Assurance Activity Report for
Venafi Trust Protection Platform v23.1

Venafi Trust Protection Platform v23.1 Security Target
Version 1.6

Protection Profile for Application Software, Version 1.4
Functional Package for Secure Shell, Version: 1.0

AAR Version 1.2, 07/24/2024

Evaluated by:

2400 Research Blvd, Suite 395
Rockville, MD 20850

Prepared for:

National Information Assurance Partnership
Common Criteria Evaluation and Validation Scheme

The Developer of the TOE:
Venafi

The Author of the Security Target:
Acumen Security, LLC.

The TOE Evaluation was Sponsored by:
Venafi

Evaluation Personnel:
Elliot Keen
Adarsh Pandey

Common Criteria Version

Common Criteria Version 3.1 Revision 5

Common Evaluation Methodology Version

CEM Version 3.1 Revision 5

Revision History

VERSION	DATE	CHANGES
0.1	06/14/2023	Initial Release
0.2	09/11/2023	Added PKG_SSH AAR activities
0.3	10/27/2023	evaluation of AAR EA’s completed
0.4	01/25/2024	Updating as per STv0.6
0.5	01/30/2024	Added Section numbers
0.6	02/20/2024	Updated as per master test plan v1.6
0.7	03/27/2024	Updates to reflect latest test plan v1.7
0.8	05/15/2024	Updated after Peer review
0.9	06/04/2024	Updates as per the ST
1.0	06/17/2024	Updates as per the Test Report
1.1	07/12/2024	Updates based on ECR comments
1.2	07/24/2024	Updated with the latest AVA search

Contents

1 TOE Overview

Venafi Trust Protection Platform is a windows application that secures and protects keys and certificates. This protection improves security posture with increased visibility, threat intelligence, policy enforcement, and faster incident response for certificate-related outages and compromises leveraging misused keys and certificates.

The platform supports all Venafi products and provides native integration with thousands of applications and common APIs for the extensive security ecosystem. Shared and extensible services enable enterprises to gain complete visibility into their key and certificate inventory, identify certificate reputation, and establish a baseline. The entire issuance and renewal process can be automated with policy enforcement and workflows, enabling new encryption dependent applications to be scaled quickly. Trust Protection Platform keeps organizations secure, helping them comply with standards and remediate key and certificate misuse.

The description above provides a general description of the functionality provided by the Venafi Trust Protection Platform. Please see Security Target (ST) sections 1.3.3.1 through 1.3.3.6 for an identification of the evaluated functionality and section 1.3.3.7 for an identification of the functionality that is not covered by the evaluation.

2 Assurance Activities Identification

The Assurance Activities contained within this document include all those defined within the *Protection Profile for Application Software*, Version 1.4, 07 October 2021 [SWAPP] and *Functional Package for Secure Shell*, Version 1.0, 13 May 2021 [SSHFP] based upon the core SFRs and those implemented based on selections within the PPs/FPs.

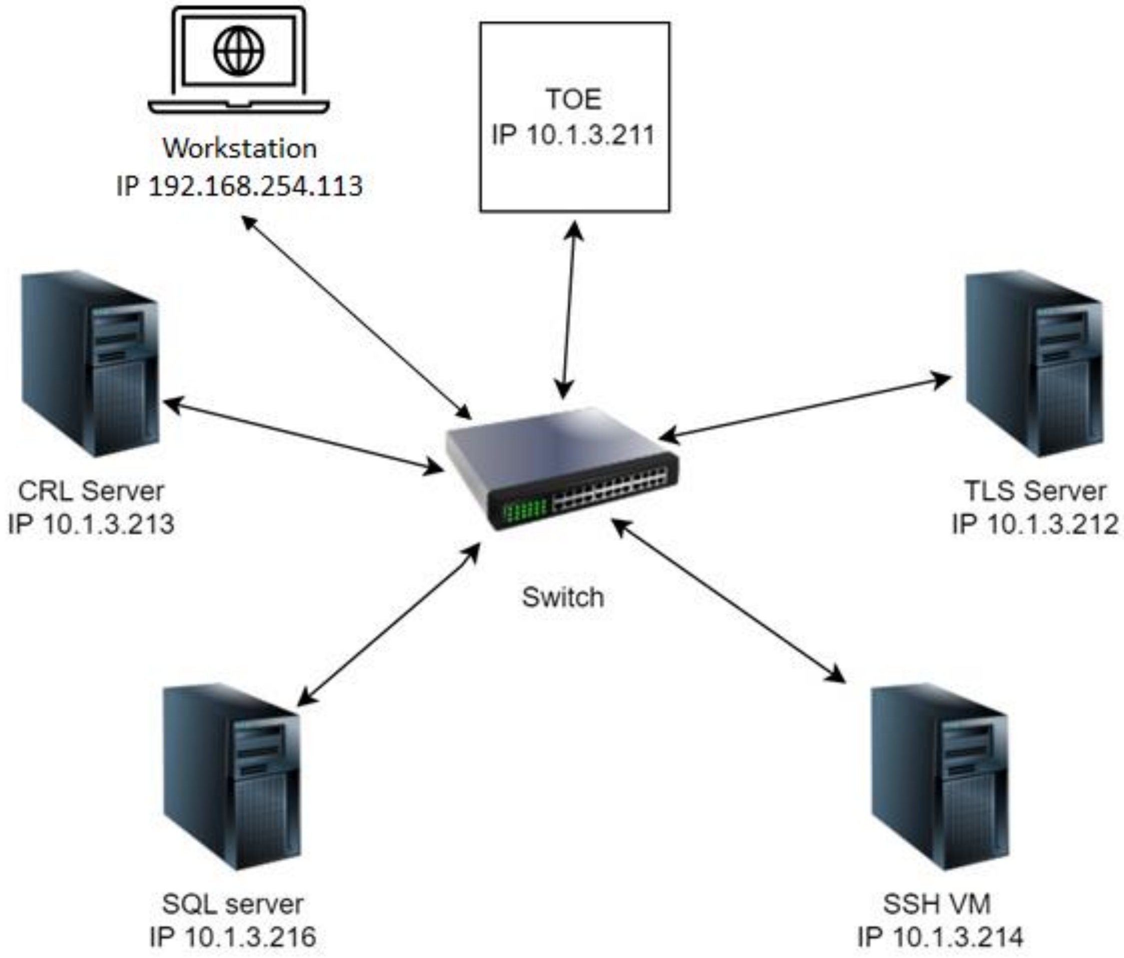
3 Test Equivalency Justification

NA - No equivalency claimed for the TOE.

4 Test Bed Descriptions

4.1 Test Bed

Below is a visual representation of the components included in the test bed:



4.2 Configuration Information

The following table provides configuration information about each device in the test environment.

Device Details		Network Details	System Details		
Role in test environment	**Device Name**	**Protocols**	**OS, including version**	**Timing Source**	**Software & Tools, including version**
Test machine to access the TOE remotely via web-based console	Workstation	TLS, RDP	Windows 10 Enterprise 22H2	Manual	Google Chrome 126.0.6478.61 Remote Desktop Connection 10.0.19041
TOE	VenafiTrust Protection Platform	SSH, TLS	Microsoft Windows Server 2016 Standard Evaluation (10.0.14393 Build 14393)	Manual	Wireshark 4.0.6 (v4.0.6-0-gac2f5a01286a) JetBrains dotPeek 2023.1.3 Google Chrome 117.0.5938.149 XCA 2.4.0 Sysinternals [Accesschk v6.15, VMmap v3.32.0, ProcMon v3.95.0] Notepad ++ v8.3.3 EMET 5.5 Microsoft Binscope 2014 Wumpbin 0.1a Microsoft SQL Server Management Studio 19.0.2 HashMyFiles v2.43
Test VM with SQL server functionality	SQL Server	SSH, TLS	Microsoft Windows Server 2016 Standard Evaluation (10.0.14393 Build 14393)	Manual	Microsoft SQL Server 2022 Developer (64-bit) 16.0.1050.5 Sysinternals VMmap v3.32.0
Test VM with TLS Server functionality	TLS Server	TLS, SSH	Ubuntu 20.04.6 LTS	Manual	OpenSSL 1.1.1f
Test VM with CRL Server functionality	CRL Server	TLS, HTTP, SSH	Ubuntu 20.04.6 LTS	Manual	OpenSSL 1.1.1f

Test VM with SSH Server functionality	SSH Server	SSH	Ubuntu 20.04.6 LTS	Manual	OpenSSH_8.2p1

4.3 Test Time and Location

All testing was carried out at the Acumen Security offices located at 2400 Research Blvd Suite #395, Rockville, MD 20850. Testing occurred from April 2023 to June 2024.

The TOE was in a physically protected, access controlled, designated test lab with no unattended entry/exit ways. At the start of each day, the test bed was verified to ensure that it was not compromised. All evaluation documentation was always kept in a secure repository.

5 Detailed Test Cases (TSS and AGD Activities)

5.1 Mandatory Requirements

5.1.1 Cryptographic Support (FCS)

5.1.1.1 FCS_CKM_EXT.1 Cryptographic Key Generation Services

5.1.1.1.1 FCS_CKM_EXT.1.1 TSS [TD0717]

Objective:

- 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.

Evaluator Findings:

The evaluator reviewed the SFR section in the Security Target and determined that the application needs asymmetric key generation services based on the selection invoke platform-provided functionality for asymmetric key generation

The evaluator reviewed the TSS section 6 and ensured that asymmetric key generation services are needed.

The relevant information is found in the following section(s): TOE Summary Specification FCS_CKM_EXT.1.

Upon investigation, the evaluator found that the TSS states that:

Through .Net the TOE is able to call the underlying Windows cryptographic modules.

The SSH and TLS components of TPP makes use of .NET cryptographic modules for the encryption and decryption of data. SSH itself is provided by Venafi’s maintained internal branch of Maverick, while the TLS protocol is provided by Microsoft .NET alongside TLS cryptography.

The key generation schemes are RSA-based with key sizes of 2048-bits or 3072-bits, or elliptic curve-based with NIST curves, P-256, P-384, or P-521, or FFC schemes using “safe-prime” groups. These schemes are used for both TLS and SSH.

Based on these findings, this assurance activity is considered satisfied.

Verdict:

PASS.

5.1.1.1.2 FCS_CKM_EXT.1.1 AGD

None.

5.1.1.2 FCS_RBG_EXT.1 Random Bit Generation Services

5.1.1.2.1 FCS_RBG_EXT.1 TSS

Objective:

- If "use no DRBG functionality" is selected, the evaluator shall review the TSS to ensure that it needs no random bit generation services.

- If "implement DRBG functionality" is selected, the evaluator shall review the TSS to ensure that additional FCS_RBG_EXT.2 elements are included in the ST.
- If "invoke platform-provided DRBG functionality" is selected, the evaluator shall perform 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.

Evaluator Findings:

- The evaluator reviewed the TSS section 6 and verified that, "invoke platform-provided DRBG functionality" is selected, and the following activities are performed:

– The TSS identifies all functions (as described by the SFRs included in the ST) that obtain random numbers from the platform RBG.

– For each of these functions, the TSS states which platform interface (API) is used to obtain the random numbers.

– Each of these interfaces corresponds to the acceptable interfaces listed for each platform below.

The relevant information is found in the following section(s): TOE Summary Specification FCS_RBG_EXT.1.

Upon investigation, the evaluator found that the TSS states that:

The TOE invokes platform provided DRBG (SecureRNG) for the purpose of generating a salt value, which is used to protect the Windows Data Protection API (DPAPI) key. The TOE invokes underlying platform's SecureRNG using the System.Security.Cryptography.RandomNumberGenerator class, which uses BCryptGenRandom. All random numbers used by the TLS and SSH SFR related functions are used by the platform's underlying cryptographic functionality indirectly.

Based on these findings, this assurance activity is considered satisfied.

Verdict:

PASS.

5.1.1.2.2 FCS_RBG_EXT.1 AGD

None.

5.1.1.3 FCS_STO_EXT.1 Storage of Credentials

5.1.1.3.1 FCS_STO_EXT.1.1 TSS

Objective:

- 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.

Evaluator Findings:

- The evaluator reviewed the TSS section 6 and ensured that it lists all persistent credentials (secret keys, PKI private keys, or passwords) needed to meet the requirements in the ST.
- The evaluator reviewed the TSS section 6 and ensured that it lists for what purpose it is used, and how it is stored.

The relevant information is found in the following section(s): TOE Summary Specification FCS_STO_EXT.1.

Upon investigation, the evaluator found that the TSS states that:

The TOE relies on the platform to securely store the following:

- DSN key - related to database connection data - the primary use case is in installation answer files that refer to DNS keys to connect to the existing TOE database.
- PKCS12 key - related to the product capability to export certificates and their private keys. This format is an option when certificate data is exported.
- PKCS8 (private key) - related to the product capability to export certificates and their private keys. This format is an option when certificate data is exported.
- Usernames, Passwords, and Customer application credentials are data types stored as hashes:
  - Application credentials are for certificate and key management of devices managed by the TOE - stored in a hashed format.
  - Usernames and passwords are used both for credentials for access to the product, and for other products as application credentials. Best practices are followed in all cases and sensitive data is stored not in plain text and always as salted hashes.

All certificates are stored in the Windows Certificate Store. The Windows Registry is used for storage of the TOE's symmetric key. An AES 256 key is used for the encryption and decryption of secrets. It is protected by the Windows Data Protection API (DPAPI). All usernames, passwords, and credentials are protected by the Windows DPAPI.

Based on these findings, this assurance activity is considered satisfied.

Verdict:

PASS.

5.1.1.3.2 FCS_STO_EXT.1 AGD

None.

5.1.1.4 FCS_SSH_EXT.1 SSH Protocol

5.1.1.4.1 FCS_SSH_EXT.1.1 TSS

Objective:

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

Evaluator Findings:

The evaluator reviewed the TSS section 6 and ensured that the selections indicated in the ST are consistent with selections in this and subsequent components.

The relevant information is found in the following section(s): TOE Summary Specification FCS_SSH_EXT.1.

Upon investigation, the evaluator found that the TSS states that:

The TOE functions as an SSH client in order to communicate with target applications and certificate authorities.

The TOE implements SSH acting as a client in accordance RFCs 4251, 4252, 4253, 4254, 4344, 5656, 6668, 8268, 8308, 8332, 8709, 8731.

Based on these findings, this assurance activity is considered satisfied.

Verdict:

PASS.

5.1.1.4.2 FCS_SSH_EXT.1.1 AGD

None.

5.1.1.4.3 FCS_SSH_EXT.1.2 TSS

Objective:

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

Evaluator Findings:

The evaluator reviewed the TSS section 6 and ensured that the authentication methods listed in the TSS are identical to those listed in this SFR component; and ensured that password-based authentication methods have been selected in the ST, these are also described. The keyboard-interactive method was not selected.

The relevant information is found in the following section(s): TOE Summary Specification FCS_SSH_EXT.1.

Upon investigation, the evaluator found that the TSS states that:

Both public-key and password-based authentication are supported.

SSH-RSA, RSA-SHA2-256, RSA-SHA2-512, ECDSA-SHA2-NISTp256, ECDSA-SHA2-NISTp384, ECDSA-SHA2-NISTp521 and SSH-ED25519 are the supported public key algorithms.

Based on these findings, this assurance activity is considered satisfied.

Verdict:

PASS.

5.1.1.4.4 FCS_SSH_EXT.1.2 AGD

Objective:

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

Evaluator Findings:

The evaluator checked the AGD and ensured that the configuration options, for authentication mechanisms provided by the TOE are described.

Upon investigation, the evaluator found that the section 4.1 titled “SSH Connectivity” in AGD activity states that:

The TOE supports password-based authentication and public key-based authentication. Configuration steps are mentioned below.

- For password based-authentication the user must configure the credentials from the Web console (Aperture):
    - Select Platform from 3x3 square near the top right of Aperture interface.
    - Select Policy tree.
    - Click on Add under Policy dropdown.

- For Public key based authentication, the user must upload the private key on the Web console(Aperture):
  - Select Platform from 3x3 square near top right of Aperture interface.
  - Select Policy tree.
  - Click on Add under Policy dropdown.
  - Select Private Key Credential under credential tab.
  - Private key can be uploaded by accessing the "Upload Private key" tab.

Note: The following algorithms are not User configurable.

The TOE supports the use of following public key algorithms:

- ssh-rsa (RFC 4253),
- rsa-sha2-256 (RFC 8332),
- rsa-sha2-512 (RFC 8332),
- ecdsa-sha2-nistp256 (RFC 5656),
- ecdsa-sha2-nistp384 (RFC 5656),
- ecdsa-sha2-nistp521 (RFC 5656), and
- ssh-ed25519 (RFC 8709).

Based on these findings, this assurance activity is considered satisfied.

Verdict:

PASS.

5.1.1.4.5 FCS_SSH_EXT.1.3 TSS

Objective:

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

Evaluator Findings:

The evaluator reviewed theTSS section 6 and ensured that it describes how "large packets" are detected and handled.

The relevant information is found in the following section(s): 6 TOE Summary Specification FCS_SSH_EXT.1.

Upon investigation, the evaluator found that the TSS states that:

TOE examines the packet_length field to determine whether the packet is a large packet or not. If the TOE receives an SSH packet larger than 35,000 bytes the packet is dropped and the SSH connection is closed.

Based on these findings, this assurance activity is considered satisfied.

Verdict:

PASS.

5.1.1.4.6 FCS_SSH_EXT.1.3 AGD

None.

5.1.1.4.7 FCS_SSH_EXT.1.4 TSS

Objective:

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

Evaluator Findings:

- The evaluator reviewed the TSS section 6 and ensured that the encryption algorithms supported are specified.
- The evaluator reviewed the TSS section 6 and ensured that the encryption algorithms specified are identical to those listed for this component.

The relevant information is found in the following section(s): TOE Summary Specification FCS_SSH_EXT.1.

Upon investigation, the evaluator found that the TSS states that:

The following SSH transport algorithms may be used:

- AES128-CBC
- AES256-CBC
- AES128-CTR
- AES256-CTR

Based on these findings, this assurance activity is considered satisfied.

Verdict:

PASS.

5.1.1.4.8 FCS_SSH_EXT.1.4 AGD

Objective:

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

Evaluator Findings:

The evaluator checked the AGD and ensured that it contains instructions to the administrator on how to ensure that only the allowed mechanisms are used in SSH connections with the TOE.

Upon investigation, the evaluator found in the section 4.1 titled “SSH Connectivity” of the AGD states that:

Note: The following algorithms are not User configurable.
The TOE supports the use of following encryption algorithms:

- aes128-ctr
- aes256-ctr
- aes128-cbc
- aes256-cbc

Also, the evaluator found in the section 2 titled “Platform Configuration” and section 2.1 titled “Common Criteria Configuration” of the AGD states that:

Because the product relies on the underlying cryptographic functionality of the Windows Server platform, Windows Server 2016 must be in FIPS mode to restrict its ciphers to Common Criteria requirements. This can be done through the Local Security policy configuration as follows:

- Open Local Security Policy
- Security Settings > Local Policies > Security Options
- Change the security setting for “System Cryptography: Use FIPS compliant algorithms for encryption, hashing, and signing” to Enabled

To configure the TOE to only use Common Criteria-compliant SSH algorithms, the following registry key needs to be added in registry at HKEY LOCAL MACHINE\SOFTWARE\Venafi\Platform .

Administrator can either create a new dword key as follows directly in the registry at HKEY LOCAL MACHINE\SOFTWARE\Venafi\Platform, or save the following in a plain text file (using a program such as Notepad), save it as a .reg file, and double click to apply it.

```
--------------------------------------------------------------------------------
Windows Registry Editor Version 5.00

[HKEY_LOCAL_MACHINE\SOFTWARE\Venafi\Platform]
"Common Criteria Compliant"=dword:00000001
--------------------------------------------------------------------------------
```

Based on these findings, this assurance activity is considered satisfied.

Verdict:

PASS.

5.1.1.4.9 FCS_SSH_EXT.1.5 TSS

Objective:

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

Evaluator Findings:

- The evaluator reviewed the TSS section 6 and ensured that the hashing algorithms supported are specified.
- The evaluator reviewed the TSS section 6 and ensured that the hashing algorithms specified are identical to those listed for this component.

The relevant information is found in the following section(s): TOE Summary Specification FCS_SSH_EXT.1.

Upon investigation, the evaluator found that the TSS states that:

HMAC-SHA2-256 and HMAC-SHA2-512 may be used for data integrity.

Based on these findings, this assurance activity is considered satisfied.

Verdict:

PASS.

5.1.1.4.10 FCS_SSH_EXT.1.5 AGD

Objective:

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

Evaluator Findings:

The evaluator checked the AGD and ensured that it contains instructions to the administrator on how to ensure that only the allowed mechanisms are used in SSH connections with the TOE.

Upon investigation, the evaluator found in the section 4.1 titled "SSH Connectivity" of the AGD states that:

Note: The following algorithms are not User configurable.

The TOE supports the use of following MAC algorithms (data integrity):

- hmac-sha2-256
- hmac-sha2-512

Also, the evaluator found that in the section 2 titled “Platform Configuration” and section 2.1 titled “Common Criteria Configuration” of the AGD states that:

Because the product relies on the underlying cryptographic functionality of the Windows Server platform, Windows Server 2016 must be in FIPS mode to restrict its ciphers to Common Criteria requirements. This can be done through the Local Security policy configuration as follows:

- Open Local Security Policy
- Security Settings > Local Policies > Security Options
- Change the security setting for “System Cryptography: Use FIPS compliant algorithms for encryption, hashing, and signing” to Enabled

To configure the TOE to only use Common Criteria-compliant SSH algorithms, the following registry key needs to be added in registry at HKEY LOCAL MACHINE\SOFTWARE\Venafi\Platform .
Administrator can either create a new dword key as follows directly in the registry at HKEY LOCAL MACHINE\SOFTWARE\Venafi\Platform, or save the following in a plain text file (using a program such as Notepad), save it as a .reg file, and double click to apply it.

```
--------------------------------------------------------------------------------
Windows Registry Editor Version 5.00

[HKEY_LOCAL_MACHINE\SOFTWARE\Venafi\Platform]
"Common Criteria Compliant"=dword:00000001
--------------------------------------------------------------------------------
```

Based on these findings, this assurance activity is considered satisfied.

Verdict:

PASS.

5.1.1.4.11 FCS_SSH_EXT.1.6 TSS

Objective:

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

Evaluator Findings:

- The evaluator reviewed the TSS section 6 and ensured that the shared secret establishment algorithms supported are specified.
- The evaluator reviewed the TSS section 6 and ensured that the shared secret establishment algorithms specified are identical to those listed for this component.

The relevant information is found in the following section(s): TOE Summary Specification FCS_SSH_EXT.1.

Upon investigation, the evaluator found that the TSS states that:

The TOE supports following algorithms to establish shared secret with its peer:

- diffie-hellman-group14-sha256,
- diffie-hellman-group16-sha512,
- diffie-hellman-group18-sha512,
- ecdh-sha2-nistp256,

- ecdh-sha2-nistp384,
- ecdh-sha2-nistp521,
- curve25519-sha256

Based on these findings, this assurance activity is considered satisfied.

Verdict:

PASS.

5.1.1.4.12 FCS_SSH_EXT.1.6 AGD

Objective:

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

Evaluator Findings:

The evaluator checked the AGD and ensured that it contains instructions to the administrator on how to ensure that only the allowed mechanisms are used in SSH connections with the TOE.

Upon investigation, the evaluator found that the section 4.1 titled “SSH Connectivity” of the AGD states that:

Note: The following algorithms are not User configurable.

The TOE supports the use of following Key Exchange method:

- diffie-hellman-group14-sha256 (RFC 8268)
- diffie-hellman-group16-sha512 (RFC 8268)
- diffie-hellman-group18-sha512 (RFC 8268)
- ecdh-sha2-nistp256 (RFC 5656)
- ecdh-sha2-nistp384 (RFC 5656)
- ecdh-sha2-nistp521 (RFC 5656)
- curve25519-sha256 (RFC 8731)

Also, the evaluator found that in the section 2 titled “Platform Configuration” and section 2.1 titled “Common Criteria Configuration” of the AGD states that:

Because the product relies on the underlying cryptographic functionality of the Windows Server platform, Windows Server 2016 must be in FIPS mode to restrict its ciphers to Common Criteria requirements. This can be done through the Local Security policy configuration as follows:

- Open Local Security Policy
- Security Settings > Local Policies > Security Options
- Change the security setting for “System Cryptography: Use FIPS compliant algorithms for encryption, hashing, and signing” to Enabled

To configure the TOE to only use Common Criteria-compliant SSH algorithms, the following registry key needs to be added in registry at HKEY LOCAL MACHINE\SOFTWARE\Venafi\Platform .
Administrator can either create a new dword key as follows directly in the registry at HKEY LOCAL MACHINE\SOFTWARE\Venafi\Platform, or save the following in a plain text file (using a program such as Notepad), save it as a .reg file, and double click to apply it.

```
-----------------------------------------------------------------------------------
Windows Registry Editor Version 5.00
```

```
[HKEY_LOCAL_MACHINE\SOFTWARE\Venafi\Platform]
"Common Criteria Compliant"=dword:00000001
--------------------------------------------------------------------------------
```

Based on these findings, this assurance activity is considered satisfied.

Verdict:

PASS.

5.1.1.4.13 FCS_SSH_EXT.1.7 TSS

Objective:

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

Evaluator Findings:

- The evaluator reviewed the TSS section 6 and ensured that the KDFs supported are specified.
- The evaluator reviewed the TSS section 6 and ensured that the KDFs specified are identical to those listed for this component.

The relevant information is found in the following section(s): TOE Summary Specification FCS_SSH_EXT.1.

Upon investigation, the evaluator found that the TSS states that:

The TOE supports SSH KDF as per RFC 4253 (Section 7.2) and RFC 5656 (Section 4).

Based on these findings, this assurance activity is considered satisfied.

Verdict:

PASS.

5.1.1.4.14 FCS_SSH_EXT.1.7 AGD

None.

5.1.1.4.15 FCS_SSH_EXT.1.8 TSS

Objective:

- The evaluator shall check the TSS to ensure that if the TOE enforces connection rekey or termination limits lower than the maximum values that these lower limits are identified.
- In cases where hardware limitation will prevent reaching data transfer threshold in less than one hour, the evaluator shall check the TSS to ensure it contains:

a. An argument describing this hardware-based limitation and

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

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

Evaluator Findings:

- The evaluator reviewed the TSS section 6 and ensured that the TOE enforces connection rekey or termination limits lower than the maximum values, these lower limits are identified.

The relevant information is found in the following section(s): TOE Summary Specification FCS_SSH_EXT.1.

Upon investigation, the evaluator found that the TSS states that:

The TOE enforces SSH session rekey if the connection time exceeds one hour or the data transfer (sent/received) exceed one gigabyte.

Based on these findings, this assurance activity is considered satisfied.

Verdict:

PASS.

5.1.1.4.16 FCS_SSH_EXT.1.8 AGD

Objective:

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

Evaluator Findings:

The evaluator checked the AGD and ensured that, if the connection rekey or termination limits are configurable, it contains instructions to the administrator on how to configure the relevant connection rekey or termination limits for the TOE.

Upon investigation, the evaluator found that the section 4.1 titled "SSH Connectivity" in the AGD activity states that:

Note: SSH session rekey limits are not User configurable.

Also, the evaluator found that in the section 2 titled "Platform Configuration" and section 2.1 titled "Common Criteria Configuration" of the AGD states that:

Because the product relies on the underlying cryptographic functionality of the Windows Server platform, Windows Server 2016 must be in FIPS mode to restrict its ciphers to Common Criteria requirements. This can be done through the Local Security policy configuration as follows:

- Open Local Security Policy
- Security Settings > Local Policies > Security Options
- Change the security setting for "System Cryptography: Use FIPS compliant algorithms for encryption, hashing, and signing" to Enabled

To configure the TOE to only use Common Criteria-compliant SSH algorithms, the following registry key needs to be added in registry at HKEY LOCAL MACHINE\SOFTWARE\Venafi\Platform .
Administrator can either create a new dword key as follows directly in the registry at HKEY LOCAL MACHINE\SOFTWARE\Venafi\Platform, or save the following in a plain text file (using a program such as Notepad), save it as a .reg file, and double click to apply it.

```
------------------------------------------------------------------------------------
Windows Registry Editor Version 5.00

[HKEY_LOCAL_MACHINE\SOFTWARE\Venafi\Platform]
"Common Criteria Compliant"=dword:00000001
------------------------------------------------------------------------------------
```

Based on these findings, this assurance activity is considered satisfied.

Verdict:

PASS.

5.1.1.5 FCS_SSHC_EXT.1 SSH Protocol - Client

5.1.1.5.1 FCS_SSHC_EXT.1.1 TSS

None.

5.1.1.5.2 FCS_SSHC_EXT.1.1 AGD

Objective:

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

Evaluator Findings:

The evaluator checked the AGD and ensured that it contains instructions to the administrator on how to ensure that only the allowed mechanisms are used in SSH connections with the TOE.

Upon investigation, the evaluator found that the section 4.1 titled "SSH Connectivity" and 4.2 "Adding/Trusting an SSH Server Hostkey to the TOE" in the AGD activity states that:

The following algorithms are not User configurable.

Note: The following algorithms are not User configurable.

The TOE supports the use of following public key algorithms to authenticate its peer (SSH server) host:

- ssh-rsa (RFC 4253)
- rsa-sha2-256 (RFC 8332)
- rsa-sha2-512 (RFC 8332)
- ecdsa-sha2-nistp256 (RFC 5656)
- ecdsa-sha2-nistp384 (RFC 5656)
- ecdsa-sha2-nistp521 (RFC 5656)
- ssh-ed25519 (RFC 8709)

Also, the evaluator found in the section 2 titled "Platform Configuration" and section 2.1 titled "Common Criteria Configuration" of the AGD states that:

Because the product relies on the underlying cryptographic functionality of the Windows Server platform, Windows Server 2016 must be in FIPS mode to restrict its ciphers to Common Criteria requirements. This can be done through the Local Security policy configuration as follows:

- Open Local Security Policy
- Security Settings > Local Policies > Security Options
- Change the security setting for "System Cryptography: Use FIPS compliant algorithms for encryption, hashing, and signing" to Enabled

To configure the TOE to only use Common Criteria-compliant SSH algorithms, the following registry key needs to be added in registry at HKEY LOCAL MACHINE\SOFTWARE\Venafi\Platform .
Administrator can either create a new dword key as follows directly in the registry at HKEY LOCAL MACHINE\SOFTWARE\Venafi\Platform, or save the following in a plain text file (using a program such as Notepad), save it as a .reg file, and double click to apply it.

```
---------------------------------------------------------------------------------
Windows Registry Editor Version 5.00

[HKEY_LOCAL_MACHINE\SOFTWARE\Venafi\Platform]
```

```
"Common Criteria Compliant"=dword:00000001
--------------------------------------------------------------------------------
```

Based on these findings, this assurance activity is considered satisfied.

Verdict:

PASS.

5.1.2 User Data Protection (FDP)

5.1.2.1 FDP_DEC_EXT.1 Access to Platform Resources

5.1.2.1.1 FDP_DEC_EXT.1.1 TSS

None.

5.1.2.1.2 FDP_DEC_EXT.1.1 AGD

Objective:

- 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.

Evaluator Findings:

- The evaluator checked the AGD and ensured that the application has access to hardware resources.
- The evaluator checked the AGD and ensured that it is consistent with the selections indicated.
- The evaluator checked the AGD and ensured that it justifies as to why access is required.

Upon investigation, the evaluator found in the section 4.5 titled “TOE Access to Platform Resources” in the AGD states that:

Network connectivity is the only platform hardware resource accessed by the TOE. The TOE leverages Microsoft IIS webserver for User or Admin authentication over web-based console. The TOE also communicates with an external database, managed hosts, CA servers, and to perform discovery services.

Based on these findings, this assurance activity is considered satisfied.

Verdict:

PASS.

5.1.2.1.3 FDP_DEC_EXT.1.2 TSS

None.

5.1.2.1.4 FDP_DEC_EXT.1.2 AGD

Objective:

- 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.

Evaluator Findings:

- The evaluator checked the AGD and ensured that the application can access sensitive information repositories.
- The evaluator checked the AGD and ensured that it is consistent with the selections indicated.
- The evaluator checked the AGD and ensured that it justifies as to why access is required.

Upon investigation, the evaluator found in the section 4.5 titled “TOE Access to Platform Resources” in the AGD states that:

System logs are the only sensitive information repository accessed by the TOE. The TOE accesses system logs (i.e. Windows Event log) for the purpose of writing events to the logs.

Based on these findings, this assurance activity is considered satisfied.

Verdict:
PASS.

5.1.2.2 FDP_NET_EXT.1 Network Communications

5.1.2.2.1 FDP_NET_EXT.1 TSS

None.

5.1.2.2.2 FDP_NET_EXT.1 AGD

None.

5.1.2.3 FDP_DAR_EXT.1 Encryption Of Sensitive Application Data

5.1.2.3.1 FDP_DAR_EXT.1.1 TSS

Objective:

- 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.
- 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.

Evaluator Findings:

- The evaluator reviewed the TSS section 6 and ensured that it describes the sensitive data processed by the application.
- The evaluator reviewed the TSS section 6 and ensured that the activities cover all of the sensitive data identified in the TSS.
- “not store any sensitive data” is not selected.

The relevant information is found in the following section(s): TOE Summary Specification FDP_DAR_EXT.1.

Upon investigation, the evaluator found that the TSS states that:

The only sensitive data stored by the TOE in non-volatile memory is listed in FCS_STO_EXT.1. No additional sensitive data is stored by the TOE.

Based on these findings, this assurance activity is considered satisfied.

Verdict:

PASS.

5.1.2.3.2 FDP_DAR_EXT.1 AGD

None.

5.1.3 Security Management (FMT)

5.1.3.1 FMT_MEC_EXT.1 Supported Configuration Mechanism

5.1.3.1.1 FMT_MEC_EXT.1.1 TSS

Objective:

- 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 AGD in response to an SFR.
- 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.

Evaluator Findings:

- The evaluator reviewed the TSS section 6 and ensured that it identifies 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.

The relevant information is found in the following section(s): TOE Summary Specification FMT_MEC_EXT.1.

Upon investigation, the evaluator found that the TSS states that:

As a Windows application, the TOE utilizes both the Windows Registry and C:\ProgramData\ directory, where changes can be observed when configuration changes are made to the TOE.

Furthermore, the ST does not claim FDP_PRT_EXT.1.

- The evaluator reviewed the TSS section 6 and ensured that it identifies a list of settings related to any SFRs and any settings that are mandated in the AGD in response to an SFR.

The relevant information is found in the following section(s): TOE Summary Specification FMT_MEC_EXT.1.

Upon investigation, the evaluator found that the TSS states that:

The TOE supports following configuration settings related to any SFRs and any settings that are mandated in the AGD:

- all management functions covered under FMT_SMF.1
- database configuration

- managed host configuration
- discovery services configuration
- certificate verification configuration
- CRL check configuration

The evaluator reviewed the relevant sections in the AGD and ensured that the AGD mandated settings are included in the TSS.

Based on these findings, this assurance activity is considered satisfied.

Verdict:

PASS.

5.1.3.1.2 FMT_MEC_EXT.1 AGD

None.

5.1.3.2 FMT_CFG_EXT.1 Secure by Default Configuration

5.1.3.2.1 FMT_CFG_EXT.1.1 TSS

Objective:

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

Evaluator Findings:

The evaluator reviewed the TSS section 6 and ensured that, if the application requires any type of credentials and if the application installs with default credentials.

The relevant information is found in the following section(s): TOE Summary Specification FMT_CFG_EXT.1.

Upon investigation, the evaluator found that the TSS states that:

There are no default credentials within the TOE. Upon installation the TOE generates a GUID for the base configuration of the system. A master administrator role (admin) is created and the password for this account is defined as part of the installation..

Based on these findings, this assurance activity is considered satisfied.

Verdict:

PASS.

5.1.3.2.2 FMT_CFG_EXT.1.1 AGD

None.

5.1.3.2.3 FMT_CFG_EXT.1.2 TSS

None.

5.1.3.2.4 FMT_CFG_EXT.1.2 AGD

None.

5.1.3.3 FMT_SMF.1 Specification of Management Functions

5.1.3.3.1 FMT_SMF.1 TSS

None.

5.1.3.3.2 FMT_SMF.1.1 AGD

Objective:

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

Evaluator Findings:

The evaluator checked the AGD and ensured that every management function mandated by the PP is described in the AGD and that the description contains the information required to perform the management duties associated with the management function.

Upon investigation, the evaluator found the section 4.6 titled “Management Functions” in the AGD includes every management function mandated by the PP and selected in the ST/TSS.

Additionally, sub-sections 4.6.1 to 4.6.4 provide all details required to perform the management duties associated with the management function.

Based on these findings, this assurance activity is considered satisfied.

Verdict:

PASS.

5.1.4 Privacy (FPR)

5.1.4.1 FPR_ANO_EXT.1 User Consent for Transmission of Personally Identifiable Information

5.1.4.1.1 FPR_ANO_EXT.1.1 TSS

Objective:

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

Evaluator Findings:

The evaluator reviewed the TSS section 6 and ensured that it identifies functionality in the application if and where PII can be transmitted.

The relevant information is found in the following section(s): TOE Summary Specification FPR_ANO_EXT.1.

Upon investigation, the evaluator found that the TSS states that:

The TOE does not transmit any PII.

Based on these findings, this assurance activity is considered satisfied.

Verdict:

PASS.

5.1.4.1.2 FPR_ANO_EXT.1.1 AGD

None.

5.1.5 Protection of the TSF (FPT)

5.1.5.1 FPT_API_EXT.1 Use of Supported Services and APIs

5.1.5.1.1 FPT_API_EXT.1.1 TSS

Objective:

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

Evaluator Findings:

The evaluator reviewed the TSS section 6 and ensured that it lists the platform APIs used in the application.

The relevant information is found in the following section(s): TOE Summary Specification FPT_API_EXT.1.

Upon investigation, the evaluator found that the TSS states that:

Microsoft .Net 4.7.2 is used by the TOE, which are listed here: https://learn.microsoft.com/en-us/dotnet/api/?view=netframework-4.7.2

Based on these findings, this assurance activity is considered satisfied.

Verdict:

PASS.

5.1.5.1.2 FPT_API_EXT.1 AGD

None.

5.1.5.2 FPT_AEX_EXT.1 Anti-Exploitation Capabilities

5.1.5.2.1 FPT_AEX_EXT.1.1 TSS [TD0798]

Objective:

The evaluator shall ensure that the TSS describes the compiler flags used to enable ASLR when the application is compiled. If any explicitly-mapped exceptions are claimed, the evaluator shall check that the TSS identifies these exceptions, describes the static memory mapping that is used, and provides justification for why static memory mapping is appropriate in this case.

Evaluator Findings:

The evaluator reviewed the TSS section 6 and ensured that it describes the compiler flags used to enable ASLR when the application is compiled.

The relevant information is found in the following section(s): TOE Summary Specification FPT_AEX_EXT.1.

Upon investigation, the evaluator found that the TSS states that:

The TOE never maps memory to explicit addresses, nor does it allocate memory regions with write and execute permissions.

It is not necessary to use compiler flags to enable ASLR. This is done by default. The TOE’s code is not run natively, but instead as managed code on top of Microsoft’s .Net.

Based on these findings, this assurance activity is considered satisfied.

Verdict:

PASS.

5.1.5.2.2 FPT_AEX_EXT.1.1 AGD

None.

5.1.5.2.3 FPT_AEX_EXT.1.2 TSS

None.

5.1.5.2.4 FPT_AEX_EXT.1.2 AGD

None.

5.1.5.2.5 FPT_AEX_EXT.1.3 TSS

None.

5.1.5.2.6 FPT_AEX_EXT.1.3 AGD

None.

5.1.5.2.7 FPT_AEX_EXT.1.4 TSS

None.

5.1.5.2.8 FPT_AEX_EXT.1.4 AGD

None.

5.1.5.2.9 FPT_AEX_EXT.1.5 TSS [TD0815]

Objective:

(Conditional: The PE or ELF automated tests fail) The evaluator shall ensure that the TSS describes the stack-based buffer overflow compiler flags.

Evaluator Findings:

The evaluator reviewed the TSS section 6 and ensured that it describes the stack-based buffer overflow compiler flags.

The relevant information is found in the following section(s): TOE Summary Specification FPT_AEX_EXT.1.

Upon investigation, the evaluator found that the TSS states that:

The TOE's code is not run natively, but instead as managed code on top of Microsoft's .Net.

Similarly, the use of a managed code base means that compiler flags aren't used for stack-based buffer overflow protection. Stack Based buffer overflows are protected in managed code by an exception being thrown by the CLR rather than having the overflow happen on the stack.

Based on these findings, this assurance activity is considered satisfied.

Verdict:
PASS.

5.1.5.2.10 FPT_AEX_EXT.1.5 AGD

None.

5.1.5.3 FPT_IDV_EXT.1 Software Identification and Versions

5.1.5.3.1 FPT_IDV_EXT.1.1 TSS

Objective:

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

Evaluator Findings:

The evaluator reviewed the TSS section 6 and ensured that if "other version information" is selected, the TSS contains an explanation of the versioning methodology.

The relevant information is found in the following section(s): TOE Summary Specification FPT_IDV_EXT.1.

Upon investigation, the evaluator found that the TSS states that:

The application version follows a major.minor.patch.build structure. The build corresponds to a git tag for that particular build.

Based on these findings, this assurance activity is considered satisfied.

Verdict:

PASS.

5.1.5.3.2 FPT_IDV_EXT.1 AGD

None.

5.1.5.4 FPT_LIB_EXT.1 Use of Third Party Libraries

5.1.5.4.1 FPT_LIB_EXT.1 TSS

None.

5.1.5.4.2 FPT_LIB_EXT.1 AGD

None.

5.1.5.5 FPT_TUD_EXT.1 Integrity for Installation and Update

5.1.5.5.1 FPT_TUD_EXT.1.1 TSS

None.

5.1.5.5.2 FPT_TUD_EXT.1.1 AGD

Objective:

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

Evaluator Findings:

The evaluator checked the AGD and ensured that it includes a description of how updates are performed.

Upon investigation, the evaluator found that the section 7 titled “TOE Updates” in the AGD states that:

To Upgrade Trust Protection Platform:

1. Back up the Trust Protection Platform database.
2. Stop IIS.
3. Stop all Trust Protection Platform services.
    a. Stop the Trust Protection Platform service.
    b. Stop the Venafi UniCERT Interface service, if present.
    c. Stop the Venafi Log Server service.
4. Close all Venafi-related Windows applications. For example, close any browsers that are logged into User Portal, Aperture, or the Web Administration console.
5. Repeat steps 2 and 3 on all Trust Protection Platform servers to ensure all Venafi-related services are stopped prior to continuing with step 6.

6. Unzip "Venafi Trust Protection Platform 23.1.1.zip". Run the VenafiTPPInstall-23.1.1.msi as an Administrator (e.g. launch the Command Prompt with "Run as Administrator" and then launch the installation MSI file). Complete the on-screen walkthroughs, per your environment's requirements.

   Note:

   - All binaries are signed using signtool.exe, which is a .Net framework tool for digital file signatures, to ensure they come from the authorized source – via download https://download.venafi.com/. Users must have a username and password to login to download the binaries.

7. Query the Trust Protection Platform:

   a. Open Venafi Trust Protection Platform

   b. Click on Help

   c. Click on About Console

   d. Check the Version and confirm update successful.

Based on these findings, this assurance activity is considered satisfied.

Verdict:
PASS.

5.1.5.5.3 FPT_TUD_EXT.1.2 TSS

None.

5.1.5.5.4 FPT_TUD_EXT.1.2 AGD

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

Evaluator Findings:
The evaluator checked the AGD and ensured that it includes a description of how to query the current version of the application.

Upon investigation, the evaluator found that the section 7 titled "TOE Updates" in the AGD states that:

- To query the current version of the TOE from Trust Protection Platform:
  - Open Venafi Trust Protection Platform.
  - Click on Help.
  - Click on About Console.
  - Check the Version of the TOE.

Based on these findings, this assurance activity is considered satisfied.

Verdict:
PASS.

5.1.5.5.5 FPT_TUD_EXT.1.3 TSS

None.

5.1.5.5.6 FPT_TUD_EXT.1.3 AGD

None.

5.1.5.5.7 FPT_TUD_EXT.1.4 TSS

Objective:

- The evaluator shall verify that the TSS identifies how updates to 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 AGD) describes how candidate updates are obtained.

Evaluator Findings:

- The evaluator reviewed the TSS section 6 and ensured that it identifies how updates to the application are signed by an authorized source. The definition of an authorized source must be contained in the TSS.
- The evaluator reviewed the TSS section 6 and ensured that it describes how candidate updates are obtained.

The relevant information is found in the following section(s): TOE Summary Specification FPT_TUD_EXT.1.

Upon investigation, the evaluator found that the TSS states that:

All binaries are signed by Venafi using signtool.exe, which is a .Net framework tool for digital file signatures. Venafi is the only authorized source to sign the executable binary. Authorized source can be verified by right-clicking the .MSI file and select Properties. Under Digital Signatures Tab, Name of signer will indicate “Venafi, Inc”.
Additionally, ensure that the binaries are downloaded from the authorized source – via https://download.venafi.com/. Users must have a username and password to login to download the binaries.

Based on these findings, this assurance activity is considered satisfied.

Verdict:

PASS.

5.1.5.5.8 FPT_TUD_EXT.1.4 AGD

None.

5.1.5.5.9 FPT_TUD_EXT.1.5 TSS

Objective:

- 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.

Evaluator Findings:

- The evaluator reviewed the TSS section 6 and ensured that it identifies how the application is distributed.
- The evaluator also reviewed the ST selections and found that, " as an additional software package to the platform OS " is selected.

The relevant information is found in the following section(s): TOE Summary Specification FPT_TUD_EXT.1.

Upon investigation, the evaluator found that the TSS states that:

The application is distributed as an additional software package to the platform OS. Updates to the TOE are distributed as .MSI installation files and are performed in the same manner as a product installation.

Based on these findings, this assurance activity is considered satisfied.

Verdict:

PASS.

5.1.5.5.10 FPT_TUD_EXT.1.5 AGD

None.

5.1.6 Trusted Path/Channels (FTP)

5.1.6.1 FTP_DIT_EXT.1 Protection of Data in Transit

5.1.6.1.1 FTP_DIT_EXT.1.1 TSS

Objective:

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

Evaluator Findings:

The evaluator reviewed the TSS section 6 and ensured that it contains the calls to the platform that TOE is leveraging to invoke the functionality.

The relevant information is found in the following section(s): TOE Summary Specification FTP_DIT_EXT.1.

Upon investigation, the evaluator found that the TSS states that:

All external communications are protected by SSH or TLS.

SSH protocol is provided by Venafi's maintained internal branch of Maverick, while the TLS protocol is provided by the underlying platform. The TOE uses .NET to invoke the platform's TLS functionality. The TOE uses System.Net APIs and System.ServiceModel APIs to leverage the underlying platform's TLS functionality.

Based on these findings, this assurance activity is considered satisfied.

Verdict:

PASS.

5.1.6.1.1 FTP_DIT_EXT.1.1 AGD

None.

5.2 Selection-Based Requirements

5.2.1 Cryptographic Support (FCS)

5.2.1.1 FCS_CKM.1/AK Cryptographic Asymmetric Key Generation

5.2.1.1.1 FCS_CKM.1.1/AK TSS

Objective:

- 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.

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

Evaluator Findings:

- The evaluator reviewed the TSS section 6 and ensured that it identifies the key sizes supported by the TOE.
- The evaluator reviewed the TSS section 6 and ensured that, if the ST specifies more than one scheme, the TSS identifies the usage for each scheme.
- The evaluator reviewed the TSS section 6 and ensured that, if the application "invokes platform-provided functionality for asymmetric key generation," the TSS describes how the key generation functionality is invoked.

The relevant information is found in the following section(s): TOE Summary Specification FCS_CKM.1/AK.

Upon investigation, the evaluator found that the TSS states that:

Through .Net the TOE is able to call the underlying Windows cryptographic modules.

The SSH and TLS components of TPP makes use of .NET cryptographic modules for the encryption and decryption of data. SSH itself is provided by Venafi's maintained internal branch of Maverick, while the TLS protocol is provided by Microsoft .NET alongside TLS cryptography.

The key generation schemes are RSA-based with key sizes of 2048-bits or 3072-bits, or elliptic curve-based with NIST curves, P-256, P-384, or P-521, or FFC schemes using "safe-prime" groups. These schemes are used for both TLS and SSH.

Based on these findings, this assurance activity is considered satisfied.

Verdict:

PASS.

5.2.1.1.2 FCS_CKM.1.1/AK AGD

The evaluator shall verify that the AGD 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.

Evaluator Findings:

The evaluator checked the AGD and ensured that it 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.

Upon investigation, the evaluator found that the section 2 titled "Platform Configuration" describes how to configure the TOE platform to use FIPS compliant algorithms and the TLS ciphers supported by the platform. The TOE relies on underlying platform for TLS. The platform provides support for the following TLS 1.2 cipher suites:

- TLS_RSA_WITH_AES_128_CBC_SHA as defined in RFC 5246,
- TLS_RSA_WITH_AES_256_CBC_SHA as defined in RFC 5246,
- TLS_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246,
- TLS_RSA_WITH_AES_256_CBC_SHA256 as defined in RFC 5246,
- TLS_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5288,
- TLS_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5288,
- TLS_DHE_RSA_WITH_AES_128_CBC_SHA as defined in RFC 5246,
- TLS_DHE_RSA_WITH_AES_256_CBC_SHA as defined in RFC 5246,
- TLS_DHE_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5288

- TLS_DHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5288
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA as defined in RFC 8422
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA as defined in RFC 8422
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289

The evaluator also found that the section 4.1 titled “SSH Connectivity” in the AGD states that:

The following algorithms are not User configurable.

The TOE supports the use of following public key algorithms:

- ssh-rsa (RFC 4253)
- rsa-sha2-256 (RFC 8332)
- rsa-sha2-512 (RFC 8332)
- ecdsa-sha2-nistp256 (RFC 5656)
- ecdsa-sha2-nistp384 (RFC 5656)
- ecdsa-sha2-nistp521 (RFC 5656), and
- ssh-ed25519 (RFC 8709).

The TOE supports the use of following Key Exchange method:

- diffie-hellman-group14-sha256 (RFC 8268),
- diffie-hellman-group16-sha512 (RFC 8268),
- diffie-hellman-group18-sha512 (RFC 8268),
- ecdh-sha2-nistp256 (RFC 5656),
- ecdh-sha2-nistp384 (RFC 5656),
- ecdh-sha2-nistp521 (RFC 5656), and
- curve25519-sha256 (RFC 8731).

Based on these findings, this assurance activity is considered satisfied.

Verdict:

PASS.

5.2.1.2 FCS_CKM.2 Cryptographic Key Establishment

5.2.1.2.1 FCS_CKM.2.1 TSS [TD0717]

Objective:

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

Evaluator Findings:

- The evaluator reviewed the ST selections and ensured that the supported key establishment schemes correspond to the key generation schemes identified in FCS_CKM.1.1/AK.
- The evaluator reviewed the TSS Section 6 and ensured that, if the ST specifies more than one scheme, it identifies the usage for each scheme.

The relevant information is found in the following section(s): 6 TOE Summary Specification FCS_CKM.2.

Upon investigation, the evaluator found that the TSS states that:

The key establishment schemes are RSA-based RSAES-PKCS1-v1_5, or elliptic curve-based with NIST curves, P-256, P-384, or P-521, or FFC schemes using “safe-prime” groups. All three of these schemes are used for TLS. The scheme used is dependent on the selected cipher suites.

For SSH, the key establishment schemes are elliptic curve-based with NIST curve P-256, P-384, P-521, and FFC schemes using “safe-prime” groups.

All of these key establishment schemes correspond to the key generation schemes identified in FCS_CKM.1.1/AK.

Based on these findings, this assurance activity is considered satisfied.

Verdict:

PASS.

5.2.1.2.2 FCS_CKM.2.1 AGD

Objective:

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

Evaluator Findings:

The evaluator checked the AGD and ensured that it instructs the administrator how to configure the TOE to use the selected key establishment scheme(s).

Upon investigation, the evaluator found that the section 2 titled “Platform Configuration” describes how to configure the TOE platform to use FIPS compliant algorithms and the TLS ciphers supported by the platform. The TOE relies on underlying platform for TLS. The platform provides support for the following TLS 1.2 cipher suites:

- TLS_RSA_WITH_AES_128_CBC_SHA as defined in RFC 5246,
- TLS_RSA_WITH_AES_256_CBC_SHA as defined in RFC 5246,
- TLS_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246,
- TLS_RSA_WITH_AES_256_CBC_SHA256 as defined in RFC 5246,
- TLS_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5288,
- TLS_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5288,
- TLS_DHE_RSA_WITH_AES_128_CBC_SHA as defined in RFC 5246,
- TLS_DHE_RSA_WITH_AES_256_CBC_SHA as defined in RFC 5246,
- TLS_DHE_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5288
- TLS_DHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5288
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA as defined in RFC 8422
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA as defined in RFC 8422
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289

- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289

The evaluator also found that the section 4.1 titled “SSH Connectivity” in the AGD states that:

The following algorithms are not User configurable.

The TOE supports the use of following Key Exchange method:

- diffie-hellman-group14-sha256 (RFC 8268)
- diffie-hellman-group16-sha512 (RFC 8268)
- diffie-hellman-group18-sha512 (RFC 8268)
- ecdh-sha2-nistp256 (RFC 5656)
- ecdh-sha2-nistp384 (RFC 5656)
- ecdh-sha2-nistp521 (RFC 5656)
- curve25519-sha256 (RFC 8731)

Based on these findings, this assurance activity is considered satisfied.

Verdict:

PASS.

5.2.1.3 FCS_COP.1/SKC Cryptographic Operation - Encryption/Decryption

5.2.1.3.1 FCS_COP.1/SKC TSS

None.

5.2.1.3.2 FCS_COP.1/SKC AGD

Objective:

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.

Evaluator Findings:

The evaluator checked the AGD and ensured that it provides documentation to determine that any configuration that is required to be done to configure the functionality for the required modes and key sizes is present.

Upon investigation, the evaluator found that the section 4.1 titled “SSH Connectivity” in the AGD states that:

The following algorithms are not User configurable.

The TOE supports the use of following encryption algorithms:

- aes128-ctr
- aes256-ctr
- aes128-cbc
- aes256-cbc

Based on these findings, this assurance activity is considered satisfied.

Verdict:

PASS.

5.2.1.4 FCS_COP.1/Hash Cryptographic Operation - Hashing

5.2.1.4.1 FCS_COP.1.1/Hash TSS

Objective:

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.

Evaluator Findings:

The evaluator reviewed the TSS section 6 and ensured that the association of the hash function with other application cryptographic functions is documented in the TSS.

The relevant information is found in the following section(s): TOE Summary Specification FCS_COP.1/Hash

Upon investigation, the evaluator found that the TSS states that:

The TOE uses the underlying platform .NET cryptographic modules for all hashing, keyed-hashing, encryption and digital signature generation and verification functions.

The TOE uses SHA-1, SHA-256, SHA-384, and SHA-512 for cryptographic hashing as part of SSH trusted channel. Hash functions are used as part of public key authentication, protecting data in transit (via HMAC), key exchange, and digital signatures as part of SSH trusted channel.

Based on these findings, this assurance activity is considered satisfied.

Verdict:

PASS.

5.2.1.4.2 FCS_COP.1/Hash AGD

None.

5.2.1.5 FCS_COP.1/KeyedHash Cryptographic Operation - Keyed-Hash Message Authentication

5.2.1.5.1 FCS_COP.1/KeyedHash TSS

None.

5.2.1.5.2 FCS_COP.1/KeyedHash AGD

None.

5.2.1.6 FCS_COP.1/Sig Cryptographic Operation - Signing

5.2.1.6.1 FCS_COP.1/Sig TSS

None.

5.2.1.6.2 FCS_COP.1/Sig AGD

None.

5.2.2 Identification and Authentication (FIA)

5.2.2.1 FIA_X509_EXT.1 X.509 Certificate Validation

5.2.2.1.1 FIA_X509_EXT.1.1 TSS

Objective:

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

Evaluator Findings:

- The evaluator reviewed the TSS section 6 and ensured that it describes where the check of validity of the certificates takes place.
- The evaluator reviewed the TSS section 6 and ensured that it provides a description of the certificate path validation algorithm.

The relevant information is found in the following section(s): TOE Summary Specification FIA_X509_EXT.1.

Upon investigation, the evaluator found that the TSS states that:

The TOE uses X.509v3 certificates as defined by RFC 5280 to support authentication for TLS connections. All certificate validation is performed invoking the underlying Windows platform, and certificates are stored in the Windows certificate store. The TOE supports a chain length four or greater.

Certificate validation paths must terminate with a trusted CA certificate that contains the basicConstraints extension and a CA flag that is set to TRUE. ExtendedkeyUsage field validation is also performed.

CRLs are configurable and may be used for certificate revocation. Checking is also done for the basicConstraints extension and the CA flag to determine whether they are present and set to TRUE. If they are not, the certificate is not accepted.

When the application cannot establish a connection to a CRL distribution point to determine certificate validity the application will reject the connection.

Based on these findings, this assurance activity is considered satisfied.

Verdict:

PASS.

5.2.2.1.2 FIA_X509_EXT.1.1 AGD

None.

5.2.2.1.3 FIA_X509_EXT.1.2 TSS

None.

5.2.2.1.4 FIA_X509_EXT.1.2 AGD

None.

5.2.2.2 FIA_X509_EXT.2 X.509 Certificate Authentication

5.2.2.2.1 FIA_X509_EXT.2.1 TSS

Objective:

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

Evaluator Findings:

- The evaluator reviewed the TSS section 6 and ensured that it describes how the TOE chooses which certificates to use, and any necessary instructions in the administrative guidance for configuring the operating environment so that the TOE can use the certificates.
- The evaluator reviewed the TSS section 6 and ensured 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 reviewed the TSS section 6 and ensured that any distinctions between trusted channels are described.
- The evaluator reviewed the TSS section 6 and ensured that it, if the requirement that the administrator is able to specify the default action, the AGD contains instructions on how this configuration action is performed.

The relevant information is found in the following section(s): 6 TOE Summary Specification FIA_X509_EXT.2.

Upon investigation, the evaluator found that the TSS states that:

The TOE uses X.509v3 certificates as defined by RFC 5280 to support authentication for TLS connections. All certificate validation is performed by the underlying Windows platform, and certificates are stored in the Windows certificate store.

CRLs are configurable and may be used for certificate revocation. Checking is also done for the basicConstraints extension and the CA flag to determine whether they are present and set to TRUE. If they are not, the certificate is not accepted.

When the platform cannot establish a connection to a CRL distribution point to determine certificate validity the platform will reject the connection.

The evaluator examined the section titled 3.2 “Manage Administrative Permissions for System Objects”, 4.3 “TLS Connectivity” and 4.4 “Configuring CA Server” to verify that it describes configuring the operating environment so that the TOE can use the certificates.

Upon investigation, the evaluator found that the AGD states that:

In Trust Protection Platform, all administrative permissions are managed at the object level. Every encryption system object—folders, Credentials, Workflows, CAs, Devices, Applications, Certificates, Notifications, Channels, Logging Applications, Discoveries, and Discovery Surveys—has a permissions tab. From the object permissions tab, you select the users or groups you want to have permissions to the current object and its subordinate objects, then you select which permissions you want the user or group to have.

To assign permissions to an object in the Web Administration Console

- Log in to the Trust Protection Platform Web Administration Console.
- Select the object you want to grant permissions to.
- Click the General > Permissions tab.
- Click Add.
- Select a User or Group Identity, then click Select.
- Select the permissions you want the User or Group Identity to have, then click Apply/Save.

The product uses TLS to connect to a number of different peers. Its TLS functionality can be testing using the HTTP Connection Test functionality. Under the Support tab go to the HTTP Connection Test tab. The URL of the connection endpoint is used as the reference identifier.

Administrator should configure the platform settings verification mode to strict for verification of certificates.

To enable Verification of certificates ,

- go to Aperture , click on the square and locate TLS protect.
- Under policy tree , select platforms in drop-down.
- Click on the configured Platform and select “Strict” under Verification Mode drop-down.

To enable CRL check, go to Aperture,

- click on the square and locate TLS protect.
- Under policy tree, select platforms in drop-down.
- Click on the configured Platform and select “Strict” under Check CRL drop-down.

In the Web Administration Console, the user entitlements are listed on a user object’s General tab.

The CA server can be configured as below:

- Policy -> Add->Credential->Password Credential.
- Enter password (e.g 123456789101234567891012345678910).
- Policy->Add->CA Template->DigiCert
- CA Server will now be configured.

Based on these findings, this assurance activity is considered satisfied.

Verdict:
PASS.

5.2.2.2.2 FIA_X509_EXT.2 AGD

None.

5.2.3 Protection of the TSF (FPT)

5.2.3.1 FPT_TUD_EXT.2 Integrity for Installation and Update

5.2.3.1.1 FPT_TUD_EXT.2.1 TSS

None.

5.2.3.1.2 FPT_TUD_EXT.2.1 AGD

None.

5.2.3.1.3 FPT_TUD_EXT.2.2 TSS

None.

5.2.3.1.4 FPT_TUD_EXT.2.2 AGD

None.

5.2.3.1.5 FPT_TUD_EXT.2.3 TSS

Objective:

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.

Evaluator Findings:

The evaluator reviewed the TSS section 6 and ensured that it identifies how the application installation package is signed by an authorized source. The definition of an authorized source must be contained in the TSS.

The relevant information is found in the following section(s): TOE Summary Specification FPT_TUD_EXT.2.

Upon investigation, the evaluator found that the TSS states that:

All binaries are signed by Venafi using signtool.exe, which is a .Net framework tool for digital file signatures. Venafi is the only authorized source to sign the executable binary. Authorized source can be verified by right-clicking the .MSI file and select Properties. Under Digital Signatures Tab, Name of signer will indicate "Venafi, Inc".
Additionally, ensure that the binaries are downloaded from the authorized source – via https://download.venafi.com/. Users must have a username and password to login to download the binaries.

Based on these findings, this assurance activity is considered satisfied.

Verdict:

PASS.

5.2.3.1.6 FPT_TUD_EXT.2.3 AGD

None.

6 Security Assurance Requirements

6.1 Security Target (ASE)

According to the PP, there are no AA requirements for this SFR TSS and AGD requirements for this SFR.

6.2 Development (ADV)

6.2.1 ADV_FSP.1 Basic Functional Specification

6.2.1.1 Evaluation Activity

Objective:

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

Evaluator Findings:

The evaluator confirmed that the functional specification documentation is provided to support the evaluation activities described in Section 5.1 Security Functional Requirements, and other activities described for AGD, ATE, and AVA SARs.

Based on these findings, this assurance activity is considered satisfied.

Verdict:

PASS.

6.3 Guidance Documentation (AGD)

6.3.1 AGD_OPE.1 Operational User Guidance

6.3.1.1 Evaluation Activity

Objective:

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

- Some of the contents of the AGD will be verified by the evaluation activities in Section 5.1 Security Functional Requirements 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 AGD 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 digital signature. The TOE will likely contain security functionality that does not fall in the scope of evaluation under this PP. The AGD shall make it clear to an administrator which security functionality is covered by the evaluation activities.

Evaluator Findings:

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

- Some of the contents of the AGD were verified by the evaluation activities in Section 5.1 Security Functional Requirements and evaluation of the TOE according to the [CEM].
- Cryptographic functions are not provided by the TOE, section 2 titled "Platform Configuration" in the AGD states that "product relies on the underlying cryptographic functionality of the Windows Server platform, Windows Server 2016 must be in FIPS mode to restrict its ciphers to Common Criteria requirements."
- The section 7 titled "TOE Updates" in the AGD describes the process for verifying updates to the TOE by verifying a digital signature, the AGD states that "All binaries are signed using signtool.exe, which is a .Net framework tool for digital file signatures."
  - The evaluator verified that this process includes the following steps:
- Section 8 "Secure Acceptance of TOE" of the AGD contains Instructions for obtaining the update itself. This includes instructions for making the update accessible to the TOE.
- Section 7 titled "TOE Updates" in the AGD contains instructions for initiating the update process, as well as discerning whether the process was successful or unsuccessful. This includes generation of the digital signature. Section 9 titled "Security Functions provided by the TOE" in the AGD makes it clear to an administrator which security functionality is covered by the evaluation activities and also identifies which security functionality does not fall in the scope of evaluation under this PP.

Based on these findings, this assurance activity is considered satisfied.

Verdict:

PASS.

6.3.2 Preparative Procedures (AGD_PRE.1)

6.3.2.1 Evaluation Activity

Objective:

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.

Evaluator Findings:

The evaluator reviewed the AGD, section 1.1 titled "Evaluation Platforms", and verified that the configuration requirements of the TOE platform are installed. The evaluator also reviewed the AGD, section 1.1 titled "Evaluation Platforms", section 2 titled "Platform Configuration", and section 3 titled "Product Configuration", and verified that the guidance provided for the TOE adequately addresses all platforms claimed for the TOE in the ST.

Based on these findings, this assurance activity is considered satisfied.

Verdict:

PASS.

6.4 Life-cycle Support (ALC)

6.4.1 ALC_CMC.1 Labeling of the TOE

6.4.1.1 Evaluation Activity

Objective:

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

Evaluator Findings:

- The evaluator reviewed the ST section 1.1 titled "Security Target and TOE Reference", to verify that the ST contains an identifier (such as a product name/version number) that specifically identifies the version that meets the requirements of the ST.
- The evaluator reviewed the AGD section 1 titled "Overview", and ensured that the version number is consistent with that in the ST.
- The evaluator reviewed the vendor's website https://docs.venafi.com/Docs/23.1/TopNav/Content/Release-Documents/Venafi%20Platform%20PDF%20Documentation.php?cshid=pdf and ensured that the information in the ST is sufficient to distinguish the product.

Based on these findings, this assurance activity is considered satisfied.

Verdict:

PASS.

6.5 TOE CM Coverage (ALC_CMS)

6.5.1 ALC_CMS.1

6.5.1.1 Evaluation Activity

Objective:

- 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 evaluation activity for ALC_CMC.1), the evaluator implicitly confirms the information required by this component. Life-cycle support is targeted aspects of the developer's life-cycle and instructions to providers of applications for the developer's devices, rather than an in-depth examination of the TSF manufacturer's development and configuration management process. This is not meant to diminish the critical role that a developer's practices play in contributing to the overall trustworthiness of a product; rather, it's a reflection on the information to be made available for evaluation.

- The evaluator shall ensure that the developer has identified (in AGD 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.

Evaluator Findings:

- The evaluator confirmed that the TOE is specifically identified and that this identification is consistent in the ST and in the AGD guidance (as done in the evaluation activity for ALC_CMC.1).
- The evaluator found that TOE is not a platform for developers, rather it is an application software running on Windows Server, also from ST TSS section "FPT_AEX_EXT.1" it is stated that

  "It is not necessary to use compiler flags to enable ASLR. The TOE's code is not run natively, but instead as managed code on top of Microsoft's .Net.

  Similarly, the use of a managed code base means that compiler flags aren't used for stack-based buffer overflow protection. Stack Based buffer overflows are protected in managed code by an exception being thrown by the CLR rather than having the overflow happen on the stack."

  Hence this is not applicable.
- The evaluator ensured that this documentation also includes an indication of whether such protections are on by default or have to be specifically enabled.

  Upon investigation, the evaluator found that AGD section 9.5 "Protection of the TSF", states that "During compilation, the TOE is built with several flags enabled that check for engineering flaws. The TOE is built with the /GS flag enabled. This reduces the possibilities of stack-based buffer overflows in the product."
- The evaluator ensured 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.

Based on these findings, this assurance activity is considered satisfied.

Verdict:

PASS.

6.6 Timely Security Updates (ALC_TSU_EXT)

6.6.1 ALC_TSU_EXT.1

6.6.1.1 Evaluation Activity

Objective:

- 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.

Evaluator Findings:

- The evaluator verified that the TSS section 6 titled "ALC_TSU_EXT.1" contains a description of the timely security update process used by the developer to create and deploy security updates.
- The evaluator verified that this description addresses the entire application.
- The evaluator also verified that, in addition to the TOE developer's process, any third-party processes are also addressed in the description.

Upon instigation the evaluator found that "There are no third-party processes".

- The evaluator also verified that each mechanism for deployment of security updates is described.
- The evaluator verified 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.

Upon investigation evaluator found that TSS section 6 titled "ALC_TSU_EXT.1" states that, "Venafi does not have a policy currently in place specifically for response times regarding public disclosure of vulnerabilities, however Venafi does have an existing policy for vulnerability SLAs for the product:

Critical: 14 days

High: 60 days

Medium: By the time of next major patch release. (e.g. 24.1 -> 24.3)

Low: By the time of next major patch release."

- The evaluator verified that this time is expressed in a number or range of days.
- The evaluator verified that this description includes the publicly available mechanisms (including either an email address or website) for reporting security issues related to the TOE.

Upon investigation evaluator found that ST section 5.6 Assurance measures "ALC_TSU_EXT.1" states that,

"Venafi uses a systematic method for identifying and providing security relevant updates to the TOEs users via its support infrastructure. Users can report issues using the Venafi Customer Portal https://customerportal.venafi.com/ or by emailing support@venafi.com"

- The evaluator verified 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.

Based on these findings, this assurance activity is considered satisfied.

Verdict:

PASS.

6.7 Tests (ATE)

6.7.1 ATE_IND.1 Independent Testing - Conformance

6.7.1.1 Evaluation Activity

Objective:

- 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 evaluation activities.
- While it is not necessary to have one test case per test listed in an evaluation activity, the evaluator must document in the test plan that each applicable testing requirement in the ST is covered. The test plan identifies the platforms to be tested, and for those platforms not included in the test plan but included in the ST, the test plan provides a justification for not testing the platforms. This justification must address the differences between the tested platforms and the untested platforms, and make an argument that the differences do not affect the testing to be performed. It is not sufficient to merely assert that the differences have no effect; 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 (e.g 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.

Evaluator Findings:

- The evaluator prepared a test plan and report documenting the testing aspects of the system, including any application crashes during testing.
- The evaluator determined the root cause of any application crashes and include that information in the report. The test plan covers all of the testing actions contained in the [CEM] and the body of this PP's evaluation activities.
- The test plan covers all of the testing actions contained in the [CEM] and the body of this PP's evaluation activities.
- The test plan covers all of the testing actions contained in the [CEM] and the body of this PP's evaluation activities.
- The test plan covers all of the testing actions contained in the [CEM] and the body of this PP's evaluation activities.

Based on these findings, this assurance activity is considered satisfied.

Verdict:

PASS.

6.8 Vulnerability Assessment (AVA)

6.8.1 AVA_VAN.1 Vulnerability Survey

6.8.1.1 Evaluation Activity

Objective:

- 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 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.

Evaluator Findings:

- The evaluator performed a Vulnerability Assessment for Venafi Trust Protection Platform v23.1, on July 24, 2024, and generated a report to document their findings with respect to this requirement. The evaluator performed 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 documented the sources consulted and the vulnerabilities found in the report.

- The evaluator examined public domain vulnerability searches by performing a keyword search. The terms used for this search were based on the vendor name, product name, and key platform features leveraged by the product. As a result, the evaluator performed a search using the following keywords:
    - Venafi
    - Trust Protection Platform v23.1
    - IronPython
    - Chaos.NaCl
    - Microsoft Intune CSR Validation
    - Sustainsys Saml2
    - Excelsior JET
    - F5 iControl Assembly for .NET
    - Bootstrap
    - Backbone
    - Underscore
    - Jquery
    - date.js
    - dateRangePicker.js, dateRangePicker.css
    - moment.js
    - easyDate.js
    - maskedInput.js
    - browser.js
    - jquery.timepicker.js
    - Select2.js
    - moment-timezone.js
    - core.js
    - dropzone.js
    - JSON.Net
    - ASP.NET Web Stack
    - Sencha Ext JS
    - Tigra Calendar
    - Pretty-Print JSON
    - D3.js
    - chart.js
    - mustache.js

- The evaluator examined the AVA document and concluded that for each vulnerability found, the report included a determination if the vulnerability applied to the product and if it did, the action that occurred to remediate the vulnerability. If a vulnerability did not apply to the product, a rational of why the vulnerability did not apply to the TOE was included. The evaluator concluded that this work unit is satisfied.

- For Windows: The evaluator also ran a virus scanner using Windows Defender Antivirus scanner with the most current virus definitions against the application files and verified that no files are flagged as malicious. The scan was performed on July 24, 2024.

Based on these findings, this assurance activity is considered satisfied.

Verdict:

PASS.

7 Test Cases

7.1 APP_V1.4

7.1.1 FCS_CKM_EXT.1.1

Item	Data
Test Assurance Activity	No test assurance activities have been identified. **TD0717 has been applied.**
Pass/Fail with Explanation	N/A. No test assurance activities have been identified. Based on these findings, this assurance activity is considered satisfied.

7.1.2 FCS_CKM.1.1/AK - RSA

Item	Data
Test Assurance Activity	If the application **"implements asymmetric key generation,"** then the following test activities shall be carried out. Evaluation Activity Note: The following tests may require the developer to provide access to a developer environment that provides the evaluator with tools that are typically available to end-users of the application. **Key Generation for FIPS PUB 186-4 RSA Schemes** The evaluator shall verify the implementation of RSA Key Generation by the TOE using the Key Generation test. This test verifies the ability of the TSF to correctly produce values for the key components including the public verification exponent e, the private prime factors p and q, the public modulus n and the calculation of the private signature exponent d. Key Pair generation specifies 5 ways (or methods) to generate the primes p and q. These include: 1. Random Primes: ○ Provable primes ○ Probable primes 2. Primes with Conditions: ○ Primes p1, p2, q1,q2, p and q shall all be provable primes ○ Primes p1, p2, q1, and q2 shall be provable primes and p and q shall be probable primes ○ Primes p1, p2, q1,q2, p and q shall all be probable primes To test the key generation method for the Random Provable primes method and for all the Primes with Conditions methods, the evaluator must seed the TSF key generation routine with sufficient data to deterministically generate the RSA key pair. This includes the random seed(s), the public exponent of the RSA key, and the desired key length. For each key length supported, the evaluator shall have the TSF generate 25 key pairs. The evaluator shall verify the correctness of the TSF's implementation by comparing values generated by the TSF with those generated from a known good implementation. If possible, the Random Probable primes method should also be verified against a known good implementation as described above. Otherwise, the evaluator shall have the TSF generate 10 keys pairs for each supported key length nlen and verify: • n = p·q,

	- p and q are probably prime according to Miller-Rabin tests, - GCD(p-1,e) = 1, - GCD(q-1,e) = 1, - $2^{16} \le e \le 2^{256}$ and e is an odd integer, - $\|p-q\| > 2^{nlen/2 - 100}$, - $p \ge 2^{nlen/2 -1/2}$, - $q \ge 2^{nlen/2 -1/2}$, - $2^{(nlen/2)} < d < LCM(p-1,q-1)$, - $e \cdot d = 1 \bmod LCM(p-1,q-1)$.
Pass/Fail with Explanation	Algorithm: RSA KeyGen (FIPS186-4) Key size / Modulus: 2048-bit or 3072-bit CAVP #: RSA 2195 Pass. Based on these findings, this assurance activity is considered satisfied.

7.1.3 FCS_CKM.1.1/AK - ECC

Item	**Data**
Test Assurance Activity	If the application **"implements asymmetric key generation,"** then the following test activities shall be carried out. Evaluation Activity Note: The following tests may require the developer to provide access to a developer environment that provides the evaluator with tools that are typically available to end-users of the application. **Key Generation for Elliptic Curve Cryptography (ECC)** FIPS 186-4 ECC Key Generation Test For each supported NIST curve, i.e., P-256, P-384 and P-521, the evaluator shall require the implementation under test (IUT) to generate 10 private/public key pairs. The private key shall be generated using an approved random bit generator (RBG). To determine correctness, the evaluator shall submit the generated key pairs to the public key verification (PKV) function of a known good implementation. FIPS 186-4 Public Key Verification (PKV) Test For each supported NIST curve, i.e., P-256, P-384 and P-521, the evaluator shall generate 10 private/public key pairs using the key generation function of a known good implementation and modify five of the public key values so that they are incorrect, leaving five values unchanged (i.e., correct). The evaluator shall obtain in response a set of 10 PASS/FAIL values.
Pass/Fail with Explanation	Algorithm: ECDSA KeyGen (FIPS186-4), ECDSA KeyVer (FIPS186-4) Curves: “NIST curves” P-384 and [P-256, P-521] CAVP #: ECDSA 911 Pass. Based on these findings, this assurance activity is considered satisfied.

7.1.4 FCS_CKM.1.1/AK - FFC

Item	**Data**
Test Assurance Activity	If the application **"implements asymmetric key generation,"** then the following test activities shall be carried out. Evaluation Activity Note: The following tests may require the developer to provide access to a developer environment that provides the evaluator with tools that are typically available to end-users of the application.

	Key Generation for Finite-Field Cryptography (FFC) The evaluator shall verify the implementation of the Parameters Generation and the Key Generation for FFC by the TOE using the Parameter Generation and Key Generation test. This test verifies the ability of the TSF to correctly produce values for the field prime p, the cryptographic prime q (dividing p-1), the cryptographic group generator g, and the calculation of the private key x and public key y. The Parameter generation specifies 2 ways (or methods) to generate the cryptographic prime q and the field prime p: Cryptographic and Field Primes: • Primes q and p shall both be provable primes • Primes q and field prime p shall both be probable primes and two ways to generate the cryptographic group generator g: Cryptographic Group Generator: • Generator g constructed through a verifiable process • Generator g constructed through an unverifiable process. The Key generation specifies 2 ways to generate the private key x: Private Key: • len(q) bit output of RBG where 1 ≤x ≤ q-1 • len(q) + 64 bit output of RBG, followed by a mod q-1 operation where 1≤ x≤q-1. The security strength of the RBG must be at least that of the security offered by the FFC parameter set. To test the cryptographic and field prime generation method for the provable primes method and/or the group generator g for a verifiable process, the evaluator must seed the TSF parameter generation routine with sufficient data to deterministically generate the parameter set. For each key length supported, the evaluator shall have the TSF generate 25 parameter sets and key pairs. The evaluator shall verify the correctness of the TSF's implementation by comparing values generated by the TSF with those generated from a known good implementation. Verification must also confirm • $g \neq 0,1$ • q divides p-1 • $g^q \bmod p = 1$ • $g^x \bmod p = y$ for each FFC parameter set and key pair.
Pass/Fail with Explanation	NA. Not claimed in ST.

7.1.5 FCS_CKM.1.1/AK - DH14 and FFC

Item	Data
Test Assurance Activity	If the application **"implements asymmetric key generation,"** then the following test activities shall be carried out. Evaluation Activity Note: The following tests may require the developer to provide access to a developer environment that provides the evaluator with tools that are typically available to end-users of the application. **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.
Pass/Fail with Explanation	Testing for FFC Schemes using safe-prime groups is done as part of testing in CKM.2.1. Pass. Based on these findings, this assurance activity is considered satisfied.

7.1.6 FCS_CKM.2.1 – SP800-56A

Item	**Data**
Test Assurance Activity	Evaluation Activity Note: The following tests require the developer to provide access to a test platform that provides the evaluator with tools that are typically not found on factory products. **Key Establishment Schemes** The evaluator shall verify the implementation of the key establishment schemes supported by the TOE using the applicable tests below. **SP800-56A Key Establishment Schemes** The evaluator shall verify a TOE's implementation of SP800-56A key agreement schemes using the following Function and Validity tests. These validation tests for each key agreement scheme verify that a TOE has implemented the components of the key agreement scheme according to the specifications in the Recommendation. These components include the calculation of the DLC primitives (the shared secret value Z) and the calculation of the derived keying material (DKM) via the Key Derivation Function (KDF). If key confirmation is supported, the evaluator shall also verify that the components of key confirmation have been implemented correctly, using the test procedures described below. This includes the parsing of the DKM, the generation of MACdata and the calculation of MACtag. **Function Test** The Function test verifies the ability of the TOE to implement the key agreement schemes correctly. To conduct this test the evaluator shall generate or obtain test vectors from a known good implementation of the TOE supported schemes. For each supported key agreement scheme-key agreement role combination, KDF type, and, if supported, key confirmation role- key confirmation type combination, the tester shall generate 10 sets of test vectors. The data set consists of one set of domain parameter values (FFC) or the NIST approved curve (ECC) per 10 sets of public keys. These keys are static, ephemeral or both depending on the scheme being tested. The evaluator shall obtain the DKM, the corresponding TOE’s public keys (static and/or ephemeral), the MAC tag(s), and any inputs used in the KDF, such as the Other Information (OtherInfo) and TOE id fields. If the TOE does not use a KDF defined in SP 800-56A, the evaluator shall obtain only the public keys and the hashed value of the shared secret. The evaluator shall verify the correctness of the TSF’s implementation of a given scheme by using a known good implementation to calculate the shared secret value, derive the keying material DKM, and compare hashes or MAC tags generated from these values.

	If key confirmation is supported, the TSF shall perform the above for each implemented approved MAC algorithm. **Validity Test** The Validity test verifies the ability of the TOE to recognize another party's valid and invalid key agreement results with or without key confirmation. To conduct this test, the evaluator shall obtain a list of the supporting cryptographic functions included in the SP800-56A key agreement implementation to determine which errors the TOE should be able to recognize. The evaluator generates a set of 24 (FFC) or 30 (ECC) test vectors consisting of data sets including domain parameter values or NIST approved curves, the evaluator's public keys, the TOE's public/private key pairs, MACTag, and any inputs used in the KDF, such as the OtherInfo and TOE id fields. The evaluator shall inject an error in some of the test vectors to test that the TOE recognizes invalid key agreement results caused by the following fields being incorrect: the shared secret value Z, the DKM, the OtherInfo field, the data to be MACed, or the generated MACTag. If the TOE contains the full or partial (only ECC) public key validation, the evaluator will also individually inject errors in both parties' static public keys, both parties' ephemeral public keys and the TOE's static private key to assure the TOE detects errors in the public key validation function and/or the partial key validation function (in ECC only). At least two of the test vectors shall remain unmodified and therefore should result in valid key agreement results (they should pass). The TOE shall use these modified test vectors to emulate the key agreement scheme using the corresponding parameters. The evaluator shall compare the TOE's results with the results using a known good implementation verifying that the TOE detects these errors.
Pass/Fail with Explanation	Algorithm: KAS-ECC Curves: P-256, P-384, P-521 CAVP #: KAS 92 Pass. Based on these findings, this assurance activity is considered satisfied.

7.1.7 FCS_CKM.2.1 – SP800-56B

Item	Data
Test Assurance Activity	Evaluation Activity Note: The following tests require the developer to provide access to a test platform that provides the evaluator with tools that are typically not found on factory products. **Key Establishment Schemes** The evaluator shall verify the implementation of the key establishment schemes supported by the TOE using the applicable tests below. **SP800-56B Key Establishment Schemes** The evaluator shall verify that the TSS describes whether the TOE acts as a sender, a recipient, or both for RSA-based key establishment schemes.

If the TOE acts as a sender, the following evaluation activity shall be performed to ensure the proper operation of every TOE supported combination of RSA-based key establishment scheme:

To conduct this test the evaluator shall generate or obtain test vectors from a known good implementation of the TOE supported schemes. For each combination of supported key establishment scheme and its options (with or without key confirmation if supported, for each supported key confirmation MAC function if key confirmation is supported, and for each supported mask generation function if KTS-OAEP is supported), the tester shall generate 10 sets of test vectors. Each test vector shall include the RSA public key, the plaintext keying material, any additional input parameters if applicable, the MacKey and MacTag if key confirmation is incorporated, and the outputted ciphertext. For each test vector, the evaluator shall perform a key establishment encryption operation on the TOE with the same inputs (in cases where key confirmation is incorporated, the test shall use the MacKey from the test vector instead of the randomly generated MacKey used in normal operation) and ensure that the outputted ciphertext is equivalent to the ciphertext in the test vector.

If the TOE acts as a receiver, the following evaluation activities shall be performed to ensure the proper operation of every TOE supported combination of RSA-based key establishment scheme:

To conduct this test the evaluator shall generate or obtain test vectors from a known good implementation of the TOE supported schemes. For each combination of supported key establishment scheme and its options (with our without key confirmation if supported, for each supported key confirmation MAC function if key confirmation is supported, and for each supported mask generation function if KTS-OAEP is supported), the tester shall generate 10 sets of test vectors. Each test vector shall include the RSA private key, the plaintext keying material (KeyData), any additional input parameters if applicable, the MacTag in cases where key confirmation is incorporated, and the outputted ciphertext. For each test vector, the evaluator shall perform the key establishment decryption operation on the TOE and ensure that the outputted plaintext keying material (KeyData) is equivalent to the plaintext keying material in the test vector. In cases where key confirmation is incorporated, the evaluator shall perform the key confirmation steps and ensure that the outputted MacTag is equivalent to the MacTag in the test vector.

The evaluator shall ensure that the TSS describes how the TOE handles decryption errors. In accordance with NIST Special Publication 800-56B, the TOE must not reveal the particular error that occurred, either through the contents of any outputted or logged error message or through timing variations. If KTS-OAEP is supported, the evaluator shall create separate contrived ciphertext values that trigger each of the three decryption error checks described in NIST Special Publication 800-56B section 7.2.2.3, ensure that each decryption attempt results in an error, and ensure that any outputted or logged error message is identical for each. If KTS-KEM-KWS is supported, the evaluator shall create separate contrived ciphertext values that trigger each of the three decryption error checks described in NIST Special Publication 800-56B section 7.2.3.3, ensure that each decryption attempt results in an error, and ensure that any outputted or logged error message is identical for each.

Pass/Fail with Explanation	NA. Not claimed in ST.

7.1.8 FCS_CKM.2.1 – RSA

Item	Data
Test Assurance Activity	Evaluation Activity Note: The following tests require the developer to provide access to a test platform that provides the evaluator with tools that are typically not found on factory products. **Key Establishment Schemes** The evaluator shall verify the implementation of the key establishment schemes supported by the TOE using the applicable tests below. **RSA-based key establishment** The evaluator shall verify the correctness of the TSF's implementation of RSAES-PKCS1-v1_5 by using a known good implementation for each protocol selected in FTP_DIT_EXT.1 that uses RSAES-PKCS1-v1_5.
Pass/Fail with Explanation	Pass. This testing was performed in conjunction with FTP_DIT_EXT.1 to demonstrate correct operation.

7.1.9 FCS_CKM.2.1 – DH14

Item	Data
Test Assurance Activity	Evaluation Activity Note: The following tests require the developer to provide access to a test platform that provides the evaluator with tools that are typically not found on factory products. **Key Establishment Schemes** The evaluator shall verify the implementation of the key establishment schemes supported by the TOE using the applicable tests below. **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.
Pass/Fail with Explanation	NA. not claimed in ST.

7.1.10 FCS_CKM.2.1 – FFC

Item	Data
Test Assurance Activity	Evaluation Activity Note: The following tests require the developer to provide access to a test platform that provides the evaluator with tools that are typically not found on factory products. **Key Establishment Schemes** The evaluator shall verify the implementation of the key establishment schemes supported by the TOE using the applicable tests below.

	FFC Schemes using "safe-prime" groups The evaluator shall verify the correctness of the TSF's implementation of safe-prime groups by using a known good implementation for each protocol selected in FTP_DIT_EXT.1 that uses safe-prime groups. This test must be performed for each safe-prime group that each protocol uses.
Pass/Fail with Explanation	This test has been successfully tested in FTP_DIT_EXT.1 that uses safe-prime groups. The evaluator tested each protocol and verified the successful connection. Pass. Based on these findings, this assurance activity is considered satisfied.

7.1.11 FCS_RBG_EXT.1 Test #1

Item	**Data**
Test Assurance Activity	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: **Platforms:Microsoft Windows...** The evaluator shall verify that rand_s, RtlGenRandom, BCryptGenRandom, or CryptGenRandom API is used for classic desktop applications. The evaluator shall verify the application uses the RNGCryptoServiceProvider class or derives a class from System.Security.Cryptography.RandomNumberGenerator API for Windows Universal Applications. It is only required that the API is called/invoked, there is no requirement that the API be used directly. In future versions of this document, CryptGenRandom may be removed as an option as it is no longer the preferred API per vendor documentation. If invocation of platform-provided functionality is achieved in another way, the evaluator shall ensure the TSS describes how this is carried out, and how it is equivalent to the methods listed here (e.g. higher-level API invokes identical low-level API).
Test Steps	• Check the API listed in TSS for invoking platform provided DRBG • The evaluator uses a decompiler to verify that the TOE is invoking acceptable platform APIs and the list matched with TSS.
Expected Test Results	Screenshot evidence of the decompiler showing that the invoked platform derives a class from: • System.Security.Cryptography.RandomNumberGenerator • BCryptGenRandom

Pass/Fail with Explanation	Pass. The System.Security.Cryptography.RandomNumberGenerator and BCryptGenRandom API appears in the Decompiler's output , this meets the testing requirement.

7.1.12 FCS_COP.1/SKC AES-CBC

Item	Data
Test Assurance Activity	The evaluator shall perform all of the following tests for each algorithm implemented by the TSF and used to satisfy the requirements of this PP: **AES-CBC Known Answer Tests** There are four Known Answer Tests (KATs), described below. In all KATs, the plaintext, ciphertext, and IV values shall be 128-bit blocks. The results from each test may either be obtained by the evaluator directly or by supplying the inputs to the implementer and receiving the results in response. To determine correctness, the evaluator shall compare the resulting values to those obtained by submitting the same inputs to a known good implementation. **KAT-1.** To test the encrypt functionality of AES-CBC, the evaluator shall supply a set of 10 plaintext values and obtain the ciphertext value that results from AES-CBC encryption of the given plaintext using a key value of all zeros and an IV of all zeros. Five plaintext values shall be encrypted with a 128-bit all-zeros key, and the other five shall be encrypted with a 256-bit all- zeros key. To test the decrypt functionality of AES-CBC, the evaluator shall perform the same test as for encrypt, using 10 ciphertext values as input and AES-CBC decryption. **KAT-2.** To test the encrypt functionality of AES-CBC, the evaluator shall supply a set of 10 key values and obtain the ciphertext value that results from AES-CBC encryption of an all-zeros plaintext using the given key value and an IV of all zeros. Five of the keys shall be 128-bit keys, and the other five shall be 256-bit keys. To test the decrypt functionality of AES-CBC, the evaluator shall perform the same test as for encrypt, using an all-zero ciphertext value as input and AES-CBC decryption. **KAT-3.** To test the encrypt functionality of AES-CBC, the evaluator shall supply the two sets of key values described below and obtain the ciphertext value that results from AES encryption of an all-zeros plaintext using the given key value and an IV of all zeros. The first set of keys shall have 128 128-bit keys, and the second set shall have 256 256-bit keys. Key i in each set shall have the leftmost i bits be ones and the rightmost N-i bits be zeros, for i in [1,N]. To test the decrypt functionality of AES-CBC, the evaluator shall supply the two sets of key and ciphertext value pairs described below and obtain the plaintext value that results from AES-CBC decryption of the given ciphertext using the given key and an IV of all zeros. The first set of key/ciphertext pairs shall have 128 128-bit key/ciphertext pairs, and the second set of key/ciphertext pairs shall have 256 256-bit key/ciphertext pairs. Key i in each set shall have the leftmost i bits be ones and the rightmost N-i bits be zeros, for i in [1,N]. The ciphertext value in each pair shall be the value that results in an all-zeros plaintext when decrypted with its corresponding key. **KAT-4.** To test the encrypt functionality of AES-CBC, the evaluator shall supply the set of 128 plaintext values described below and obtain the two ciphertext values that

result from AES-CBC encryption of the given plaintext using a 128-bit key value of all zeros with an IV of all zeros and using a 256-bit key value of all zeros with an IV of all zeros, respectively. Plaintext value i in each set shall have the leftmost i bits be ones and the rightmost 128-i bits be zeros, for i in [1,128].

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

AES-CBC Multi-Block Message Test

The evaluator shall test the encrypt functionality by encrypting an i-block message where 1 < i <= 10. The evaluator shall choose a key, an IV and plaintext message of length i blocks and encrypt the message, using the mode to be tested, with the chosen key and IV. The ciphertext shall be compared to the result of encrypting the same plaintext message with the same key and IV using a known good implementation. The evaluator shall also test the decrypt functionality for each mode by decrypting an i-block message where 1 < i <=10. The evaluator shall choose a key, an IV and a ciphertext message of length i blocks and decrypt the message, using the mode to be tested, with the chosen key and IV. The plaintext shall be compared to the result of decrypting the same ciphertext message with the same key and IV using a known good implementation. AES-CBC Monte Carlo Tests The evaluator shall test the encrypt functionality using a set of 200 plaintext, IV, and key 3- tuples. 100 of these shall use 128 bit keys, and 100 shall use 256 bit keys. The plaintext and IV values shall be 128-bit blocks. For each 3-tuple, 1000 iterations shall be run as follows:

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

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

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

Pass/Fail with Explanation	Algorithm: AES-CBC Key size: 128 bits, 256 bits CAVP #: AES 4064 Pass. Based on these findings, this assurance activity is considered satisfied.

7.1.13 FCS_COP.1/SKC AES-GCM

Item	**Data**

Test Assurance Activity	The evaluator shall perform all of the following tests for each algorithm implemented by the TSF and used to satisfy the requirements of this PP: **AES-GCM Monte Carlo Tests** The evaluator shall test the Cryptoenticated encrypt functionality of AES-GCM for each combination of the following input parameter lengths: • 128 bit and 256 bit keys • Two plaintext lengths. One of the plaintext lengths shall be a non-zero integer multiple of 128 bits, if supported. The other plaintext length shall not be an integer multiple of 128 bits, if supported. • Three AAD lengths. One AAD length shall be 0, if supported. One AAD length shall be a non-zero integer multiple of 128 bits, if supported. One AAD length shall not be an integer multiple of 128 bits, if supported. • Two IV lengths. If 96 bit IV is supported, 96 bits shall be one of the two IV lengths tested. The evaluator shall test the encrypt functionality using a set of 10 key, plaintext, AAD, and IV tuples for each combination of parameter lengths above and obtain the ciphertext value and tag that results from AES-GCM Cryptoenticated encrypt. Each supported tag length shall be tested at least once per set of 10. The IV value may be supplied by the evaluator or the implementation being tested, as long as it is known. The evaluator shall test the decrypt functionality using a set of 10 key, ciphertext, tag, AAD, and IV 5-tuples for each combination of parameter lengths above and obtain a Pass/Fail result on Cryptoentication and the decrypted plaintext if Pass. The set shall include five tuples that Pass and five that Fail. The results from each test may either be obtained by the evaluator directly or by supplying the inputs to the implementer and receiving the results in response. To determine correctness, the evaluator shall compare the resulting values to those obtained by submitting the same inputs to a known good implementation.
Pass/Fail with Explanation	NA. Not claimed in ST.

7.1.14 FCS_COP.1/SKC AES-XTS

Item	**Data**
Test Assurance Activity	The evaluator shall perform all of the following tests for each algorithm implemented by the TSF and used to satisfy the requirements of this PP: **AES-XTS Tests** The evaluator shall test the encrypt functionality of XTS-AES for each combination of the following input parameter lengths: 256 bit (for AES-128) and 512 bit (for AES-256) keys Three data unit (i.e., plaintext) lengths. One of the data unit lengths shall be a non-zero integer multiple of 128 bits, if supported. One of the data unit lengths shall be an

	integer multiple of 128 bits, if supported. The third data unit length shall be either the longest supported data unit length or 216 bits, whichever is smaller. Using a set of 100 (key, plaintext and 128-bit random tweak value) 3-tuples and obtain the ciphertext that results from XTS-AES encrypt. The evaluator may supply a data unit sequence number instead of the tweak value if the implementation supports it. The data unit sequence number is a base-10 number ranging between 0 and 255 that implementations convert to a tweak value internally. The evaluator shall test the decrypt functionality of XTS-AES using the same test as for encrypt, replacing plaintext values with ciphertext values and XTS-AES encrypt with XTS-AES decrypt.
Pass/Fail with Explanation	NA. Not claimed in ST.

7.1.15 FCS_COP.1/SKC AES-CCM

Item	Data
Test Assurance Activity	The evaluator shall perform all of the following tests for each algorithm implemented by the TSF and used to satisfy the requirements of this PP: **AES-CCM Tests** It is not recommended that evaluators use values obtained from static sources such as http://csrc.nist.gov/groups/STM/cavp/documents/mac/ccmtestvectors.zip or use values not generated expressly to exercise the AES-CCM implementation. The evaluator shall test the generation-encryption and decryption-verification functionality of AES-CCM for the following input parameter and tag lengths: • Keys: All supported and selected key sizes (e.g., 128, 256 bits). • Associated Data: Two or three values for associated data length: The minimum (≥ 0 bytes) and maximum (≤ 32 bytes) supported associated data lengths, and 2^16 (65536) bytes, if supported. • Payload: Two values for payload length: The minimum (≥ 0 bytes) and maximum (≤ 32 bytes) supported payload lengths. • Nonces: All supported nonce lengths (7, 8, 9, 10, 11, 12, 13) in bytes. • Tag: All supported tag lengths (4, 6, 8, 10, 12, 14, 16) in bytes. The testing for CCM consists of five tests. To determine correctness in each of the below tests, the evaluator shall compare the ciphertext with the result of encryption of the same inputs with a known good implementation. **Variable Associated Data Test** For each supported key size and associated data length, and any supported payload length, nonce length, and tag length, the evaluator shall supply one key value, one nonce value, and 10 pairs of associated data and payload values, and obtain the resulting ciphertext. **Variable Payload Test**

	For each supported key size and payload length, and any supported associated data length, nonce length, and tag length, the evaluator shall supply one key value, one nonce value, and 10 pairs of associated data and payload values, and obtain the resulting ciphertext. **Variable Nonce Test** For each supported key size and nonce length, and any supported associated data length, payload length, and tag length, the evaluator shall supply one key value, one nonce value, and 10 pairs of associated data and payload values, and obtain the resulting ciphertext. **Variable Tag Test** For each supported key size and tag length, and any supported associated data length, payload length, and nonce length, the evaluator shall supply one key value, one nonce value, and 10 pairs of associated data and payload values, and obtain the resulting ciphertext. **Decryption-Verification Process Test** To test the decryption-verification functionality of AES-CCM, for each combination of supported associated data length, payload length, nonce length, and tag length, the evaluator shall supply a key value and 15 sets of input plus ciphertext, and obtain the decrypted payload. Ten of the 15 input sets supplied should fail verification and five should pass.
Pass/Fail with Explanation	NA. Not claimed in ST.

7.1.16 FCS_COP.1/SKC AES-CTR

Item	**Data**
Test Assurance Activity	The evaluator shall perform all of the following tests for each algorithm implemented by the TSF and used to satisfy the requirements of this PP: **Test 1: Known Answer Tests (KATs)** There are four Known Answer Tests (KATs) described below. For all KATs, the plaintext, IV, and ciphertext values shall be 128-bit blocks. The results from each test may either be obtained by the validator directly or by supplying the inputs to the implementer and receiving the results in response. To determine correctness, the evaluator shall compare the resulting values to those obtained by submitting the same inputs to a known good implementation. To test the encrypt functionality, the evaluator shall supply a set of 10 plaintext values and obtain the ciphertext value that results from encryption of the given plaintext using a key value of all zeros and an IV of all zeros. Five plaintext values shall be encrypted with a 128-bit all zeros key, and the other five shall be encrypted with a 256-bit all zeros key. To test the decrypt functionality, the evaluator shall perform the same test as for encrypt, using 10 ciphertext values as input. To test the encrypt functionality, the evaluator shall supply a set of 10 key values and obtain the ciphertext value that results from encryption of an all zeros plaintext using the given key value and an IV of all zeros. Five of the key values shall be 128-bit keys,

and the other five shall be 256-bit keys. To test the decrypt functionality, the evaluator shall perform the same test as for encrypt, using an all zero ciphertext value as input.

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

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

Test 2: Multi-Block Message Test
The evaluator shall test the encrypt functionality by encrypting an i-block message where 1 less-than i less-than-or-equal to 10. For each i the evaluator shall choose a key, IV, and plaintext message of length i blocks and encrypt the message, using the mode to be tested, with the chosen key. The ciphertext shall be compared to the result of encrypting the same plaintext message with the same key and IV using a known good implementation. The evaluator shall also test the decrypt functionality by decrypting an i-block message where 1 less-than i less-than-or-equal to 10. For each i the evaluator shall choose a key and a ciphertext message of length i blocks and decrypt the message, using the mode to be tested, with the chosen key. The plaintext shall be compared to the result of decrypting the same ciphertext message with the same key using a known good implementation.

Test 3: Monte-Carlo Test
For AES-CTR mode perform the Monte Carlo Test for ECB Mode on the encryption engine of the counter mode implementation. There is no need to test the decryption engine.

The evaluator shall test the encrypt functionality using 200 plaintext/key pairs. 100 of these shall use 128 bit keys, and 100 of these shall use 256 bit keys. The plaintext values shall be 128-bit blocks. For each pair, 1000 iterations shall be run as follows:

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

	The ciphertext computed in the 1000th iteration is the result for that trial. This result shall be compared to the result of running 1000 iterations with the same values using a known good implementation.
Pass/Fail with Explanation	Algorithm: AES-CTR Key size: 128 bits, 256 bits CAVP #: AES 4064 Pass. Based on these findings, this assurance activity is considered satisfied.

7.1.17 FCS_COP.1/Hash

Item	Data
Test Assurance Activity	The TSF hashing functions can be implemented in one of two modes. The first mode is the byte-oriented mode. In this mode the TSF hashes only messages that are an integral number of bytes in length; i.e., the length (in bits) of the message to be hashed is divisible by 8. The second mode is the bit-oriented mode. In this mode the TSF hashes messages of arbitrary length. As there are different tests for each mode, an indication is given in the following sections for the bit-oriented vs. the byte-oriented testmacs. The evaluator shall perform all of the following tests for each hash algorithm implemented by the TSF and used to satisfy the requirements of this PP. The following tests require the developer to provide access to a test application that provides the evaluator with tools that are typically not found in the production application. **Test 1:** Short Messages Test - Bit oriented Mode. The evaluators devise an input set consisting of m+1 messages, where m is the block length of the hash algorithm. The length of the messages range sequentially from 0 to m bits. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF. **Test 2:** Short Messages Test - Byte oriented Mode. The evaluators devise an input set consisting of m/8+1 messages, where m is the block length of the hash algorithm. The length of the messages range sequentially from 0 to m/8 bytes, with each message being an integral number of bytes. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF. **Test 3:** Selected Long Messages Test - Bit oriented Mode. The evaluators devise an input set consisting of m messages, where m is the block length of the hash algorithm. The length of the ith message is 512 + 99*i, where 1 ≤ i ≤ m. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF. **Test 4:** Selected Long Messages Test - Byte oriented Mode. The evaluators devise an input set consisting of m/8 messages, where m is the block length of the hash

	algorithm. The length of the ith message is 512 + 8*99*i, where 1 ≤ i ≤ m/8. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF. **Test 5:** Pseudorandomly Generated Messages Test. This test is for byte-oriented implementations only. The evaluators randomly generate a seed that is n bits long, where n is the length of the message digest produced by the hash function to be tested. The evaluators then formulate a set of 100 messages and associated digests by following the algorithm provided in Figure 1 of [SHAVS]. The evaluators then ensure that the correct result is produced when the messages are provided to the TSF.
Pass/Fail with Explanation	Algorithm: SHA-1, SHA2-256, SHA2-384, SHA2-512 CAVP #: SHS 3347 Pass. Based on these findings, this assurance activity is considered satisfied.

7.1.18 FCS_COP.1/KeyedHash

Item	Data
Test Assurance Activity	For each of the supported parameter sets, the evaluator shall compose 15 sets of test data. Each set shall consist of a key and message data. The evaluator shall have the TSF generate HMAC tags for these sets of test data. The resulting MAC tags shall be compared to the result of generating HMAC tags with the same key and IV using a known-good implementation.
Pass/Fail with Explanation	Algorithm: HMAC (SHA2-256, SHA2-384, SHA2-512) CAVP #: HMAC 2651 Pass. Based on these findings, this assurance activity is considered satisfied.

7.1.19 FCS_COP.1/Sig RSA

Item	Data
Test Assurance Activity	The following tests require the developer to provide access to a test application that provides the evaluator with tools that are typically not found in the production application. RSA Signature Algorithm Tests **Test 1:** Signature Verification Test. The evaluator shall perform the Signature Verification test to verify the ability of the TOE to recognize another party's valid and invalid signatures. The evaluator shall inject errors into the test vectors produced during the Signature Verification Test by introducing errors in some of the public keys, e, messages, IR format, and/or signatures. The TOE attempts to verify the signatures and returns success or failure. **Test 2:** Signature Generation Test. The evaluator shall verify the implementation of RSA Signature Generation by the TOE using the Signature Generation Test. To conduct this test the evaluator must generate or obtain 10 messages from a trusted reference implementation for each modulus size/SHA combination supported by the TSF. The

	evaluator shall have the TOE use their private key and modulus value to sign these messages. The evaluator shall verify the correctness of the TSF's signature using a known good implementation and the associated public keys to verify the signatures.
Pass/Fail with Explanation	Algorithm: RSA SigGen, RSA SigVer Key size / Modulus: 2048-bit or 3072-bit CAVP #: RSA 2192, RSA 2193 Pass. Based on these findings, this assurance activity is considered satisfied.

7.1.20 FCS_COP.1/Sig ECDSA

Item	Data
Test Assurance Activity	The following tests require the developer to provide access to a test application that provides the evaluator with tools that are typically not found in the production application. ECDSA Algorithm Tests Test 1: ECDSA FIPS 186-4 Signature Generation Test. For each supported NIST curve (i.e., P-256, P-384 and P-521) and SHA function pair, the evaluator shall generate 10 1024-bit long messages and obtain for each message a public key and the resulting signature values R and S. To determine correctness, the evaluator shall use the signature verification function of a known good implementation. Test 2: ECDSA FIPS 186-4 Signature Verification Test. For each supported NIST curve (i.e., P-256, P-384 and P-521) and SHA function pair, the evaluator shall generate a set of 10 1024-bit message, public key and signature tuples and modify one of the values (message, public key or signature) in five of the 10 tuples. The evaluator shall obtain in response a set of 10 PASS/FAIL values.
Pass/Fail with Explanation	Algorithm: ECDSA SigGen, ECDSA SigVer Curves: P-256, P-384 and P-521 CAVP #: ECDSA 911 Pass. Based on these findings, this assurance activity is considered satisfied.

7.1.21 FCS_STO_EXT.1.1 Test #1

Item	Data
Test Assurance Activity	For all credentials for which **the application implements functionality**, the evaluator shall verify credentials are encrypted according to FCS_COP.1/SKC or conditioned according to FCS_CKM.1.1/AK and FCS_CKM.1/PBKDF.
Pass/Fail with Explanation	NA. the ST does not select '**the application implements functionality'**.

7.1.22 FCS_STO_EXT.1.1 Test #2

Item	Data
Test Assurance Activity	For all credentials for which the application **invokes platform-provided functionality**, the evaluator shall perform the following actions which vary per platform.

	Platforms:Microsoft Windows... The evaluator shall verify that all certificates are stored in the Windows Certificate Store. The evaluator shall verify that other credentials, like passwords, are stored in the Windows Credential Manager or stored using the Data Protection API (DPAPI). For Windows Universal Applications, the evaluator shall verify that the application is using the ProtectData class and storing credentials in IsolatedStorage.
Test Steps	• Open Windows Certificate store • Verify that all certificates are stored in the Windows Certificate Store • Perform Static code analysis on the TOE and verify that other credentials, like passwords, are stored in the Windows Credential Manager or stored using the Data Protection API(DPAPI)
Expected Test Results	• all certificates are stored in the Windows Certificate Store • other credentials, like passwords, are stored in the Windows Credential Manager or stored using the Data Protection API(DPAPI)
Pass/Fail with Explanation	Pass. All certificates are stored in the Windows Certificate Store. Other credentials, like passwords, are stored using the Data Protection API (DPAPI).

7.1.23 FDP_DAR_EXT.1.1 Test #1

Item	Data
Test Assurance Activity	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. If "implement functionality to encrypt sensitive data as defined in the PP-Module for File Encryption" or "protect sensitive data in accordance with FCS_STO_EXT.1" is selected, 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. **TD0756 has been applied**
Pass/Fail with Explanation	Pass, All the sensitive data listed is covered as part of FCS_STO_EXT.1. There is no sensitive data listed that are not covered as part of FCS_STO_EXT.1.

7.1.24 FDP_DAR_EXT.1.1 Test #2

Item	Data
Test Assurance Activity	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. 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. **Platforms:Microsoft Windows...** The Windows 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, such as BitLocker or Encrypting File System (EFS), clear to the end user.

Pass/Fail with Explanation	NA. “**leverage platform-provided functionality”** is Not selected in the ST.

7.1.25 FDP_DEC_EXT.1.1 Test #1

Item	Data
Test Assurance Activity	**Platforms:Microsoft Windows...** For Windows Universal Applications the evaluator shall check the AppxManifest.xml file for a list of required hardware capabilities. The evaluator shall verify that the user is made aware of the required hardware capabilities when the application is first installed. This includes permissions such as ID_CAP_ISV_CAMERA, ID_CAP_LOCATION, ID_CAP_NETWORKING, ID_CAP_MICROPHONE, ID_CAP_PROXIMITY and so on. A complete list of Windows App permissions can be found at: http://msdn.microsoft.com/en-US/library/windows/apps/jj206936.aspx For Windows Desktop Applications the evaluator shall identify in either the application software or its documentation the list of the required hardware resources. **TD0822 has been applied**
Test Steps	• List the required hardware capabilities as per the ST. • The evaluator shall check the AGD and verify that it lists the required hardware resources.
Expected Test Results	• Screenshot evidence of the ST where it lists the hardware capabilities of the TOE. • Screenshot evidence of the AGD where it lists the hardware capabilities of the TOE.
Pass/Fail with Explanation	Pass. The TOE documentation lists the required hardware resources along with necessary justification.

7.1.26 FDP_DEC_EXT.1.2 Test #1

Item	Data
Test Assurance Activity	**Platforms:Microsoft Windows...** For Windows Universal Applications the evaluator shall check the AppxManifest.xml file for a list of required capabilities. The evaluator shall identify the required information repositories when the application is first installed. This includes permissions such as ID_CAP_CONTACTS,ID_CAP_APPOINTMENTS,ID_CAP_MEDIALIB and so on. A complete list of Windows App permissions can be found at: http://msdn.microsoft.com/en-US/library/windows/apps/jj206936.aspx For Windows Desktop Applications the evaluator shall identify in either the application software or its documentation the list of sensitive information repositories it accesses. **TD0822 has been applied**
Test Steps	• List the required sensitive information repositories as per the ST. • The evaluator shall check the AGD and verify that it lists the required sensitive information repositories.
Expected Test Results	• Screenshot evidence of the ST where it lists the sensitive information repositories of the TOE.

	- Screenshot evidence of the AGD where it lists the sensitive information repositories of the TOE.
Pass/Fail with Explanation	Pass. The TOE documentation lists the sensitive information repositories it accesses along with necessary justification.

7.1.27 FDP_NET_EXT.1.1 Test #1

Item	**Data**
Test Assurance Activity	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.
Test Steps	- Stop the Venafi TPP services. - Verify communications through packet capture in the absence of Venafi TPP services. - Start the Venafi TPP services. - The evaluator shall run the application (TOE) while sniffing network traffic. - Verify the network communication on the packet capture. - Verify that the network communications witnessed are documented in the TSS.
Expected Test Results	- Screenshot evidence of the running application. - Evidence of the packet captures to verify the network communications. - Screenshot evidence of the TSS mentioned with all the witnessed network communications.
Pass/Fail with Explanation	Pass. It has been verified that the network communications witnessed are documented in the TSS.

7.1.28 FDP_NET_EXT.1.1 Test #2

Item	**Data**
Test Assurance Activity	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).
Pass/Fail with Explanation	NA. ST does not select the third selection and its assignment (*"respond to [assignment: list of remotely initiated communication]"*).

7.1.29 FDP_NET_EXT.1.1 Test #3

Item	**Data**
Test Assurance Activity	**Platforms:Android...** If **"no network communication"** is selected, the evaluator shall ensure that the application's AndroidManifest.xml file does not contain a uses-permission or uses-permission-sdk-23 tag containing android:name="android.permission.INTERNET". In this case, it is not necessary to perform the above Tests 1 and 2, as the platform will not allow the application to perform any network communication.
Pass/Fail with Explanation	NA. platform is Microsoft Windows and not Android.

7.1.30 FIA_X509_EXT.1.1 Test #1

Item	Data
Test Assurance Activity	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.
Test Steps	• The evaluator shall create a chain of four certificates using the XCA tool. • The evaluator shall configure the TOE's verification mode to strict. • **By establishing a certificate path in which one of the issuing certificates is not a CA certificate** ○ The evaluator uses the XCA tool to ensure that one of the issuing certificates ICA2 in the certificate path is not a CA certificate by transforming the original ICA2 issuing certificate and making it into a non-CA certificate. ○ The evaluator imports the Self-signed CA certificate Root_CA to the TOE's trust store. ○ The evaluator imports the Intermediate certificates ICA1 and ICA2 to the TOE's trust store. ○ The evaluator imports the server certificate to the TLS Server. ○ The evaluator uses openssl s_server command to listen on port 443 for TLS handshake with the client. ○ The evaluator tries to establish a connection to the TLS Server from the TOE and ensures it failed.

- The evaluator verifies the unsuccessful TLS connection with the help of packet capture.

- **By omitting the basicConstraints field in one of the issuing certificates**
  - The evaluator uses the XCA tool to ensure that one of the issuing certificates ICA2 in the certificate path does not have the basicConstraints by transforming the original ICA2 issuing certificate to a new ICA2 omitting the basicConstraints field.
  - The evaluator ensures that the Self-signed CA certificate Root_CA is present in the TOE's trust store.
  - The evaluator imports the Intermediate certificates ICA1 and ICA2 to the TOE's trust store.
  - The evaluator imports the server certificate to the TLS Server.
  - The evaluator uses openssl s_server command to listen on port 443 for TLS handshake with the client.
  - The evaluator tries to establish a connection to the TLS Server from the TOE and ensures it failed.
  - The evaluator verifies the unsuccessful TLS connection with the help of packet capture.

- **By setting the basicConstraints field in an issuing certificate to have CA=False**
  - The evaluator uses the XCA tool to ensure that one of the issuing certificates ICA2 in the certificate path to have CA=False.
  - The evaluator imports the Self-signed CA certificate Root_CA to the TOE's trust store.
  - The evaluator imports the Intermediate certificates ICA1 and ICA2 to the TOE's trust store.
  - The evaluator imports the server certificate to the TLS Server.
  - The evaluator uses openssl s_server command to listen on port 443 for TLS handshake with the client.
  - The evaluator tries to establish a connection to the TLS Server from the TOE and ensures it failed.
  - The evaluator verifies the unsuccessful TLS connection with the help of packet capture.

- **By omitting the CA signing bit of the key usage field in an issuing certificate**
  - The evaluator uses the XCA tool to ensure that one of the issuing certificates ICA2 in the certificate path does not have the CA signing bit of the key usage field by transforming the original ICA2 issuing certificate and not selecting the Certificate Sign on the key usage field.

	- The evaluator ensures that the Self-signed CA certificate Root_CA is present in the TOE's trust store. - The evaluator imports the Intermediate certificates ICA1 and ICA2 to the TOE's trust store. - The evaluator imports the server certificate to the TLS Server. - The evaluator uses openssl s_server command to listen on port 443 for TLS handshake with the client. - The evaluator tries to establish a connection to the TLS Server from the TOE and ensures it failed. - The evaluator verifies the unsuccessful TLS connection with the help of packet capture. - **By setting the path length field of a valid CA field to a value strictly less than the certificate path** - The evaluator uses the XCA tool to ensure that one of the issuing certificates (ICA1) in the certificate path has the path length field set to a value 0 that is strictly lesser than the certificate path. i.e., a CA with a path length constraint of zero cannot have any subordinate CAs. However, the ICA1 has a subordinate ICA2 while the path length is set to 0. - The evaluator ensures that the Self-signed CA certificate Root_CA is present in the TOE's trust store. - The evaluator imports the Intermediate certificates ICA1 and ICA2 to the TOE's trust store. - The evaluator imports the server certificate to the TLS Server. - The evaluator uses openssl s_server command to listen on port 443 for TLS handshake with the client. - The evaluator tries to establish a connection to the TLS Server from the TOE and ensures it failed. - The evaluator verifies the unsuccessful TLS connection with the help of packet capture. **Valid certificate chain** - The evaluator used the XCA tool to create a valid 4-length chain certificate with the node certificate as 10.1.3.212, the two Intermediate CAs as ICA1 and ICA2, and the self-signed Root CA certificate as Root_CA. - The evaluator ensures that the Self-signed CA certificate Root_CA is present in the TOE's trust store. - The evaluator imports the Intermediate certificates ICA1 and ICA2 to the TOE's trust store. - The evaluator imports the server certificate to the TLS Server.

	○ The evaluator uses openssl s_server command to listen on port 443 for TLS handshake with the client. ○ The evaluator tries to establish a connection to the TLS Server from the TOE and ensures it succeeds. ○ The evaluator verifies the successful TLS connection with the help of packet capture. **Invalid certificate chain** ○ The evaluator removes the Intermediate certificate ICA1 and ensures that only the Intermediate certificate ICA2 is present on the TLS Server's certificate trust store. ○ The evaluator uses openssl s_server command to listen on port 443 for TLS handshake with the client. ○ The evaluator tries to establish a connection to the TLS Server from the TOE and ensures it failed. ○ The evaluator further verifies the logs on the underlying windows platform and makes sure that the TOE is not able to find the ICA1 and Root_CA certificates from the chain. ○ The evaluator verifies the unsuccessful TLS connection with the help of packet capture.
Expected Test Results	• Screenshot evidence of the TOE rejecting the TLS connection without a valid certificate path. • Screenshot evidence of the TOE successfully establishing the TLS connection with a valid certificate path. • Screenshot evidence of the packet capture demonstrating failed TLS connection without a valid certification path. • Screenshot evidence of the packet capture demonstrating successful TLS connection with a valid certificate path.
Pass/Fail with Explanation	Pass. The TOE will not validate a certificate without a valid certification path, but it will accept that same certificate when it has the valid Certificate chain. This meets the testing requirements.

7.1.31 FIA_X509_EXT.1.1 Test #2

Item	Data
Test Assurance Activity	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 2: The evaluator shall demonstrate that validating an expired certificate results in the function failing.
Test Steps	• The evaluator uses the XCA tool to create an expired certificate that expired on 02 August 2022 4:50:00 • The evaluator imports the self-signed CA certificate (Root_CA) to the TOE's trust store. • The evaluator imports the intermediate certificates (Root_ICA1 and Root_ICA2) to the TOE's trust store. • The evaluator imports the server certificate (10.1.3.212_Exp) to the TLS server. • Use Acumen-tlsc tool to execute this test. • Attempt a connection from the TOE to the TLS Server and verify that the connection fails. • Verify the modified bytes from logs of the Acumen TLSC tool. Verify from PCAP that the TOE rejects the connection attempt by sending a RST packet to the TLS Server
Expected Test Results	• Screenshot evidence of the expired server certificate created within the 4-length chain certificate. • Screenshot evidence of the mmc console showing that the certificates are placed on their required paths. • Screenshot evidence of the TOE rejecting the TLS connection. • Screenshot evidence of the logs captured on the Underlying windows platform. • Evidence of the packet capture showing the unsuccessful TLS connection.
Pass/Fail with Explanation	Pass. The TOE does not validate an expired certificate and the TLS connection failed. This meets the testing requirements.

7.1.32 FIA_X509_EXT.1.1 Test #3

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

	o The evaluator shall test revocation of the node certificate. o 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 RFC 6066 is the only supported revocation method, this test is omitted. o 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.
Test Steps	**Note: OCSP option is not selected.** **For CRL** • **Attempt a connection with the valid certificates** o The evaluator uses the XCA tool to generate the 4-length chain certificates. o The evaluator generates the CRL's using the XCA tool. o The evaluator ensures that the self-signed CA certificate (RootCA) is present in the TOE's trust store. o The evaluator imports the intermediate certificates (Root_ICA1 and Root_ICA2) to the TOE's trust store. o The evaluator imports the server certificate to the TLS server. o The evaluator imports the CRL's to the CRL server. o The evaluator uses openssl s_server command to listen on port 443 for TLS handshake with the client. o The evaluator shall try to establish a connection to the TLS Server from the TOE and ensure it succeeds. o The evaluator verifies on the CRL server that the TOE tries to fetch the CRL's. o The evaluator verifies the successful TLS connection with the help of packet capture. • **Attempt a connection with a revoked server certificate** o The evaluator uses the XCA tool to revoke the server certificate on the 4-length chain certificate. o The evaluator generates the CRL's using the XCA tool. o The evaluator ensures that the self-signed CA certificate (RootCA) is present in the TOE's trust store. o The evaluator imports the intermediate certificates (Root_ICA1 and Root_ICA2) to the TOE's trust store. o The evaluator imports the server certificate to the TLS server. o The evaluator imports the CRL's to the CRL server. o The evaluator uses the python command to run the CRL web server. o The evaluator uses openssl s_server command to listen on port 443 for TLS handshake with the client. o The evaluator shall try to establish a connection to the TLS Server from the TOE and ensure it fails.

	○ The evaluator verifies on the CRL server that the TOE tries to fetch the CRL's. ○ The evaluator verifies the unsuccessful TLS connection with the help of packet capture. - **Attempt a connection with a revoked Intermediate CA certificate** ○ The evaluator now unrevoked the node(server) certificate and revokes the intermediate certificate Root_ICA2. ○ The evaluator generates the CRL's using the XCA tool. ○ The evaluator imports the self-signed CA certificate (RootCA) to the TOE's trust store. ○ The evaluator imports the intermediate certificates (Root_ICA1 and Root_ICA2) to the TLS Server. ○ The evaluator imports the server certificate to the TLS server. ○ The evaluator imports the CRL's to the CRL server. ○ The evaluator uses the python command to run the CRL web server. ○ The evaluator uses openssl s_server command to listen on port 443 for TLS handshake with the client. ○ The evaluator shall try to establish a connection to the TLS Server from the TOE and ensure it fails. ○ The evaluator verifies on the CRL server that the TOE tries to fetch the CRL's. ○ The evaluator verifies the unsuccessful TLS connection with the help of packet capture.
Expected Test Results	- Screenshot evidence of the generated 4-length chain certificate. - Screenshot evidence of intermediate certificates uploaded to TOE's trust store. - Screenshot of packet capture indicating that the TOE successfully connects to the server when a valid server certificate is used. - Screenshot of packet capture indicating that the TOE fails to connect to the server when a revoked server certificate is used. - Screenshot of packet capture indicating that the TOE fails to connect to the server when a revoked intermediate certificate is used.
Pass/Fail with Explanation	Pass. The TOE successfully connects to the server when a valid server certificate is used. The TOE fails to connect to the server when a revoked server certificate is used. The TOE fails to connect to the server when a revoked intermediate certificate is used. This meets the testing requirement.

7.1.33 FIA_X509_EXT.1.1 Test #4

Item	**Data**
Test Assurance Activity	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 4: If any OCSP option is selected, the evaluator shall configure the TSF to reject certificates if it cannot access valid status information, if so configurable. Then the evaluator shall ensure the TSF has no other source of revocation information available and configure the OCSP server or use a man-in-the-middle tool to present an OCSP response signed by a certificate that does not have the OCSP signing purpose and which is the only source of revocation status information advertised by the CA issuing the certificate being validated. The evaluator shall verify that validation of the OCSP response fails and that the TOE treats the certificate being checked as invalid and rejects the connection. If CRL is selected, the evaluator shall likewise configure the CA to be the only source of revocation status information, and sign a CRL with a certificate that does not have the cRLsign key usage bit set. The evaluator shall verify that validation of the CRL fails and that the TOE treats the certificate being checked as invalid and rejects the connection. Note: The intent of this test is to ensure a TSF does not trust invalid revocation status information. A TSF receiving invalid revocation status information from the only advertised certificate status provider should treat the certificate whose status is being checked as invalid. This should generally be treated differently from the case where the TSF is not able to establish a connection to check revocation status information, but it is acceptable that the TSF ignore any invalid information and attempt to find another source of revocation status (another advertised provider, a locally configured provider, or cached information) and treat this situation as not having a connection to a valid certificate status provider. **TD0780 has been applied**
Test Steps	Note: OCSP option is not selected. **CRL:** • The evaluator uses the XCA tool to create a 4-length chain certificates where the intermediate certificate ICA2 does not have the CRLsign key usage bit set. • The evaluator generates the CRL's using the XCA tool. • The evaluator imports the self-signed CA certificate (CA.crt) to the TOE's trust store. • The evaluator imports the intermediate certificates (ICA1.crt and ICA2.crt) to the TOE's trust store. • The evaluator imports the server certificate to the TLS server. • The evaluator imports the CRL's to the CRL server.

	- The evaluator uses openssl s_server command to listen on port 443 for TLS handshake with the client. - The evaluator shall try to establish a connection to the TLS Server from the TOE and ensure it fails. - The evaluator verifies on the CRL server that the TOE tries to fetch the CRL's. - The evaluator verifies the unsuccessful TLS connection with the help of packet capture.
Expected Test Results	- Screenshot evidence of the generated 4-length chain certificate. - Screenshot evidence of intermediate certificates uploaded to TOE's trust store. - Screenshot of packet capture showing the validation of the CRL fails if CRL is signed with a certificate that does not have cRLsign key usage
Pass/Fail with Explanation	Pass. The TOE treats the certificate being checked as invalid and rejects the connection, when a CRL is signed with a certificate that does not have the cRLsign key usage bit set. This meets the testing requirement.

7.1.34 FIA_X509_EXT.1.1 Test #5

Item	Data
Test Assurance Activity	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 5: The evaluator shall modify any byte in the first eight bytes of the certificate and demonstrate that the certificate fails to validate. (The certificate will fail to parse correctly.)
Test Steps	- The evaluator shall create a chain of four certificates using the XCA tool. - The evaluator ensures that the Self-signed CA certificate Root_CA is present in the TOE's trust store. - The evaluator imports the Intermediate certificates Root_ICA1 and Root_ICA2 to the TOE's trust store. - The evaluator imports the server certificate to the TLS Server. - Use Acumen-tlsc tool to execute this test. - Attempt a connection from the TOE to the TLS Server and verify that the connection fails. - Verify the modified bytes from logs of the Acumen TLSC tool. - Verify from PCAP that the TOE rejects the connection attempt by sending a RST packet to the TLS Server
Expected Test Results	- Screenshot evidence of the 4-chain length certificate created using XCA tool.

	- Screenshot evidence of the certmgr showing that the certificates are placed on their required paths. - Screenshot evidence of the TOE rejecting the TLS connection. - Screenshot evidence of the packet capture showing the unsuccessful TLS connection. - Screenshot evidence of the acumen-tlsc tool showing the modified first eight bytes of the certificate.
Pass/Fail with Explanation	Pass. The TOE rejects connections when a byte in the first 8 bytes of the certificate is modified. This meets the testing requirements.

7.1.35 FIA_X509_EXT.1.1 Test #6

Item	**Data**
Test Assurance Activity	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 6: The evaluator shall modify any byte in the last byte of the certificate and demonstrate that the certificate fails to validate. (The signature on the certificate will not validate.)
Test Steps	- The evaluator shall create a chain of four certificates using the XCA tool. - The evaluator ensures that the Self-signed CA certificate Root_CA is present in the TOE's trust store. - The evaluator imports the Intermediate certificates Root_ICA1 and Root_ICA2 to the TOE's trust store. - The evaluator imports the server certificate to the TLS Server. - Use Acumen-tlsc tool to execute this test. - Attempt a connection from the TOE to the TLS Server and verify that the connection fails. - Verify the modified bytes from the output of the Acumen TLSC tool. - Verify from PCAP that the TOE rejects the connection attempt.
Expected Test Results	- Screenshot evidence of the 4-chain length certificate created using XCA tool. - Screenshot evidence of the certmgr showing that the certificates are placed on their required paths. - Screenshot evidence of the TOE rejecting the TLS connection. - Screenshot evidence of the packet capture showing the unsuccessful TLS connection. - Screenshot evidence of the acumen-tlsc tool showing the modified last bytes of the certificate.

Pass/Fail with Explanation	Pass. The TOE rejects connections when the last byte of the certificate is modified. This meets the testing requirements.

7.1.36 FIA_X509_EXT.1.1 Test #7

Item	Data
Test Assurance Activity	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 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.)
Test Steps	• The evaluator shall create a chain of four certificates using the XCA tool. • The evaluator ensures that the Self-signed CA certificate Root_CA is present in the TOE's trust store. • The evaluator imports the Intermediate certificates Root_ICA1 and Root_ICA2 to the TOE's trust store. • The evaluator imports the server certificate to the TLS Server. • Use Acumen-tlsc tool to execute this test. • Attempt a connection from the TOE to the TLS Server and verify that the connection fails. • Verify the modified public key bytes from the output of the Acumen TLSC tool. • Verify from PCAP that the TOE rejects the connection attempt.
Expected Test Results	• Screenshot evidence of the 4-chain length certificate created using XCA tool. • Screenshot evidence of the certmgr showing that the certificates are placed on their required paths. • Screenshot evidence of the TOE rejecting the TLS connection. • Screenshot evidence of the packet capture showing the unsuccessful TLS connection. • Screenshot evidence of the acumen-tlsc tool showing the modified public key bytes of the certificate.
Pass/Fail with Explanation	Pass. The TOE rejects connections when the public key of the certificate is modified. This meets the testing requirements.

7.1.37 FIA_X509_EXT.1.1 Test #8

Item	Data
Test Assurance Activity	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 8: (**Conditional on support for EC certificates as indicated in FCS_COP.1/Sig**). The evaluator shall establish a valid, trusted certificate chain consisting of an EC leaf certificate, an EC Intermediate CA certificate not designated as a trust anchor, and an EC certificate designated as a trusted anchor, where the elliptic curve parameters are specified as a named curve. The evaluator shall confirm that the TOE validates the certificate chain.
Pass/Fail with Explanation	NA. ECDSA schemes are claimed in FCS_COP.1/Sig, however, they are only used by the SSH implementation for public key authentication. The support for EC certificates is not claimed in the ST.

7.1.38 FIA_X509_EXT.1.1 Test #9

Item	Data
Test Assurance Activity	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 9: (**Conditional on support for EC certificates as indicated in FCS_COP.1/Sig**). The evaluator shall replace the intermediate certificate in the certificate chain for Test 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.
Pass/Fail with Explanation	NA. ECDSA schemes are claimed in FCS_COP.1/Sig, however, they are only used by the SSH implementation for public key authentication. The support for EC certificates is not claimed in the ST.

7.1.39 FIA_X509_EXT.1.2 Test #1

Item	Data
Test Assurance Activity	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. The evaluator shall ensure that the certificate of at least one of the CAs 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.
Pass/Fail with Explanation	Pass. Section (i) and (ii) of this test was performed in conjunction with FIA_X509_EXT.1.1 Test #1 by omitting the basicConstraints field in one of the issuing certificates and ensuring the connection fails.

7.1.40 FIA_X509_EXT.1.2 Test #2

Item	Data
Test Assurance Activity	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. The evaluator shall ensure that the certificate of at least one of the CAs 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

Pass/Fail with Explanation	Pass. Section (i) and (ii) of this test was performed in conjunction with FIA_X509_EXT.1.1 Test #1 by setting the basicConstraints field in an issuing certificate to have CA=False and ensuring the connection fails.

7.1.41 FIA_X509_EXT.2.2 Test #1

Item	**Data**
Test Assurance Activity	The evaluator shall perform the following test for each trusted channel: The evaluator shall demonstrate that using a valid certificate that requires certificate validation checking to be performed in at least some part by communicating with a non-TOE IT entity. The evaluator shall then manipulate the environment so that the TOE is unable to verify the validity of the certificate, and observe that the action selected in FIA_X509_EXT.2.2 is performed. If the selected action is administrator-configurable, then the evaluator shall follow the operational guidance to determine that all supported administrator-configurable options behave in their documented manner.
Test Steps	Note1: The TOE only claims TLS as a trusted channel to support authentication using X.509v3 certificates as defined by RFC 5280. Note 2: The selection in FIA_X509_EXT.2.2 is "not accept the certificate". 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 uses the XCA tool to create a valid 4-length chain certificate with the node certificate as Server_cert, the two Intermediate CAs as ICA1 and ICA2, and the self-signed Root CA certificate as CA. • The evaluator generates the CRL's using the XCA tool. • The evaluator imports the self-signed CA certificate (CA) to the TOE's trust store. • The evaluator imports the intermediate certificates (ICA1 and ICA2) to the TOE's trust store. • The evaluator configures the TOE to check for CRL • The evaluator imports the server certificate to the TLS server. • The evaluator imports the CRL's to the CRL server. • The evaluator shall try to establish a connection to the TLS Server from the TOE and ensure it succeeds. o Listening on port 443 on TLS server o Initiating Connection from TOE WinAdmin Console o Connection status on TLS server • The evaluator verifies on the CRL server that the TOE tries to fetch the CRL's. • The evaluator verifies the successful TLS connection with the help of packet capture. Then manipulating the environment so that the TOE is unable to verify the validity of the certificate. • Disable the apache2 server on CRL VM. • The evaluator shall try to establish a connection to the TLS Server from the TOE and ensure it fails. o Listening on port 443 on TLS server o Initiating Connection from TOE WinAdmin Console

	○ Connection status on TLS server • The evaluator verifies the failed TLS connection with the help of packet capture.
Expected Test Results	• Screenshot evidence of the generated 4-length chain certificate. • Screenshot evidence of intermediate certificates uploaded to TOE's trust store. • Screenshot of packet capture showing the TOE does not accept the certificate, When environment is manipulated so that the TOE is unable to verify the validity of the certificate.
Pass/Fail with Explanation	Pass. The TOE checks for CRL on CRL server(non-TOE IT entity) and requires certificate validation checking to be performed. When environment is manipulated so that the TOE is unable to verify the validity of the certificate, the TOE does not accept the certificate which meets the requirement selected in FIA_X509_EXT.2.2.

7.1.42 FIA_X509_EXT.2.2 Test #2

Item	**Data**
Test Assurance Activity	The evaluator shall perform the following test for each trusted channel: 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.
Pass/Fail with Explanation	Note: The TOE only claims TLS as a trusted channel to support authentication using X.509v3 certificates as defined by RFC 5280. Pass. This test is performed in conjunction with FIA_X509_EXT.1.1 Test #3 and FIA_X509_EXT.1 Test#4. The connection is rejected when an invalid certificate is presented. This meets the testing requirements.

7.1.43 FMT_CFG_EXT.1.1 Test #1

Item	**Data**
Test Assurance Activity	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.
Pass/Fail with Explanation	N/A. The TOE does not support default credentials.

7.1.44 FMT_CFG_EXT.1.1 Test #2

Item	**Data**
Test Assurance Activity	If the application uses any default credentials the evaluator shall run the following tests. **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.
Pass/Fail with Explanation	N/A. The TOE does not support default credentials.

7.1.45 FMT_CFG_EXT.1.1 Test #3

Item	Data
Test Assurance Activity	If the application uses any default credentials the evaluator shall run the following tests. **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.
Pass/Fail with Explanation	N/A. The TOE does not support default credentials.

7.1.46 FMT_CFG_EXT.1.2 Test #1

Item	Data
Test Assurance Activity	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. **Platforms:Microsoft Windows...** The evaluator shall run the SysInternals tools, Process Monitor and Access Check (or tools of equivalent capability, like icacls.exe) for Classic Desktop applications to verify that files written to disk during an application's installation have the correct file permissions, such that a standard user cannot modify the application or its data files. For Windows Universal Applications the evaluator shall consider the requirement met because of the AppContainer sandbox.
Test Steps	• Start SysInternals tools and then run the TOE application after installation. • View ProcMon on the TOE machine and verify the permissions. • Run the tool accesscheck and verify the files permission.
Expected Test Results	• Check the different services the application is running have the correct file permissions.
Pass/Fail with Explanation	Pass. Files associated with the TOE have the correct file permissions, and are owned by the administrative user and therefore cannot be modified by a non-administrator.

7.1.47 FMT_MEC_EXT.1.1 Test #1

Item	Data
Test Assurance Activity	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: **Platforms:Microsoft Windows...** The evaluator shall determine and verify that Windows Universal Applications use either the Windows.Storage namespace, Windows.UI.ApplicationSettings namespace, or the IsolatedStorageSettings namespace for storing application specific settings. For .NET applications, the evaluator shall determine and verify that the application uses one of the locations listed in https://docs.microsoft.com/en-us/dotnet/framework/configure-apps/ for storing application specific settings. For Classic Desktop applications, the evaluator shall run the application while monitoring it with the SysInternals tool ProcMon and make changes to its configuration. The

	evaluator shall verify that ProcMon logs show corresponding changes to the the Windows Registry or C:\ProgramData\ directory.
Test Steps	- Run the TOE application. - Start ProcMon on the TOE machine. - Make Changes to TOE configuration. - Verify that ProcMon logs show corresponding changes to the Windows Registry or C:\ProgramData\ directory.
Expected Test Results	- Screenshot evidence of ProcMon logs showing corresponding changes to the Windows Registry or C:\ProgramData\ directory.
Pass/Fail with Explanation	Pass. ProcMon logs show corresponding changes to the Windows Registry. This meets the testing requirements.

7.1.48 FMT_MEC_EXT.1.1 Test #2

Item	Data
Test Assurance Activity	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.
Pass/Fail with Explanation	NA. "implement functionality to encrypt and store configuration options as defined by FDP_PRT_EXT.1 in the PP-Module for File Encryption" is not selected in ST.

7.1.49 FMT_SMF.1.1 Test #1

Item	Data
Test Assurance Activity	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.
Test Steps	- Logging: Start Venafi Configuration Console found in Venafi\Platform\Venafi Configuration Console.msc. Enable/Disable the Logging. - Stack Traces: Navigate to C:\Program Files\Venafi\Web\Aperture\API\web.conf file. Change the “showStackTrace” value from 0 to 1. This enables Stack Traces for the TOE. - Enable/Disable Service Modules: Navigate to Platform Tree. Select any Service/Module and check /uncheck on Disabled. - Web Applications: Start Venafi Configuration Console found in Venafi\Platform\Venafi Configuration Console.msc. Enable/Disable the IIS website.
Expected Test Results	The TOE can be configured as stated in the ST and Guidance documentation.
Pass/Fail with Explanation	Pass. The evaluator verified that the TOE can be configured as stated in the ST and Guidance documentation.

7.1.50 FPR_ANO_EXT.1.1 Test #1

Item	Data
Test Assurance Activity	If **require user approval before executing** is selected, the evaluator shall run the application and exercise the functionality responsibly for transmitting PII and verify that user approval is required before transmission of the PII.
Pass/Fail with Explanation	Pass. “require user approval before executing” is not selected and, TOE does not transmit any PII. Therefore, this test is considered satisfied.

7.1.51 FPT_AEX_EXT.1.1 Test #1

Item	Data
Test Assurance Activity	The evaluator shall perform either a static or dynamic analysis to determine that no memory mappings are placed at an explicit and consistent address except for any exceptions claimed in the SFR. For these exceptions, the evaluator shall verify that this analysis shows explicit mappings that are consistent with what is claimed in the TSS. 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. **Platforms:Microsoft Windows...** The evaluator shall run the same application on two different Windows systems and run a tool that will list all memory mapped addresses for the application. The evaluator shall then verify the two different instances share no mapping locations. The Microsoft SysInternals tool, VMMap, could be used to view memory addresses of a running application. The evaluator shall use a tool such as Microsoft's BinScope Binary Analyzer to confirm that the application has ASLR enabled. **TD0798 has been applied**
Test Steps	• Start the application on two separate platforms. • Windows Server 2016 (Platform 1) ○ The evaluator shall run the VMMap on the TOE and verify the memory mapping. ○ The evaluator shall run the BinScope and perform a DBCheck to verify that ASLR was enabled. ○ Verify the report of BinScope check and ensure no failures are indicated. • Windows Server 2016 (Platform 2) ○ The evaluator shall run the VMMap on the TOE and verify the memory mapping.
Expected Test Results	• The application running on two different machines should not share memory mapping location. • Screenshots of VMMap tool showing that two different instances do not share mapping locations. • Screenshot of the Binscope check result which shows no failed DBcheck to verify ASLR is enabled.

Pass/Fail with Explanation	Pass. Positions in address space for the TOE were different on identical systems running the same version of the TOE software. BinScope check showed that ASLR is enabled on the TOE.

7.1.52 FPT_AEX_EXT.1.2 Test #1

Item	Data
Test Assurance Activity	The evaluator shall verify that no memory mapping requests are made with write and execute permissions. The method of doing so varies per platform. **Platforms:Microsoft Windows...** The evaluator shall use a tool such as Microsoft's BinScope Binary Analyzer to confirm that the application passes the NXCheck. The evaluator may also ensure that the /NXCOMPAT flag was used during compilation to verify that DEP protections are enabled for the application.
Test Steps	• Run the BinScope with NXCheck option to verify the correct usage of /NX. • Verify the report of BinScope check and ensure no failures are indicated. • Run Wumpbin.exe and look under the OPTIONAL HEADER section for the NX compatible flag.
Expected Test Results	• Screenshot evidence of the BinScope result ensuring no failed NXCheck. • Screenshot evidence of the OPTIONAL HEADER section ensuring /NXCOMPAT flag is used.
Pass/Fail with Explanation	Pass. Microsoft BinScope states that there are no failed checks which means that the NXCheck has passed. Wumpbin also verified that /NXCOMPACT flag was used during compilation to verify that DEP protections are enabled for the application.

7.1.53 FPT_AEX_EXT.1.3 Test #1

Item	Data
Test Assurance Activity	The evaluator shall configure the platform in the ascribed manner and carry out one of the prescribed tests: **Platforms:Microsoft Windows...** If the OS platform supports Windows Defender Exploit Guard (Windows 10 version 1709 or later), then the evaluator shall ensure that the application can run successfully with Windows Defender Exploit Guard Exploit Protection configured with the following minimum mitigations enabled; Control Flow Guard (CFG), Randomize memory allocations (Bottom-Up ASLR), Export address filtering (EAF), Import address filtering (IAF), and Data Execution Prevention (DEP). The following link describes how to enable Exploit Protection, https://learn.microsoft.com/en-us/microsoft-365/security/defender-endpoint/enable-exploit-protection?view=o365-worldwide. If the OS platform supports the Enhanced Mitigation Experience Toolkit (EMET) which can be installed on Windows 10 version 1703 and earlier, then the evaluator shall ensure that the application can run successfully with EMET configured with the following minimum mitigations enabled; Memory Protection Check, Randomize memory allocations (Bottom-Up ASLR), Export address filtering (EAF), and Data Execution Prevention (DEP). The application shall be compatible with security features provided by the platform vendor.

	TD0823 has been applied
Test Steps	- Stop the TOE services. - Download, install and configure EMET on TOE's underlying platform. - Start the TOE services after EMET configuration and verify that TOE is running.
Expected Test Results	Screenshot of TOE running along with EMET configuration enabled with following: Memory Protection Check, Randomize memory allocations (Bottom-Up ASLR), Export address filtering (EAF), and Data Execution Prevention (DEP).
Pass/Fail with Explanation	Pass. TOE runs successfully when EMET is configured.

7.1.54 FPT_AEX_EXT.1.4 Test #1

Item	**Data**
Test Assurance Activity	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: **Platforms:Microsoft Windows...** For Windows Universal Applications the evaluator shall consider the requirement met because the platform forces applications to write all data within the application working directory (sandbox). For Windows Desktop Applications 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.
Test Steps	- Run the TOE application. - Start ProcMon on the TOE machine. ○ Adjust/Set the filter to record any file system activity, and any process creation activity/operation. - Operate TOE so that user-modifiable files are written. ○ Login to TOE web GUI. ○ Create a policy by navigating to "Policy Tree" tab and add a user credential. - Note where all user-modifiable files are written. ○ Ensure that executable files are not stored in the same directories to which the application wrote user-modifiable files.
Expected Test Results	No executable files should be stored in the same directories to which the application wrote user-modifiable files.
Pass/Fail with Explanation	Pass. The evaluator executed a few tasks to mimic the normal usage of the TOE, and verified that the user-modifiable files are not written in the same directory where the executable files are stored. This meets testing requirements.

7.1.55 FPT_AEX_EXT.1.5 Test #1

Item	**Data**
Test Assurance Activity	The evaluator will inspect every native executable included in the TOE to ensure that stack-based buffer overflow protection is present.

	Platforms:Microsoft Windows... Applications that run as Managed Code in the .NET Framework do not require these stack protections. Applications developed in Object Pascal using the Delphi IDE compiled with RangeChecking enabled comply with this element. For other code, the evaluator shall review the TSS and verify that the /GS flag was used during compilation. The evaluator shall run a tool like, BinSkim, that can verify the correct usage of /GS. **For PE** , the evaluator will disassemble each and ensure the following sequence appears: mov rcx, QWORD PTR [rsp+(...)] xor rcx, (...) call (...) **For ELF executables**, the evaluator will ensure that each contains references to the symbol __stack_chk_fail. Tools such as Canary Detector may help automate these activities. If these automated tests fail, the evaluator shall perform the above, conditional TSS activity. **TD0815 has been applied**
Pass/Fail with Explanation	Pass. The TOE's code is not run natively, but instead as managed code on top of Microsoft's .Net. Applications that run as Managed Code in the .NET Framework do not require these stack protections. This meets the testing requirements.

7.1.56 FPT_API_EXT.1.1 Test #1

Item	**Data**
Test Assurance Activity	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.
Test Steps	• The evaluator verified that the TSS lists the platform APIs used in the application. • The evaluator then compared 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.
Expected Test Results	Ensure that all APIs listed in the TSS are supported
Pass/Fail with Explanation	Pass. all APIs listed in the TSS are supported.

7.1.57 FPT_IDV_EXT.1.1 Test #1

Item	**Data**
Test Assurance 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.
Test Steps	• Run the application (TOE). • Check for version information.

	- Verify from documentation correct version.
Expected Test Results	Screenshot Evidence with Correct version information is displayed.
Pass/Fail with Explanation	Pass. Correct version information of the TOE is displayed.

7.1.58 FPT_LIB_EXT.1.1 Test #1

Item	**Data**
Test Assurance 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.
Test Steps	• Survey the installation directory of the application for dynamic libraries. • Verify the libraries employed with the application are limited to those in the assignment.
Expected Test Results	As per TSS, The TOE shall be packaged with below Third party libraries: Libraries found: • IronPython • Chaos.NaCl • Microsoft Intune CSR Validation • Sustainsys Saml2 • Excelsior JET • F5 iControl Assembly for .NET • Bootstrap • Backbone • Underscore • Jquery • date.js • dateRangePicker.js, dateRangePicker.css • moment.js • easyDate.js • maskedInput.js • browser.js • jquery.timepicker.js • Select2.js • moment-timezone.js • core.js • dropzone.js • JSON.Net • ASP.NET Web Stack • Sencha Ext JS • Tigra Calendar • Pretty-Print JSON • D3.js • chart.js • mustache.js
Pass/Fail with Explanation	Pass. Libraries found to be packaged with or employed by the application are limited to those in the assignment.

7.1.59 FPT_TUD_EXT.1.1 Test #1

Item	Data
Test Assurance Activity	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.
Test Steps	From Web App: • Start the TOE. • Click on About. • Click on Upgrade Status. From Console: • Start the TOE (Venafi Trust Protection Platform Console). • Click on Help. • Click on About Console.
Expected Test Results	TOE is updated or no update is available.
Pass/Fail with Explanation	Pass. Followed procedure to check for an update and verified that the application does not issue an error, no update is available, and the latest version of TOE is being used. This meets testing requirements.

7.1.60 FPT_TUD_EXT.1.2 Test #1

Item	Data
Test Assurance Activity	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.
Pass/Fail with Explanation	Pass. This test is performed in conjunction with FPT_IDV_EXT.1.1 Test #1.

7.1.61 FPT_TUD_EXT.1.3 Test #1

Item	Data
Test Assurance Activity	The evaluator shall verify that the application's executable files are not changed by the application. **For all other platforms**, the evaluator shall perform the following test: 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.
Test Steps	• Install the TOE and generate a hash of all the executable files and save it to tud_ext.1.3_t1_hash_before_run.txt • Run the TOE and exercise all features as described in the ST. • Generate a hash of all the executable files again and save it to tud_ext.1.3_t1_hash_after_run.txt.

	- Verify that both the files are identical.
Expected Test Results	Both hash files before and after execution of TOE functions need to be identical.
Pass/Fail with Explanation	Pass. The evaluator verified that all the executable files are identical. This meets the testing requirements.

7.1.62 FPT_TUD_EXT.1.4 Test #1

None.

7.1.63 FPT_TUD_EXT.1.5 Test #1

None.

7.1.64 FPT_TUD_EXT.2.1 Test #1

Item	Data
Test Assurance Activity	If a container image is claimed the evaluator shall verify that application updates are distributed as container images. If the format of the platform-supported package manager is claimed, the evaluator shall verify that application updates are distributed in the correct format. This varies per platform: **Platforms:Microsoft Windows...** The evaluator shall ensure that the application is packaged in the standard Windows Installer (.MSI) format, the Windows Application Software (.EXE) format signed using the Microsoft Authenticode process, or the Windows Universal Application package (.APPX) format. See https://msdn.microsoft.com/en-us/library/ms537364(v=vs.85).aspx for details regarding Authenticode signing. **TD0628 has been applied**
Test Steps	- Confirm that the TOE’s installation package comes in MSI format.
Expected Test Results	Screenshot evidence showing TOE's installation package is in MSI format.
Pass/Fail with Explanation	Pass. TOE’s installation package is in the MSI format.

7.1.65 FPT_TUD_EXT.2.2 Test #1

Item	Data
Test Assurance Activity	**All Other Platforms...** 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. **TD0664 has been applied**
Test Steps	- Record the path of every file on the entire filesystem by executing the command “dir /B /S > before_install.txt” before installing the TOE which saves the output to “before_install.txt”

	• Install and run the TOE. • Uninstall the TOE. • Record the path of every file on the entire filesystem by executing the command dir /B /S and save the output to after_uninstall.txt • Verify that no files, other than configuration, output, and audit/log files, have been added to the filesystem.
Expected Test Results	No files, other than configuration, output, and audit/log files, should be added to the filesystem, When comparing the resulting filesystem to the initial record.
Pass/Fail with Explanation	Pass. The evaluator verified that no other files, other than configuration, output, and audit/log files, have been added to the filesystem by the TOE. This meets testing requirements.

7.1.66 FPT_TUD_EXT.2.3 Test #1

None.

7.1.67 FTP_DIT_EXT.1.1 Test #1

Item	**Data**
Test Assurance Activity	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.
Pass/Fail with Explanation	Pass. This test is performed in conjunction with FDP_NET_EXT.1.1 Test #1, FCS_SSH_EXT.1.6 Test #1, and FIA_X509_EXT.1.1 Test#1 as explained below. **For connection to Managed Host** This verified in FDP_NET_EXT.1.1 Test #1 (subsection Accessing "Managed host" to capture traffic). In the pcap file "Managedhost", the traffic is encrypted with SSH and no sensitive data is transmitted in the clear. **For connection to User/Admin Authentication** This verified in FDP_NET_EXT.1.1 Test #1 (subsection Accessing TOE from Workstation (192.168.254.113) for User or Admin authentication via web-based console). In the pcap file "FDP_NET_EXT.1.1_Test#1_Authentication_IIS", the traffic is encrypted with TLS and no sensitive data is transmitted in the clear. **For connection to Remote Database** This verified in FDP_NET_EXT.1.1 Test #1. In the pcap file "FDP_NET_EXT.1.1_Test#1_with_VenafiTPP", after filtering the traffic just between the TOE (10.1.3.211) and the remote database (10.1.3.216), the traffic is encrypted with TLS and no sensitive data is transmitted in the clear. **For connection to Discovery Services** This verified in FDP_NET_EXT.1.1 Test #1 (subsection Performing User-Configured "Discovery Services"). In the pcap file "discovery", the traffic is encrypted with TLS and no sensitive data is transmitted in the clear.

	For test FCS_CKM.2.1 - RSA The TOE uses RSAES-PKCS1-v1_5 key establishment scheme in only TLS protocol. This verified in FIA_X509_EXT.1.1 Test#1 (subsection Valid certificate chain). In the pcap file “FIA_X509_EXT.1.1 Test #1_validchain”, the client hello packet show that the TOE supports TLS_RSA_WITH_AES_xxx ciphers, which uses known good implementation of RSAES-PKCS1-v1_5 key establishment scheme. **For test FCS_CKM.2.1 - FFC** The TOE uses FFC “safe-prime” groups key establishment scheme in TLS and SSH protocols. For TLS, this verified in FIA_X509_EXT.1.1 Test#1 (subsection Valid certificate chain). In the pcap file “FIA_X509_EXT.1.1 Test #1_validchain”, the client hello packet show that the TOE supports TLS_DHE_RSA_WITH_AES_xxx ciphers, which uses known good implementation of FFC “safe-prime” groups key establishment scheme. For SSH, this verified in FCS_SSH_EXT.1.6 Test #1. In the pcap file “FCS_SSH_EXT.1.4_Test #1”, the client key exchange init packet show that the TOE supports DH-14 (SHA-256), DH-16 (SHA-512), DH-18 (SHA-512), which uses known good implementation of FFC “safe-prime” groups key establishment scheme.

7.1.68 FTP_DIT_EXT.1.1 Test #2

Item	Data
Test Assurance Activity	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.
Pass/Fail with Explanation	Pass. This test is performed in conjunction with FTP_DIT_EXT.1.1 Test #1

7.1.69 FTP_DIT_EXT.1.1 Test #3

Item	Data
Test Assurance Activity	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.
Test Steps	For SSH • Set credentials to a known value on the TOE. • Attempt to establish a connection from the TOE to the SSH Server. • Verify that the credentials are not sent in plaintext and unable to find them in the Packet capture. For TLS • Attempt to establish a connection from the workstation to the TOE using known credentials. • Verify that the credentials are not sent in plaintext and unable to find them in the Packet capture.

Expected Test Results	The credentials should not be found in the PCAP.
Pass/Fail with Explanation	Pass. The evaluator performed a string search of the captured network packets and verify that the plaintext credential previously set by the evaluator is not found. This meets testing requirements.

7.1.70 FTP_DIT_EXT.1.1 Test #4

Item	Data
Test Assurance Activity	**Platforms:Android...** If "**not transmit any data**" is selected, the evaluator shall ensure that the application's AndroidManifest.xml file does not contain a uses-permission or uses-permission-sdk-23 tag containing android:name="android.permission.INTERNET". In this case, it is not necessary to perform the above Tests 1, 2, or 3, as the platform will not allow the application to perform any network communication.
Pass/Fail with Explanation	NA. Platform is Microsoft Windows and not Android.

7.1.71 FTP_DIT_EXT.1.1 Test #5

Item	Data
Test Assurance Activity	**Platforms:Apple iOS...** If "**encrypt all transmitted data**" is selected, the evaluator shall ensure that the application's Info.plist file does not contain the NSAllowsArbitraryLoads or NSExceptionAllowsInsecureHTTPLoads keys, as these keys disable iOS's Application Transport Security feature.
Pass/Fail with Explanation	NA. Platform is Microsoft Windows and not Apple iOS.

7.2 PKG_SSH

7.2.1 FCS_SSH_EXT.1.1 Test #1

None.

7.2.2 FCS_SSH_EXT.1.2 Test #1

Item	Data
Test Assurance Activity	**[conditional] If the TOE is acting as SSH Server:** a) The evaluator shall use a suitable SSH Client to connect to the TOE, enable debug messages in the SSH Client, and examine the debug messages to determine that only the configured authentication methods for the TOE were offered by the server. b) **[conditional] If the SSH server supports X509 based Client authentication options:** a. The evaluator shall initiate an SSH session from a client where the username is associated with the X509 certificate. The evaluator shall verify the session is successfully established.

	b. Next the evaluator shall use the same X509 certificate as above but include a username not associated with the certificate. The evaluator shall verify that the session does not establish. c. Finally, the evaluator shall use the correct username (from step a above) but use a different X509 certificate which is not associated with the username. The evaluator shall verify that the session does not establish.
Pass/Fail with Explanation	NA. TOE is not a SSH Server and does not support X509 based Client authentication options.

7.2.3 FCS_SSH_EXT.1.2 Test #2

Item	Data
Test Assurance Activity	**[conditional] If the TOE is acting as SSH Client,** the evaluator shall test for a successful configuration setting of each authentication method as follows: a) The evaluator shall initiate a SSH session using the authentication method configured and verify that the session is successfully established. b) Next, the evaluator shall use bad authentication data (e.g. incorrectly generated certificate or incorrect password) and ensure that the connection is rejected. Steps a-b shall be repeated for each independently configurable authentication method supported by the server.
Test Steps	For Password based authentication. • Attempt SSH connection to the SSH server from the TOE o Add the credentials of the SSH server on the Web console of TOE for successful connection. o Configure the SSH client for connection to SSH server. o Click on connect on the WinAdmin Console. • Verify that the password was accepted and the connection was successful. o TOE logs. o Packet capture. For Password based authentication with incorrect password. • Attempt SSH connection to the SSH Server from the TOE o Enter the incorrect password of the SSH server on the Web console of TOE. o Configure the SSH client for connection to SSH server. o Click on connect on the WinAdmin Console. • Verify that the incorrect password was not accepted, and the connection was unsuccessful o TOE logs. o Packet Capture. For Public-key based authentication. ssh-rsa • Create a private/public key pair o Using ssh-keygen tool to generate the key pair. • Copy the public key to the SSH Server • Upload the private key to the TOE web console.

- Attempt to connect to the SSH Server from the TOE
  - Configure the TOE web console to select the SSH device credential as ssh-rsa.
  - Click on connect in WinAdmin console.

- Verify the connection succeeds.
  - TOE logs.
  - Packet capture.

Bad authentication data

- Modify the public key on the SSH Server
  - Change string “AAA” to “ABC”.
- Attempt to connect to the SSH Server from the TOE
- Verify that connection fails.
  - TOE logs.
  - Packet capture.

rsa-sha2-256

- Create a private/public key pair
  - Using ssh-keygen tool to generate the key pair.
- Copy the public key to the SSH Server
- Upload the private key to the TOE web console.
- Attempt to connect to the SSH Server from the TOE
  - Configure the TOE web console to select the SSH device credential as rsa-sha2-256.
  - Click on connect in WinAdmin console.
- Verify the connection succeeds.
  - TOE logs.
  - Packet capture.

Bad authentication data

- Modify the public key on the SSH Server
  - Change string “AAA” to “ABC”.
- Attempt to connect to the SSH Server from the TOE
- Verify that connection fails.
  - TOE logs.
  - Packet capture.

rsa-sha2-512

- Create a private/public key pair
  - Using ssh-keygen tool to generate the key pair.
- Copy the public key to the SSH server
- Upload the private key to the TOE web console.
- Attempt to connect to the SSH server from the TOE
  - Configure the TOE web console to select the SSH device credential as rsa-sha2-512.
  - Click on connect in WinAdmin console.
- Verify the connection succeeds.
  - TOE logs.
  - Packet capture.

Bad authentication data

- Modify the public key on the SSH Server
  - Change string “AAA” to “ABC”.
- Attempt to connect to the SSH Server from the TOE
- Verify that connection fails.
  - TOE Logs.
  - Packet capture.

ecdsa-sha2-nistp256

- Create a private/public key pair
  - Using ssh-keygen tool to generate the key pair.
- Copy the public key to the SSH server
- Upload the private key to the TOE web console.
- Attempt to connect to the SSH server from the TOE
  - Configure the TOE web console to select the SSH device credential as ecdsa-sha2-nistp256.
  - Click on connect in WinAdmin console.
- Verify the connection succeeds.
  - TOE logs.
  - Packet capture.

Bad authentication data

- Modify the public key on the SSH Server
  - Change string “AAA” to “ABC”.
- Attempt to connect to the SSH Server from the TOE
- Verify that connection fails.
  - TOE logs.
  - Packet capture.

ecdsa-sha2-nistp384

- Create a private/public key pair
  - Using ssh-keygen tool to generate the key pair.
- Copy the public key to the SSH server
- Upload the private key to the TOE web console.
- Attempt to connect to the SSH server from the TOE
  - Configure the TOE web console to select the SSH device credential as ecdsa-sha2-nistp384.
  - Click on connect in WinAdmin console.
- Verify the connection succeeds
  - TOE logs.
  - Packet capture.

Bad authentication data

Modify the public key on the SSH Server

  - Change string “AAA” to “ABC”.
- Attempt to connect to the SSH Server from the TOE
- Verify that connection fails.
  - TOE logs.
  - Packet capture.

ecdsa-sha2-nistp521

	• Create a private/public key pair ○ Using ssh-keygen tool to generate the key pair. • Copy the public key to the SSH server • Upload the private key to the TOE web console. • Attempt to connect to the SSH server from the TOE ○ Configure the TOE web console to select the SSH device credential as ecdsa-sha2-nistp521. ○ Click on connect in WinAdmin console. • Verify the connection succeeds ○ TOE logs. ○ Packet capture. Bad authentication data • Modify the public key on the SSH Server ○ Change string “AAA” to “ABC”. • Attempt to connect to the SSH Server from the TOE • Verify that connection fails. ○ TOE logs. ○ Packet Capture. ssh-ed25519 • Create a private/public key pair ○ Using ssh-keygen tool to generate the key pair. • Copy the public key to the SSH server • Upload the private key to the TOE web console. • Attempt to connect to the SSH server from the TOE ○ Configure the TOE web console to select the SSH device credential as ssh-rsa. ○ Click on connect in WinAdmin console. • Verify the connection succeeds ○ TOE logs. ○ Packet capture. Bad authentication data • Modify the public key on the SSH Server ○ Change string “AAA” to “ABC”. • Attempt to connect to the SSH Server from the TOE • Verify that connection fails. ○ TOE logs. ○ Packet Capture.
Expected Test Results	Screenshot evidence for successful connection when login with correct authentication data. Screenshot evidence for connection failure when login with bad authentication data
Pass/Fail with Explanation	Pass. The user connection from the TOE to the server is successfully authenticated with correct login credentials and the user connection from the TOE to the server with incorrect login credentials is rejected for each of the authentication methods. This satisfies the test requirements.

7.2.4 FCS_SSH_EXT.1.2 Test #3

Item	Data
Test Assurance Activity	**[conditional] If the TOE is acting as SSH Client,** the evaluator shall verify that the connection fails upon configuration mismatch as follows: The evaluator shall configure the Client with an authentication method not supported by the Server. The evaluator shall verify that the connection fails.
Test Steps	• Configure the TOE with an authentication method not supported by the SSH Server ○ Configuring the SSH server for GSSAPI authentication. • Initiate a SSH session from the TOE ○ Add credentials of the SSH server to the TOE ○ Create a policy on the TOE to connect to the SSH server. ○ Click on connect. • Verify that the connection fails ○ TOE logs. ○ Packet capture.
Expected Test Results	The SSH connection must fail upon using authentication method not supported by the SSH server.
Pass/Fail with Explanation	Pass. The TOE does not connect to the SSH Server if the authentication method is not supported by the Server. This satisfies the test requirements.

7.2.5 FCS_SSH_EXT.1.3 Test #1

Item	Data
Test Assurance Activity	The evaluator shall demonstrate that the TOE accepts the maximum allowed packet size.
Test Steps	• Add credentials of the SSH server to the TOE. • Create a policy on the TOE to connect to the SSH server. • Start the acumen-sshc tool on VM to send a packet of maximum allowed packet size to the TOE. • Initiate a connection from TOE to SSH server. • Verify that the TOE accepts the packet. ○ Packet capture.
Expected Test Results	TOE accepts the packet of maximum allowed packet size.
Pass/Fail with Explanation	Pass. The TOE accepts the maximum allowed packet size. This meets the testing requirements.

7.2.6 FCS_SSH_EXT.1.3 Test #2

Item	Data
Test Assurance Activity	This test is performed to verify that the TOE drops packets that are larger than size specified in the component. a. The evaluator shall establish a successful SSH connection with the peer. b. Next the evaluator shall craft a packet that is slightly larger than the maximum size specified in this component and send it through the established SSH

	connection to the TOE. The packet should not be greater than the maximum packet size + 16 bytes. If the packet is larger, the evaluator shall justify the need to send a larger packet. **c.** The evaluator shall verify that the packet was dropped by the TOE. The method of verification will vary by the TOE. Examples include reviewing the TOE audit log for a dropped packet audit or observing the TOE terminates the connection. **TD0732 has been applied**
Test Steps	• Add credentials of the SSH server to the TOE. • Create a policy on the TOE to connect to the SSH server. • Start the acumen-sshc tool on VM to send a packet slightly larger than the size specified for TOE. • Initiate the SSH connection from the TOE. • Verify that the packet is dropped by the TOE.
Expected Test Results	Packet capture evidence that TOE terminates the connection.
Pass/Fail with Explanation	Pass. The TOE terminates the connection when the packet size is larger than specified. The TOE satisfies the test requirements.

7.2.7 FCS_SSH_EXT.1.4 Test #1

Item	**Data**
Test Assurance Activity	If the TOE can be both a client and a server, these tests must be performed for both roles. Test 1: The evaluator must ensure that only claimed algorithms and cryptographic primitives are used to establish an SSH connection. To verify this, the evaluator shall establish an SSH connection with a remote endpoint. The evaluator shall capture the traffic exchanged between the TOE and the remote endpoint during protocol negotiation (e.g. using a packet capture tool or information provided by the endpoint, respectively). The evaluator shall verify from the captured traffic that the TOE offers only the algorithms defined in the ST for the TOE for SSH connections. The evaluator shall perform one successful negotiation of an SSH connection and verify that the negotiated algorithms were included in the advertised set. If the evaluator detects that not all algorithms defined in the ST for SSH are advertised by the TOE or the TOE advertises additional algorithms not defined in the ST for SSH, the test shall be regarded as failed. The data collected from the connection above shall be used for verification of the advertised hashing and shared secret establishment algorithms in FCS_SSH_EXT.1.5 and FCS_SSH_EXT.1.6 respectively.
Test Steps	Note the TOE is only SSH Client. • Configure the TOE to initiate a SSH connection to the SSH Server. ○ Add the credentials of the SSH server on the Web console of TOE for successful connection. ○ Configure the SSH client for connection to SSH server. • Configure the /etc/ssh/sshd_config file of the SSH server with aes128-ctr only . • Initiate a SSH connection from the TOE to the SSH Server.

	- Verify the TOE is successfully able to establish the connection with the SSH Server. ○ TOE logs. ○ Packet Capture.
Expected Test Results	Only the claimed algorithms and cryptographic primitives must be used to establish an SSH connection.
Pass/Fail with Explanation	Pass. The TOE is advertising only the claimed algorithms from ST. This meets the testing requirements.

7.2.8 FCS_SSH_EXT.1.4 Test #2

Item	Data
Test Assurance Activity	If the TOE can be both a client and a server, these tests must be performed for both roles. For the connection established in Test 1, the evaluator shall terminate the connection and observe that the TOE terminates the connection.
Test Steps	Note the TOE is only SSH Client. - Terminate the SSH session established in Test 1. ○ Click on disconnect. - Verify that connection was terminated.
Expected Test Results	For the connection established in Test 1, The TOE should be able to terminate the connection.
Pass/Fail with Explanation	Pass. The TOE successfully terminates the connection established in Test#1. This satisfies the testing requirement.

7.2.9 FCS_SSH_EXT.1.4 Test #3

Item	Data
Test Assurance Activity	If the TOE can be both a client and a server, these tests must be performed for both roles. The evaluator shall configure the remote endpoint to only allow a mechanism that is not included in the ST selection. The evaluator shall attempt to connect to the TOE and observe that the attempt fails.
Test Steps	Note the TOE is only SSH Client. - Configure the SSH server to only allow a mechanism that is not included in the ST selection. ○ Configuring the /etc/ssh/sshd_config file of the SSH server with aes192-ctr only. - From SSH server initiate a connection to the TOE. ○ Add the credentials of the SSH server on the Web console of TOE for successful connection. ○ Configure the SSH client for connection to SSH server. ○ Click on connect. - Verify that connection fails. ○ TOE logs. ○ Packet capture.

Expected Test Results	The connection to the TOE via SSH must fail, if the only mechanism allowed by the SSH server is not included in the ST selection.
Pass/Fail with Explanation	Pass. When a remote endpoint is only allowing a mechanism other than claimed in ST, the connection gets terminated. This meets the testing requirement.

7.2.10 FCS_SSH_EXT.1.5 Test #1

Item	**Data**
Test Assurance Activity	The evaluator shall use the test data collected in FCS_SSH_EXT.1.4, Test 1 to verify that appropriate mechanisms are advertised.
Test Steps	• Verify that appropriate mechanisms are advertised from packet capture collected in FCS_SSH_EXT.1.4, Test 1.
Expected Test Results	The TOE should advertise appropriate mechanisms selected in the ST.
Pass/Fail with Explanation	Pass. The TOE is advertising appropriate mechanisms as claimed in the ST.

7.2.11 FCS_SSH_EXT.1.5 Test #2

Item	**Data**
Test Assurance Activity	The evaluator shall configure an SSH peer to allow only a hashing algorithm that is not included in the ST selection. The evaluator shall attempt to establish an SSH connection and observe that the connection is rejected.
Test Steps	• Configure the SSH server to allow only a hashing algorithm that is not included in the ST • Initiate a SSH connection to the SSH server from the TOE • Verify that the connection failed
Expected Test Results	The SSH connection must fail when the hashing algorithm is different on the SSH server from the ST selection.
Pass/Fail with Explanation	Pass. The SSH connection is rejected when there is hashing algorithm used other than the one specified in the ST. This meets the testing requirement.

7.2.12 FCS_SSH_EXT.1.6 Test #1

Item	**Data**
Test Assurance Activity	The evaluator shall use the test data collected in FCS_SSH_EXT.1.4, Test 1 to verify that appropriate mechanisms are advertised.
Test Steps	• Verify that the appropriate key exchange algorithms were advertised from packet capture collected in FCS_SSH_EXT.1.4, Test 1.
Expected Test Results	The TOE should advertise appropriate mechanisms selected in the ST.
Pass/Fail with Explanation	Pass. The TOE advertised appropriate key exchange algorithms as selected in the ST. This meets the testing requirements.

7.2.13 FCS_SSH_EXT.1.6 Test #2

Item	Data
Test Assurance Activity	The evaluator shall configure an SSH peer to allow only a key exchange method that is not included in the ST selection. The evaluator shall attempt to establish an SSH connection and observe that the connection is rejected.
Test Steps	• Configure the SSH server to offer key exchange algorithm not included in the ST. • Attempt connection to TOE from the SSH server • Verify the connection attempt fails
Expected Test Results	The TOE should not connect to SSH peer when a key-exchange method not included in ST is used.
Pass/Fail with Explanation	Pass. The SSH connection is rejected when there is a key exchange method used other than the one specified in the ST. This meets the testing requirement.

7.2.14 FCS_SSH_EXT.1.7 Test #1

None.

7.2.15 FCS_SSH_EXT.1.8 Test #1

Item	Data
Test Assurance Activity	The test harness needs to be configured so that its connection rekey or termination limits are greater than the limits supported by the TOE -- it is expected that the test harness should not be initiating the connection rekey or termination. Test 1: Establish an SSH connection. Wait until the identified connection rekey limit is met. Observed that a connection rekey or termination is initiated. This may require traffic to periodically be sent, or connection keep alive to be set, to ensure that the connection is not closed due to an idle timeout.
Test Steps	Time based Rekey limit of 1 hour. • Initiate connection from TOE to the SSH server to start a SSH session. • Note the connection start time from TOE logs. • Execute “date” command for 2 hours to keep connection alive. • Verify that the rekeying is initiated from the TOE after 1 hour of start time from SSH server logs.
Expected Test Results	TOE initiates a rekey when rekey limits are reached.
Pass/Fail with Explanation	Pass. The TOE initiates a rekey when time based rekeying limit of 1 hour is reached.

7.2.16 FCS_SSH_EXT.1.8 Test #2

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

	Establish an SSH connection. Transmit data from the TOE until the identified connection rekey or termination limit is met. Observe that a connection rekey or termination is initiated.
Test Steps	Data based rekey limit of 1GB of transmitted data. • Configure the SSH server with Rekey limit of 2GB data. • Initiate connection from TOE to the SSH server to start a SSH session. • Transfer 1GB data from TOE to SSH Server. • Verify that the rekeying occurs after transfer of 1GB data from SSH server logs.
Expected Test Results	TOE initiates a rekey when rekey limits are reached.
Pass/Fail with Explanation	Pass. The TOE initiates a rekey when data based rekeying limit of 1 GB of transmitted data is reached.

7.2.17 FCS_SSH_EXT.1.8 Test #3

Item	Data
Test Assurance Activity	The test harness needs to be configured so that its connection rekey or termination limits are greater than the limits supported by the TOE -- it is expected that the test harness should not be initiating the connection rekey or termination. Establish an SSH connection. Send data to the TOE until the identified connection rekey limit or termination is met. Observe that a connection rekey or termination is initiated.
Test Steps	Data based rekey limit of 1GB of received data. • Configure the SSH server with Rekey limit of 2GB data. • Initiate connection from TOE to the SSH server to start a SSH session. ○ Click on Connect to start connection. • Transfer 1GB data from SSH Server to TOE. ○ Downloading 1 GB file from the SSH Server. ○ Click on “Download from path” to start transfer. ○ Note time after transfer is complete. • Verify that the rekeying occurs after transfer of 1GB data from SSH server logs.
Expected Test Results	TOE initiates a rekey when rekey limits are reached.
Pass/Fail with Explanation	Pass. The TOE initiates a rekey when data based rekeying limit of 1 GB of received data is reached.

7.2.18 FCS_SSHC_EXT.1 Test #1

Item	Data
Test Assurance Activity	**[conditional] If using a local database by associating each host name with its corresponding public key**, the evaluator shall configure the TOE with only a single host name and corresponding public key in the local database. The evaluator shall verify that the TOE can successfully connect to the host identified by the host name.
Test Steps	• Configure the TOE with a single hostname corresponding fingerprint of the public key. • Initiate a connection to SSH Server to verify connection. ○ Click on connect.

	○ Verify successful connection from TOE logs. ○ Verify successful connection from packet capture.
Expected Test Results	TOE successfully connects to SSH Server , when TOE is configured with only a single host name and corresponding public key in the local database.
Pass/Fail with Explanation	Pass. SSH connection is successful when TOE is configured with a single host name and corresponding public key fingerprint in the local database. This meets the testing requirement.

7.2.19 FCS_SSHC_EXT.1 Test #2

Item	Data
Test Assurance Activity	**[conditional] If using a local database by associating each host name with its corresponding public key,** the evaluator shall configure the TOE with only a single host name and non-corresponding public key in the local database. The evaluator shall verify that the TOE fails to connect to a host not identified by the host name.
Test Steps	• Configure the TOE with a single hostname and non-corresponding fingerprint of the public key. • Initiate a connection to SSH Server to verify connection. ○ Click on connect. ○ Verify failed connection from TOE logs. ○ Verify failed connection from packet capture.
Expected Test Results	TOE fails to connect to SSH Server , when TOE is configured with only a single host name and non-corresponding public key in the local database.
Pass/Fail with Explanation	Pass. TOE fails to connect to SSH Server , when TOE is configured with only a single host name and non-corresponding public key fingerprint in the local database. This meets the testing requirement.

7.2.20 FCS_SSHC_EXT.1 Test #3

Item	Data
Test Assurance Activity	**[conditional] If using a local database by associating each host name with its corresponding public key**, the evaluator shall try to connect to a host not configured in the local database. The evaluator shall verify that the TOE either fails to connect to a host identified by the host name or there is a prompt provided to store the public key in the local database.
Test Steps	• Configure the TOE with a single hostname without configuring any public key fingerprint in the local database. • Initiate a connection to SSH Server to verify connection. ○ Click on connect. ○ Verify failed connection from TOE logs. ○ Verify failed connection from packet capture.
Expected Test Results	TOE fails to connect to SSH Server , when TOE is not configured with any public key in the local database.

7.2.21 FCS_SSHC_EXT.1 Test #4

Item	Data

Test Assurance Activity	**[conditional] If using a list of trusted certification authorities**, the evaluator shall configure the TOE with only a single trusted certification authority corresponding to the host. The evaluator shall verify that the TOE can successfully connect to the host identified by the host name.
Pass/Fail with Explanation	NA. not selected in the ST. TOE is not using *“a list of trusted certification authorities”.*

7.2.22 FCS_SSHC_EXT.1 Test #5

Item	**Data**
Test Assurance Activity	**[conditional] If using a list of trusted certification authorities**, the evaluator shall configure the TOE with only a single trusted certification authority that does not correspond to the host. The evaluator shall verify that the TOE fails to the host identified by the host name.
Pass/Fail with Explanation	NA. not selected in the ST. TOE is not using *“a list of trusted certification authorities”.*

8 Conclusion

The testing shows that all test cases required for conformance have passed testing.

A. Appendix: CAVP Mapping

SFR	Algorithm in ST	Implementation name	CAVP Alg.	CAVP Cert #
FCS_CKM.1 /AK	[RSA schemes] using cryptographic key sizes of [2048-bit or greater] that meet the following: [FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.3].	Microsoft Windows 10 Anniversary Update, Windows Server 2016, Windows Storage Server 2016; Microsoft Surface Book, Surface Pro 4, Surface Pro 3 and Surface 3 w/ Windows 10 Anniversary Update; Microsoft Lumia 950 and Lumia 650 w/ Windows 10 Mobile Anniversary Update RSA Key Generation Implementation	RSA KeyGen (FIPS186-4)	RSA 2195
	[ECC schemes] using ["NIST curves" P-384 and [P-256, P-521]] that meet the following: [FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.4].	Microsoft Windows 10 Anniversary Update, Windows Server 2016, Windows Storage Server 2016; Microsoft Surface Book, Surface Pro 4, Surface Pro 3 and Surface 3 w/ Windows 10 Anniversary Update; Microsoft Lumia 950 and Lumia 650 w/ Windows 10 Mobile Anniversary Update MsBignum Cryptographic Implementations	ECDSA KeyGen (FIPS186-4) ECDSA KeyVer (FIPS186-4)	ECDSA 911
	[FFC Schemes] using ["safe-prime" groups] that meet the following: [NIST Special Publication 800-56A Revision 3, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography" and [RFC 3526]].	N/A	No NIST CAVP, CCTL has performed all assurance/evaluation activities and documented in the ETR and AAR accordingly.	N/A. Testing for FFC Schemes using safe-prime groups is done as part of testing in CKM.2.1.
FCS_CKM.2	[RSA-based key establishment schemes] that meet the following: RSAES-PKCS1-v1_5 as specified in Section 7.2 of RFC 8017, "Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography	N/A	No NIST CAVP, CCTL has performed all assurance/evaluation activities and documented in the ETR and AAR accordingly.	N/A. This testing was performed in conjunction with FTP_DIT_EXT.1 to demonstrate

SFR	Algorithm in ST	Implementation name	CAVP Alg.	CAVP Cert #
	Specifications Version 2.1” ,			correct operation.
	[Elliptic curve-based key establishment schemes] that meets the following: [NIST Special Publication 800-56A, “Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography”].	Microsoft Windows 10 Anniversary Update, Windows Server 2016, Windows Storage Server 2016; Microsoft Surface Book, Surface Pro 4, Surface Pro 3 and Surface 3 w/ Windows 10 Anniversary Update; Microsoft Lumia 950 and Lumia 650 w/ Windows 10 Mobile Anniversary Update Cryptography Next Generation (CNG) Implementations	KAS-ECC	KAS 92
	[FFC Schemes using “safe-prime” groups] that meet the following: ‘NIST Special Publication 800-56A Revision 3, “Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography” and [RFC 3526].	N/A	No NIST CAVP, CCTL has performed all assurance/evaluation activities and documented in the ETR and AAR accordingly.	N/A. This test has been successfully tested in FTP_DIT_EXT.1 that uses safe-prime groups.
FCS_COP.1 /SKC	AES used in [CBC, CTR] (as defined in NIST SP 800-38A) mode and cryptographic key sizes [128 bits, 256 bits]	Microsoft Windows 10 Anniversary Update, Windows Server 2016, Windows Storage Server 2016; Microsoft Surface Book, Surface Pro 4, Surface Pro 3 and Surface 3 w/ Windows 10 Anniversary Update; Microsoft Lumia 950 and Lumia 650 w/ Windows 10 Mobile Anniversary Update SymCrypt Cryptographic Implementations	AES-CBC AES-CTR	AES 4064
FCS_COP.1 / Sig	RSA schemes using cryptographic key sizes of 2048-bit or greater that meet the following: FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Section 4.	Microsoft Windows 10 Anniversary Update, Windows Server 2016, Windows Storage Server 2016; Microsoft Surface Book, Surface Pro 4, Surface Pro 3 and Surface 3 w/ Windows 10 Anniversary Update; Microsoft Lumia 950 and Lumia 650 w/ Windows 10 Mobile Anniversary Update MsBignum Cryptographic Implementations	RSA SigGen (FIPS186-4) RSA SigVer (FIPS186-4)	RSA 2193

SFR	Algorithm in ST	Implementation name	CAVP Alg.	CAVP Cert #
		Microsoft Windows 10 Anniversary Update, Windows Server 2016, Windows Storage Server 2016; Microsoft Surface Book, Surface Pro 4, Surface Pro 3 and Surface 3 w/ Windows 10 Anniversary Update; Microsoft Lumia 950 and Lumia 650 w/ Windows 10 Mobile Anniversary Update Cryptography Next Generation (CNG) Implementations	RSA SigGen (FIPS186-4) RSA SigVer (FIPS186-4)	RSA 2192
	For ECDSA schemes: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Section 6 and Appendix D, Implementing "NIST curves" [P-256, P-384, and P521]	Microsoft Windows 10 Anniversary Update, Windows Server 2016, Windows Storage Server 2016; Microsoft Surface Book, Surface Pro 4, Surface Pro 3 and Surface 3 w/ Windows 10 Anniversary Update; Microsoft Lumia 950 and Lumia 650 w/ Windows 10 Mobile Anniversary Update MsBignum Cryptographic Implementations	ECDSA SigGen (FIPS186-4) ECDSA SigVer (FIPS186-4)	ECDSA 911
FCS_COP.1 / Hash	[SHA-1, SHA-256, SHA-384, SHA-512] and message digest sizes [160,256, 384, 512] bits	Microsoft Windows 10 Anniversary Update, Windows Server 2016, Windows Storage Server 2016; Microsoft Surface Book, Surface Pro 4, Surface Pro 3 and Surface 3 w/ Windows 10 Anniversary Update; Microsoft Lumia 950 and Lumia 650 w/ Windows 10 Mobile Anniversary Update SymCrypt Cryptographic Implementations	SHA-1 SHA2-256 SHA2-384 SHA2-512	SHS 3347
FCS_COP.1 / KeyedHash	[HMAC-SHA- 256, HMAC-SHA-384, HMAC-SHA-512] and cryptographic key sizes [key size (in bits) used in HMAC] and message digest sizes [256, 384, 512] bits	Microsoft Windows 10 Anniversary Update, Windows Server 2016, Windows Storage Server 2016; Microsoft Surface Book, Surface Pro 4, Surface Pro 3 and Surface 3 w/ Windows 10 Anniversary Update; Microsoft Lumia 950 and Lumia 650 w/ Windows 10 Mobile Anniversary Update SymCrypt Cryptographic Implementations	HMAC-SHA2-256 HMAC-SHA2-384 HMAC-SHA2-512	HMAC 2651
FCS_RBG_EXT.1	invoke platform-provided DRBG functionality	Microsoft Windows 10 Anniversary Update, Windows Server 2016, Windows Storage Server 2016; Microsoft Surface Book, Surface Pro 4, Surface Pro 3 and Surface 3 w/ Windows 10 Anniversary Update; Microsoft Lumia 950 and Lumia 650 w/ Windows 10 Mobile Anniversary Update SymCrypt Cryptographic Implementations	Counter DRBG	DRBG 1217

End of Document