurance activities, including the functions in FIA_X509_EXT.2.1. If the application supports chains of length four or greater, the evaluator shall create a chain of at least four certificates: the node certificate to be tested, two Intermediate CAs, and the self-signed Root CA. If the application supports a maximum trust depth of two, then a chain with no Intermediate CA should instead be created.

Test 1: The evaluator shall ~~construct a certificate path, such~~ ensure that the certificate of at least one of the CAs ~~issuing the TOE's certificate~~ in the chain does not contain the basicConstraints extension. The evaluator shall confirm that validation of the certificate path fails (i) as part of the validation of the peer certificate belonging to this chain; and/or (ii) when attempting to add the CA certificate without the basicConstraints extension to the TOE's trust store.

This test is performed in conjunction with FIA_X509_EXT.1.1 Test 1.

Test 2: The evaluator ~~shall construct a certificate path, such~~ ensure that the certificate of at least one of the CAs ~~issuing the TOE's certificate~~ in the chain has the CA flag in the basicConstraints extension not set (or set to FALSE). The evaluator shall confirm that validation of the certificate path fails (i) as part of the validation of the peer certificate belonging to this chain; and/or (ii) when attempting to add the CA certificate with the CA flag not set (or set to FALSE) in the basicConstraints extension to the TOE's trust store.

~~**Test 3**: The evaluator shall construct a certificate path, such that the certificate of the CA issuing the TOE's certificate has the CA flag in the basicConstraints extension set to TRUE. The validation of the certificate path succeeds.~~

This test is performed in conjunction with FIA_X509_EXT.1.1 Test 1.

2.3.2 X.509 Certificate Authentication (FIA_X509_EXT.2)

2.3.2.1 TSS Activities

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.

Section 6.4 of [ST] ("Identification and Authentication") states the TOE chooses what certificates to use to validate the authenticity of remote adapter sources based on what is presented to it as part of establishing a TLS session. The TOE is only assigned one certificate for its own use, so there is only one certificate that it will present in cases where a remote entity may need to validate it.

The guidance provided by [CCECG] includes a description of how to configure the operational environment to enable the TOE to use its certificate. Refer to "Setting Up Secure Communications in Axonius Cybersecurity Asset Management Platform v4.0-f", subsection "TLS and Certificates".

The evaluator shall examine the TSS to confirm that it describes the behavior of the TOE when a connection cannot be established during the validity check of a certificate used in establishing a trusted channel. The evaluator shall verify that any distinctions between trusted channels are described.

Section 6.4 of [ST] states in the event that the revocation status of a certificate cannot be verified because the OCSP responder cannot be reached, the TOE will not accept the certificate. There is no distinction between trusted channels.

2.3.2.2 Guidance Activities

If the requirement that the administrator is able to specify the default action, then the evaluator shall ensure that the operational guidance contains instructions on how this configuration action is performed.

The statement of FIA_X509_EXT.2.2 in Section 5.2.3.2 of [ST] ("FIA_X509_EXT.2 X.509 Certificate Authentication") selects "not accept the certificate". Therefore, there is no capability for the administrator to specify the default action, and no requirement for operational guidance to provide such instructions.

2.3.2.3 Test Activities

The evaluator shall perform the following test for each trusted channel:

Test 1: 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 operational guidance to determine that all supported administrator-configurable options behave in their documented manner.

The first part of this test was covered by FIA_X509_EXT.1.1 Test 3. That test required the TOE to perform certificate validation by connecting to a non-TOE IT entity. In the second part of the test, the evaluator verified the TOE rejected the connection attempt when it was unable to verify the validity of the certificate.

Test 2: The evaluator shall demonstrate that an invalid certificate that requires certificate validation checking to be performed in at least some part by communicating with a non-TOE IT entity cannot be accepted.

This test is covered by testing of FIA_X509_EXT.1 where invalid certificate tests resulted in connection failure.

2.4 Security Management (FMT)

2.4.1 Secure by Default Configuration (FMT_CFG_EXT.1)

2.4.1.1 FMT_CFG_EXT.1.1

2.4.1.1.1 TSS Activities

The evaluator shall check the TSS to determine if the application requires any type of credentials and if the application installs with default credentials.

Section 6.5 of [ST] ("Security Management") states the TOE provides a web-based GUI that requires user authentication to access. During initial configuration of the TOE, the administrator must specify an initial username and password to be used to login to the TOE—the TOE is not pre-loaded with default credentials.

2.4.1.1.2 Guidance Activities

None defined.

2.4.1.1.3 Test Activities

If the application uses any default credentials the evaluator shall run the following tests.

Test 1: The evaluator shall install and run the application without generating or loading new credentials and verify that only the minimal application functionality required to set new credentials is available.

Test 2: The evaluator shall attempt to clear all credentials and verify that only the minimal application functionality required to set new credentials is available.

Test 3: The evaluator shall run the application, establish new credentials and verify that the original default credentials no longer provide access to the application.

The TOE does not use any default credentials. As such, these tests are not applicable.

2.4.1.2 FMT_CFG_EXT.1.2

2.4.1.2.1 TSS Activities

None defined.

2.4.1.2.2 Guidance Activities

None defined.

2.4.1.2.3 Test Activities

Modified in accordance with TD0519.

The evaluator shall install and run the application. The evaluator shall inspect the filesystem of the platform (to the extent possible) for any files created by the application and ensure that their permissions are adequate to protect them. The method of doing so varies per platform.

For Linux: The evaluator shall run the command `find -L . -perm /002` inside the application's data directories to ensure that all files are not world-writable. The command should not print any files.

The evaluator executed "`find -L . -perm /002`" inside the TOE's data directory (`/home/ubuntu/cortex`) and verified that all files are not world-writable. The command did not print any files.

2.4.2 Supported Configuration Mechanism (FMT_MEC_EXT.1)

2.4.2.1 TSS Activities

Modified in accordance with TD0437.

The evaluator shall review the TSS to identify the application's configuration options (e.g. settings) and determine whether these are stored and set using the mechanisms supported by the platform or implemented by the application in accordance with the PP-Module for File Encryption. At a minimum the TSS shall list settings related to any SFRs and any settings that are mandated in the operational guidance in response to an SFR.

Section 6.5 of [ST] ("Security Management") states the TOE is installed in `/home/ubuntu/cortex` and executes as the Ubuntu user. Configuration settings for the TOE, comprising adapter connections, custom CA certificates, and hostname, are stored in the Ubuntu user's home directory in the `cortex` subdirectory.

Conditional: If "***implement functionality to encrypt and store configuration options as defined by FDP_PRT_EXT.1 in the PP-Module for File Encryption***" is selected, the evaluator shall ensure that the TSS identifies those options, as well as indicates where the encrypted representation of these options is stored.

The ST does not make this selection.

2.4.2.2 Guidance Activities

None defined.

2.4.2.3 Test Activities

Modified in accordance with TD0437, TD0465, and TD0543.

If "***invoke the mechanisms recommended by the platform vendor for storing and setting configuration options***" is chosen, the method of testing varies per platform as follows:

For Linux: The evaluator shall run the application while monitoring it with the utility `strace`. The evaluator shall make security-related changes to its configuration. The evaluator shall verify that `strace` logs corresponding changes to configuration files that reside in `/etc` (for system-specific configuration), in the user's home directory (for user-specific configuration), or `/var/lib/` (for configurations controlled by UI and not intended to be directly modified by an administrator).

The evaluator modified the TOE's settings for certificate verification depth while monitoring the TOE with `strace`. Changes were written to the appropriate directories for the application.

If "***implement functionality to encrypt and store configuration options as defined by FDP_PRT_EXT.1 in the PP-Module for File Encryption***" is selected, for all configuration options listed in the TSS as being stored and protected using encryption, the evaluator shall examine the contents of the configuration option storage (identified in the TSS) to determine that the options have been encrypted.

The ST does not make this selection.

2.4.3 Specification of Management Functions (FMT_SMF.1)

2.4.3.1 TSS Activities

None defined.

2.4.3.2 Guidance Activities

The evaluator shall verify that every management function mandated by the PP is described in the operational guidance and that the description contains the information required to perform the management duties associated with the management function.

As described in Section 6.5 of [ST] ("Security Management"), the TOE supports the following security-relevant management functions:

- Configuration of adapter connections
- Upload custom certificates
- Set hostname.

Section "Configuring the Adapter Connection" of [CCECG] describes how the administrator can configure a new adapter connection.

Section "TLS and Certificates" of [CCECG] describes how the administrator uploads custom certificates the TOE can use to verify TLS connections to adapters.

Section "Initial Configuration" of [CCECG] describes how the administrator sets the hostname, by configuring a static IP address or configuring the TOE to use a dynamically assigned IP address if DHCP is being used.

2.4.3.3 Test Activities

The evaluator shall test the application's ability to provide the management functions by configuring the application and testing each option selected from above. The evaluator is expected to test these functions in all the ways in which the ST and guidance documentation state the configuration can be managed.

The evaluator demonstrated the ability to access the managed functionality specified in [CCECG].

2.5 Privacy (FPR)

2.5.1 User Consent for Transmission of Personally Identifiable Information (FPR_ANO_EXT.1)

2.5.1.1 TSS Activities

The evaluator shall inspect the TSS documentation to identify functionality in the application where PII can be transmitted.

Section 6.6 of [ST] ("Privacy") states the TOE does not collect PII or transmit it over a network.

2.5.1.2 Guidance Activities

None defined.

2.5.1.3 Test Activities

If ***require user approval before executing*** is selected, the evaluator shall run the application and exercise the functionality responsible for transmitting PII and verify that user approval is required before transmission of the PII.

The ST does not make this selection.

2.6 Protection of the TSF (FPT)

2.6.1 Anti-Exploitation Capabilities (FPT_AEX_EXT.1)

2.6.1.1 FPT_AEX_EXT.1.1

2.6.1.1.1 TSS Activities

The evaluator shall ensure that the TSS describes the compiler flags used to enable ASLR when the application is compiled.

Section 6.7 of [ST] ("Protection of the TSF") states the TOE is implemented in interpreted code and is not just-in-time compiled, and therefore compiler flags to enforce ASLR are not necessary.

2.6.1.1.2 Guidance Activities

None Defined.

2.6.1.1.3 Test Activities

Modified in accordance with TD0544

The evaluator shall perform either a static or dynamic analysis to determine that no memory mappings are placed at an explicit and consistent address. The method of doing so varies per platform. For those platforms requiring the same application running on two different systems, the evaluator may alternatively use the same device. After collecting the first instance of mappings, the evaluator must uninstall the application, reboot the device, and reinstall the application to collect the second instance of mappings.

For Linux: The evaluator shall run the same application on two different Linux systems. The evaluator shall then compare their memory maps using `pmap -x PID` to ensure the two different instances share no mapping locations.

The evaluator ran the TOE on two different Linux systems. A comparison of their memory maps showed that the two instances did not share memory mapping locations.

2.6.1.2 FPT_AEX_EXT.1.2

2.6.1.2.1 TSS Activities

None defined.

2.6.1.2.2 Guidance Activities

None defined.

2.6.1.2.3 Test Activities

The evaluator shall verify that no memory mapping requests are made with write and execute permissions. The method of doing so varies per platform.

For Linux: The evaluator shall perform static analysis on the application to verify that both

- `mmap` is never be invoked with both the `PROT_WRITE` and `PROT_EXEC` permissions, and
- `mprotect` is never invoked with the `PROT_EXEC` permission.

The evaluator performed a text search of the application source code and confirmed the strings “mmap” and “mprotect” do not appear anywhere in the application.

2.6.1.3 FPT_AEX_EXT.1.3

2.6.1.3.1 TSS Activities

None defined.

2.6.1.3.2 Guidance Activities

None defined.

2.6.1.3.3 Test Activities

Modified in accordance with TD0435.

The evaluator shall configure the platform in the ascribed manner and carry out one of the prescribed tests:

For Linux: The evaluator shall ensure that the application can successfully run on a system with either SELinux or AppArmor enabled and ~~enforcing~~ in enforce mode.

The evaluator installed and executed the TOE in accordance with the guidance documentation. The evaluator confirmed AppArmor was enabled and in enforce mode and that the TOE ran successfully.

2.6.1.4 FPT_AEX_EXT.1.4

2.6.1.4.1 TSS Activities

None defined.

2.6.1.4.2 Guidance Activities

None defined.

2.6.1.4.3 Test Activities

The evaluator shall run the application and determine where it writes its files. For files where the user does not choose the destination, the evaluator shall check whether the destination directory contains executable files. This varies per platform:

For Linux: The evaluator shall run the program, mimicking normal usage, and note where all user-modifiable files are written. The evaluator shall ensure that there are no executable files stored in the same directories to which the application wrote user-modifiable files.

The evaluator ran the TOE, mimicking normal usage. The evaluator then verified no executable were stored in the same directories to which the application wrote user-modifiable files.

2.6.1.5 FPT_AEX_EXT.1.5

2.6.1.5.1 TSS Activities

None defined.

2.6.1.5.2 Guidance Activities

None defined.

2.6.1.5.3 Test Activities

The evaluator will inspect every native executable included in the TOE to ensure that stack-based buffer overflow protection is present.

The TOE comprises interpreted code without any just-in-time compilation, so this activity is not applicable.

2.6.2 Use of Supported Services and APIs (FPT_API_EXT.1)

2.6.2.1 TSS Activities

The evaluator shall verify that the TSS lists the platform APIs used in the application.

Section 6.7 of [ST] ("Protection of the TSF") states the TOE does not use any undocumented platform APIs and no system calls are directly invoked by the TOE. The TOE is entirely Dockerized Python/JavaScript, so all calls are indirect.

2.6.2.2 Guidance Activities

None defined.

2.6.2.3 Test Activities

The evaluator shall then compare the list with the supported APIs (available through e.g. developer accounts, platform developer groups) and ensure that all APIs listed in the TSS are supported.

As stated in the TSS, the TOE does not directly invoke any APIs, and so this activity is not applicable.

2.6.3 Use of Third Party Libraries (FPT_LIB_EXT.1)

2.6.3.1 TSS Activity

None defined.

2.6.3.2 Guidance Activity

None defined.

2.6.3.3 Test Activity

The evaluator shall install the application and survey its installation directory for dynamic libraries. The evaluator shall verify that libraries found to be packaged with or employed by the application are limited to those in the assignment.

The evaluator surveyed the TOE's installation directory for dynamic libraries. The evaluator determined the installation directory and verified it did not contain any dynamic libraries.

Due to the TOE being a containerized application a survey of the installation directory did not fully account for the libraries used by the TOE. The evaluator identified a trio of package manifest files that tracked the libraries used by the TOE. These were examined and found to be consistent with those specified in the assignment.

2.6.4 Software Identification and Versions (FPT_IDV_EXT.1)

2.6.4.1 TSS Activity

If **"other version information"** is selected the evaluator shall verify that the TSS contains an explanation of the versioning methodology.

Section 5.2.6.3 of [ST] ("FPT_IDV_EXT.1 Software Identification and Versions") selects "other version information" in FPT_IDV_EXT.1.1 and completes the assignment with "semantic versioning (SemVer)". Section 6.7 of [ST] ("Protection of the TSF") explains the TOE versioning methodology. The TOE uses the version format x.y.z-f (or x.y.z-fed), where x is the major version (an integer), y is the minor version (an integer), z is the patch version (an integer), and 'f' (or 'fed') is fixed and flags this as a 'federal' version of the product.

2.6.4.2 Guidance Activity

None defined.

2.6.4.3 Test Activity

The evaluator shall install the application, then check for the / existence of version information. If SWID tags is selected the evaluator shall check for a .swidtag file. The evaluator shall open the file and verify that is contains at least a SoftwareIdentity element and an Entity element.

This test was performed in conjunction with FPT_TUD_EXT.1.2, which demonstrates the ability of the TOE to display version information. The TOE does not use SWID tags, so that portion of the test is not applicable.

2.6.5 Integrity for Installation and Update (FPT_TUD_EXT.1)

2.6.5.1 FPT_TUD_EXT.1.1

2.6.5.1.1 TSS Activities

None defined.

2.6.5.1.2 Guidance Activities

The evaluator shall check to ensure the guidance includes a description of how updates are performed.

Section "Software Download and Installation" of [CCECG] provides a description of how updates are performed. When an update is available, the administrator is notified and provided with a download link by their Axonius account representative. The administrator downloads the update as a .deb package and places it in the `/home/ubuntu` directory on the Linux platform and then installs the update using the `dpkg` package manager command line instruction.

2.6.5.1.3 Test Activities

The evaluator shall check for an update using procedures described in either the application documentation or the platform documentation and verify that the application does not issue an error. If it is updated or if it reports that no update is available this requirement is considered to be met.

The evaluator followed the guidance documentation to check for a TOE update. The TOE reported that no update was available.

2.6.5.2 FPT_TUD_EXT.1.2

2.6.5.2.1 TSS Activities

None defined.

2.6.5.2.2 Guidance Activities

The evaluator shall verify guidance includes a description of how to query the current version of the application.

Section “Software Download and Installation” of [CCECG] provides a description of how to query the current version of the TOE by going to the **Settings -> About** page of the Web UI, which displays the current version of the TOE.

2.6.5.2.3 Test Activities

The evaluator shall query the application for the current version of the software according to the operational user guidance. The evaluator shall then verify that the current version matches that of the documented and installed version.

The evaluator followed the instructions in the guidance documentation to query the current version of the TOE. The evaluator verified this matched the documented and installed version.

2.6.5.3 FPT_TUD_EXT.1.3

2.6.5.3.1 TSS Activities

None defined.

2.6.5.3.2 Guidance Activities

None defined.

2.6.5.3.3 Test Activities

Modified in accordance with TD0548.

For iOS: The evaluator shall consider the requirement met because the platform forces applications to write all data within the application working directory (sandbox).

For all other platforms: The evaluator shall install the application and then locate all of its executable files. The evaluator shall then, for each file, save off either a hash of the file or a copy of the file itself. The evaluator shall then run the application and exercise all features of the application as described in the ST. The evaluator shall then compare each executable file with the either the saved hash or the saved copy of the files. The evaluator shall verify that these are identical.

The evaluator obtained a hash of all of the TOE's executable files. After performing some testing activity, a second hash of these files was generated. It was found to be unchanged from the first.

2.6.5.4 FPT_TUD_EXT.1.4

2.6.5.4.1 TSS Activities

Modified in accordance with TD0561.

The evaluator shall verify that the TSS identifies how ~~the application installation package and~~ updates to ~~it~~ the application are signed by an authorized source. The definition of an authorized source must be contained in the TSS. The evaluator shall also ensure that the TSS (or the operational guidance) describes how candidate updates are obtained.

Section 6.7 of [ST] ("Protection of the TSF") states the TOE is packaged as a digitally signed .deb file, signed using an ECC (NIST P384) key. Axonius signs the .deb package using its private key and the Debian Package Manager utility automatically verifies the signature of the .deb package using the published public key before proceeding with package installation.

Section 6.7 of [ST] states the TOE can check for software updates. The system regularly checks a service controlled by Axonius for the latest version. When the installed software is one or more minor versions behind, the user receives a message from Axonius prompting the user to request an upgrade, with an email link and the latest available version number. The user obtains candidate updates by downloading them directly from Axonius's website and placing them into the `/home/ubuntu` directory. At this point, the administrator updates the TOE using standard `dpkg` package manager instructions for a .deb installation.

2.6.5.4.2 Guidance Activities

None defined.

2.6.5.4.3 Test Activities

None defined.

2.6.5.5 FPT_TUD_EXT.1.5

2.6.5.5.1 TSS Activities

The evaluator shall verify that the TSS identifies how the application is distributed. If "with the platform" is selected the evaluated shall perform a clean installation or factory reset to confirm that TOE software is included as part of the platform OS. If "as an additional package" is selected the evaluator shall perform the tests in FPT_TUD_EXT.2.

Section 6.7 of [ST] states the TOE is distributed as a .deb package.

Section 5.2.6.5 of [ST] ("FPT_TUD_EXT.1 Integrity for Installation and Update") specifies the application is distributed "as an additional software package to the platform OS". Refer to Section 2.6.6 for evaluation activities associated with FPT_TUD_EXT.2.

2.6.5.5.2 Guidance Activities

None defined.

2.6.5.5.3 Test Activities

None defined.

2.6.6 Integrity for Installation and Update (FPT_TUD_EXT.2)

2.6.6.1 FPT_TUD_EXT.2.1

2.6.6.1.1 TSS Activities

None defined.

2.6.6.1.2 Guidance Activities

None defined.

2.6.6.1.3 Test Activities

The evaluator shall verify that application updates are distributed in the format supported by the platform. This varies per platform:

For Linux: The evaluator shall ensure that the application is packaged in the format of the package management infrastructure of the chosen distribution. For example, applications running on Red Hat and Red Hat derivatives shall be packaged in RPM format. Applications running on Debian and Debian derivatives shall be packaged in DEB format.

The evaluator verified that the TOE's installer is packaged in DEB format.

2.6.6.2 FPT_TUD_EXT.2.2

2.6.6.2.1 TSS Activities

None defined.

2.6.6.2.2 Guidance Activities

None defined.

2.6.6.2.3 Test Activities

For All Other Platforms [i.e., not Android or iOS]: The evaluator shall record the path of every file on the entire filesystem prior to installation of the application, and then install and run the application. Afterwards, the evaluator shall then uninstall the application, and compare the resulting filesystem to the initial record to verify that no files, other than configuration, output, and audit/log files, have been added to the filesystem.

The evaluator recorded the path of every file on the entire file system prior to installation of the TOE. The evaluator then installed and ran the TOE, after which the evaluator uninstalled the TOE. The evaluator again recorded the path of every file on the entire file system and verified that no files have been added to the file system, other than configuration, output, and audit/log files.

2.6.6.3 FPT_TUD_EXT.2.3

Added in accordance with TD0561.

2.6.6.3.1 TSS Activities

The evaluator shall verify that the TSS identifies how the application installation package is signed by an authorized source. The definition of an authorized source must be contained in the TSS.

Section 6.7 of [ST] ("Protection of the TSF") states the TOE is distributed as a digitally signed .deb file, signed using an ECC (NIST P384) key. Axonius signs the .deb package using its private key and the Debian Package Manager utility automatically verifies the signature of the .deb package using the published public key before proceeding with package installation.

2.6.6.3.2 Guidance Activities

None.

2.6.6.3.3 Test Activities

None.

2.7 Trusted Path/Channels (FTP)

2.7.1 Protection of Data in Transit (FTP_DIT_EXT.1)

2.7.1.1 TSS Activities

For platform-provided functionality, the evaluator shall verify the TSS contains the calls to the platform that TOE is leveraging to invoke the functionality.

Section 5.2.7.1 of [ST] ("FTP_DIT_EXT.1 Protection of Data in Transit") specifies the application shall encrypt all transmitted data with HTTPS as client, HTTPS as server, TLS, and SSH. As such, the TOE does not invoke platform-provided functionality for protection of data in transit.

2.7.1.2 Guidance Activities

None defined.

2.7.1.3 Test Activities

The evaluator shall perform the following tests:

Test 1: The evaluator shall exercise the application (attempting to transmit data; for example by connecting to remote systems or websites) while capturing packets from the application. The evaluator shall verify from the packet capture that the traffic is encrypted with HTTPS, TLS, DTLS, SSH, or IPsec in accordance with the selection in the ST.

This test was conducted in conjunction with testing of the applicable protocol SFRs (FCS_HTTPS_EXT.1/Client, FCS_HTTPS_EXT.1/Server, FCS_SSHC_EXT.1, FCS_TLSC_EXT.1, and FCS_TLSS_EXT.1), where packet captures from those tests showed that traffic was encrypted with the applicable protocol.

Test 2: The evaluator shall exercise the application (attempting to transmit data; for example by connecting to remote systems or websites) while capturing packets from the application. The evaluator shall review the packet capture and verify that no sensitive data is transmitted in the clear.

This test was conducted in conjunction with testing of the applicable protocol SFRs (FCS_HTTPS_EXT.1/Client, FCS_HTTPS_EXT.1/Server, FCS_SSHC_EXT.1, FCS_TLSC_EXT.1, and FCS_TLSS_EXT.1), where packet captures from those tests confirmed no sensitive data (or indeed any data) was transmitted in the clear.

Test 3: The evaluator shall inspect the TSS to determine if user credentials are transmitted. If credentials are transmitted the evaluator shall set the credential to a known value. The evaluator shall capture packets from the application while causing credentials to be transmitted as described in the TSS. The evaluator shall perform a string search of the captured network packets and verify that the plaintext credential previously set by the evaluator is not found.

The evaluator examined the TSS in [ST] and determined user credentials are transmitted in the following circumstances when the TOE communicates with a remote system via an installed adapter. In this case, the TOE transmits credentials to the adapter on the remote system in order to gain access to the remote system. The evaluator set the credential to a known value and attempted to establish a connection from the TOE to an SSH server. The evaluator captured packets transmitted from the TOE to the SSH server and performed a string search for the plaintext password credential. The evaluator verified the string was not found in the captured packets.

3. Security Assurance Requirement Assurance Activities

3.1 Development (ADV)

3.1.1 Basic Functional Specification (ADV_FSP.1)

3.1.1.1 Assurance Activity

There are no specific assurance activities associated with these SARs, except ensuring the information is provided. The functional specification documentation is provided to support the evaluation activities described in Section 5.1, and other activities described for AGD, ATE, and AVA SARs. The requirements on the content of the functional specification information is implicitly assessed by virtue of the other assurance activities being performed; if the evaluator is unable to perform an activity because there is insufficient interface information, then an adequate functional specification has not been provided.

The Assurance Activities identified above provided sufficient information to determine the appropriate content for the TSS section and to perform the assurance activities. Since these are directly associated with the SFRs, and are implicitly already done, no additional documentation or analysis is necessary.

3.2 Guidance Documents (AGD)

3.2.1 Operational User Guidance (AGD_OPE.1)

3.2.1.1 Assurance Activity

Some of the contents of the operational guidance will be verified by the assurance activities in Section 5.1 and evaluation of the TOE according to the [CEM]. The following additional information is also required. If cryptographic functions are provided by the TOE, the operational guidance shall contain instructions for configuring the 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. The documentation must describe the process for verifying updates to the TOE by verifying a digital signature – this may be done by the TOE or the underlying platform. The evaluator shall verify that this process includes the following steps: Instructions for obtaining the update itself. This should include instructions for making the update accessible to the TOE (e.g., placement in a specific directory). Instructions for initiating the update process, as well as discerning whether the process was successful or unsuccessful. This includes generation of the hash/digital signature. The TOE will likely contain security functionality that does not fall in the scope of evaluation under this PP. The operational guidance shall make it clear to an administrator which security functionality is covered by the evaluation activities.

Cryptographic functions are provided by the TOE, which implements a single cryptographic engine (OpenSSL 1.0.2 with the FIPS Object Module 2.0.16). As described in the Guidance Assurance activities in Section 2.1 of this document, no administrator action is required to configure the cryptographic module.

The guidance provided by [CCECG] includes a description of how updates are performed. The description covers how to obtain the update and make it accessible to the TOE and how to initiate the update process. Refer to “Software Download and Installation”. All updates are performed using the `dpkg` package manager command, which verifies the digital signature on the upgrade package as part of the upgrade process. The administrator can discern the success or otherwise of an upgrade attempt by viewing the running TOE version on the **Settings -> About** page of the Web UI.

Section “Logical Boundaries” of [CCECG] describes the security functionality of the TOE that falls within the scope of evaluation, while section “Excluded from the Evaluation” lists specific TOE functionality not covered by the evaluation.

3.2.2 Preparative Procedures (AGD_PRE.1)

3.2.2.1 Assurance Activity

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

The TOE in its evaluated configuration is supported on a single platform that is adequately addressed in the guidance documentation. Section “Physical Boundaries” of [CCECG] states the TOE consists of exactly one instance of the Axonius Cybersecurity Asset Management Platform provided as either a .deb file or a .deb file packaged as an OVA file, installed and executed on the following platform:

- Docker runtime engine v19.0.3
- VMware ESXi 6.5
- AMD Ryzen Threadripper 1950X (Zen microarchitecture) processor.

3.3 Tests (ATE)

3.3.1 Independent Testing – Conformance (ATE_IND.1)

3.3.1.1 Assurance Activity

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

While it is not necessary to have one test case per test listed in an Assurance Activity, the evaluator must document in the test plan that each applicable testing requirement in the ST is covered. The test plan identifies the platforms to be tested, and for those platforms not included in the test plan but included in the ST, the test plan provides a justification for not testing the platforms. This justification must address the differences between the tested platforms and the untested platforms, and make an argument that the differences do not affect the testing to be performed. It is not sufficient to merely assert that the differences have no affect; rationale must be provided. If all platforms claimed in the ST are tested, then no rationale is necessary. The test plan describes the composition of each platform to be tested, and any setup that is necessary beyond what is contained in the AGD documentation. It should be noted that the evaluator is expected to follow the AGD documentation for installation and setup of each platform either as part of a test or as a standard pre-test condition. This may include special test drivers or tools. For each driver or tool, an argument (not just an assertion) should be provided that the driver or tool will not adversely affect the performance of the functionality by the TOE and its platform.

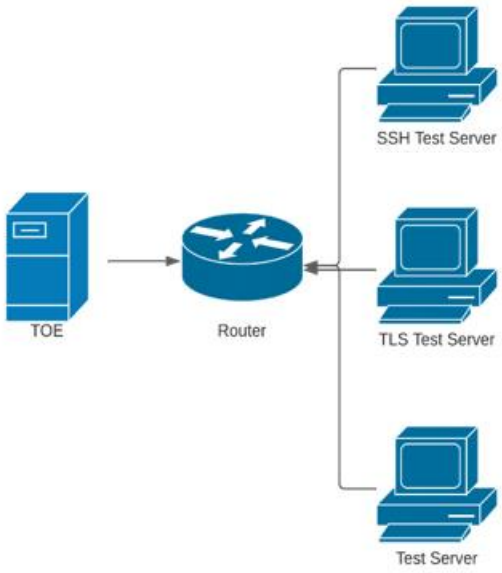
This also includes the configuration of the cryptographic engine to be used. The cryptographic algorithms implemented by this engine are those specified by this PP and used by the cryptographic protocols being evaluated (IPsec, TLS, SSH). The test plan identifies high-level test objectives as well as the test procedures to be followed to achieve those objectives. These procedures include expected results.

The test report (which could just be an annotated version of the test plan) details the activities that took place when the test procedures were executed, and includes the actual results of the tests. This shall be a cumulative account, so if there was a test run that resulted in a failure; a fix installed; and then a successful re-run of the test, the report would show a "fail" and "pass" result (and the supporting details), and not just the "pass" result.

Testing of the TOE was performed at the Leidos Accredited Testing and Evaluation Lab located in Columbia, Maryland from June 2021 through February 2022.

The evaluation team compiled a detailed test plan and report with a complete set of activities that follow [PP_APP_v1.3], [PKG_TLS_V1.1], and [PP_SSH_EP_v1.0]. Each test case was documented with evidence of passing results and a pass verdict.

The following figure depicts the test environment established for testing the TOE.



The test configuration included the following devices in the operational environment of the TOE:

- TOE platform—execution environment for the TOE, comprising:
  - Docker runtime engine v19.0.3
  - Ubuntu 16.04
  - VMware ESXi 6.5
  - AMD Ryzen Threadripper 1950X (Zen microarchitecture) processor
- Test Server—used for general testing and collection of test artifacts. It included the following software:
  - Ubuntu 20.04
  - Sslyze v5.0.0

- TLS Test Server (with OCSP Responder)—used to test TLS client, TLS server, X.509 requirements. It included the following software:
    - Ubuntu 18.04.1
    - OpenSSL v1.1.1
    - Wireshark v2.6.1.0
    - XCA v1.3.2
    - Nmap v7.6
    - Hex Editor v1.2.13
    - CCTL's custom TLS Server and TLS Client test tools
- SSH Server—used to test SSH client. It included the following software:
    - Ubuntu 16.04.7
    - OpenSSH v7.2p2
    - Wireshark v2.6.1.0.

3.4 Vulnerability Assessment (AVA)

3.4.1 Vulnerability Survey (AVA_VAN.1)

3.4.1.1 Assurance Activity

Modified in accordance with TD0554.

The evaluator shall generate a report to document their findings with respect to this requirement. This report could physically be part of the overall test report mentioned in ATE_IND, or a separate document. The evaluator performs a search of public information to find vulnerabilities that have been found in similar applications with a particular focus on network protocols the application uses and document formats it parses. The evaluator shall also run a virus scanner with the most current virus definitions against the application files and verify that no files are flagged as malicious.

The evaluator documents the sources consulted and the vulnerabilities found in the report.

For each vulnerability found, the evaluator either provides a rationale with respect to its non-applicability, or the evaluator formulates a test (using the guidelines provided in ATE_IND) to confirm the vulnerability, if suitable. Suitability is determined by assessing the attack vector needed to take advantage of the vulnerability. If exploiting the vulnerability requires expert skills and an electron microscope, for instance, then a test would not be suitable and an appropriate justification would be formulated.

For Windows, Linux, macOS and Solaris: The evaluator shall also run a virus scanner with the most current virus definitions against the application files and verify that no files are flagged as malicious.

The evaluation team performed a search of the National Vulnerability Database (https://nvd.nist.gov/).

The evaluation team initially performed searches on 14 January 2022 and repeated those searches on 8 February 2022. The evaluation team performed final searches, including new search terms suggested by the validators, on 28 February 2022, using the following search terms:

- "axonius"
- "asset management platform"
- "mongodb"
- "openssl 1.0.2"

- “python”
- “tls 1.2”
- “paramiko ssh”
- “esxi 6.5”
- “ubuntu 16.04”
- “docker 19.03”

No vulnerabilities were identified for the TOE.

The evaluator ran a virus scan with up to date virus definitions against the TOE executables and verified that no files were flagged as malicious.

The evaluation team determined that no residual vulnerabilities exist that are exploitable by attackers with Basic Attack Potential.

3.5 Life-Cycle Support (ALC)

3.5.1 Labeling of the TOE (ALC_CMC.1)

3.5.1.1 Assurance Activity

The evaluator shall check the ST to ensure that it contains an identifier (such as a product name/version number) that specifically identifies the version that meets the requirements of the ST. Further, the evaluator shall check the AGD guidance and TOE samples received for testing to ensure that the version number is consistent with that in the ST. If the vendor maintains a web site advertising the TOE, the evaluator shall examine the information on the web site to ensure that the information in the ST is sufficient to distinguish the product.

Section 1.1 of [ST] (“Security Target, TOE and CC Identification”) includes the TOE identification. The TOE is identified in terms of the software included in the evaluated configuration. This consists of Axonius Cybersecurity Asset Management Platform v4.0-f. This is consistent with the version number of the TOE identified in [CCECG] and the version identified by the TOE sample received for testing.

3.5.2 TOE Coverage (ALC_CMS.1)

3.5.2.1 Assurance Activity

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

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

As described in Section 3.5.1 above, the evaluator confirmed the TOE is labelled with unique software version identifiers. Section 6.7 of [ST] ("Protection of the TSF") describes how the TOE uses security features and APIs provided by the Linux platform. This includes data execution protection, AppArmor, and stack-based buffer overflow protection.

3.5.3 Timely Security Update (ALC_TSU_EXT.1)

3.5.3.1 Assurance Activity

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

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

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

Section 6.1 of [ST] ("Timely Security Updates") describes the timely security update process used by the developer to create and deploy TOE security updates. The description encompasses the entirety of the TOE.

Axonius regularly releases new versions of its software which are deployed according to the customer's requirements (whether remotely by Axonius over a secure Internet-based connection, or by Axonius scheduling time for an "offline" upgrade procedure as determined by the organization's requirements, during which the customer will be provided with a secure download link for an updater file, which can be loaded onto the existing Axonius VM via the customer's preferred side channel). Each update contains both operating system package upgrades and bug fixes within Axonius' code for any known security issues addressed since the prior version. Users are made aware of updates by utilizing the "check for update" feature that lets a customer know whether updates are available.

Axonius welcomes security issue reports from anyone – customer or not – via email sent to security@axonius.com, and offers a public PGP key ID, published at https://www.axonius.com/security and reachable only via HTTPS, for the purpose of encrypting these issue reports. Once a report has been received, it is validated by a member of Axonius' Security Team, and prioritized for remediation by the

R&D team according to its technical severity and business impact. If found to be a security issue, the target time for a patch depends on the severity of the issue as follows: Critical – 15 days; High – 30 Days; Medium – 60 days; Low – 90 days. Informational issues have no target time for a patch.

In addition to public reports, Axonius proactively utilizes both manual and automated tooling to attempt to discover issues on their own, and engages with a third-party firm for a penetration test of the TOE on at least an annual basis. Issues discovered via either of these mechanisms are prioritized and remediated according to the same technical severity + business impact system as publicly-reported issues.

Security updates to the TOE are delivered as regular update packages in the same manner as a functional update. The TOE is distributed as a digitally signed .deb file, signed using an ECC (NIST P384) key. The Debian Package Manager utility automatically verifies the signature of the .deb package before proceeding with package installation.