Windows 10 IPsec VPN Client
Common Criteria
Assurance Activities Report

Version 1.2

Date

October 28, 2016

Prepared by:

Leidos Inc.
Common Criteria Testing Laboratory
6841 Benjamin Franklin Drive
Columbia, MD 21046

Prepared for:

National Information Assurance Partnership
Common Criteria Evaluation and Validation Scheme

The Developer of the TOE:

Microsoft Corporation
Corporate Headquarters
One Microsoft Way
Redmond, WA 98052-6399

The TOE Evaluation was Sponsored by:

Microsoft Corporation
Corporate Headquarters
One Microsoft Way
Redmond, WA 98052-6399

Evaluation Personnel:

Greg Beaver
Gary Grainger
Kevin Steiner

Common Criteria Versions

- Common Criteria for Information Technology Security Evaluation Part 1: Introduction, Version 3.1, Revision 4, dated: September 2012.
- Common Criteria for Information Technology Security Evaluation Part 2: Security Functional Components, Revision 4, dated: September 2012.
- Common Criteria for Information Technology Security Evaluation Part 3: Security Assurance Components, Revision 4, dated: September 2012.

Common Evaluation Methodology Versions

- Common Methodology for Information Technology Security Evaluation, Evaluation Methodology, Version 3.1, Revision 4, dated: September 2012.

Protection Profiles

- Protection Profile for IPsec Virtual Private Network (VPN) Clients, 21 October 2013, Version 1.4

Contents

1 Introduction

This document presents assurance activity evaluation results of the Windows 10 IPsec VPN Client evaluation. There are three types of assurance activities and the following is provided for each:

1. TOE Summary Specification (TSS)—an indication that the required information is in the TSS section of the Security Target
2. Guidance—a specific reference to the location in the guidance is provided for the required information
3. Test—a summary of the test procedure and result is provided for each required test activity.

This Assurance Activities Report contains sections for each functional class and family and sub-sections addressing each of the SFRs specified in the Security Target.

Note that [VPN Client] specifies SFRs to be satisfied by the VPN Client (the TOE), and SFRs that can be satisfied by some combination of the TOE, the underlying TOE platform, and the VPN Gateway (indicated by appropriate selections in the SFRs).

1.1 Evidence

[ST]	*Microsoft Windows 10 IPsec VPN Client Security Target*, v0.07, October 28, 2016
[Guide]	*Windows 10 IPsec VPN Client Operational Guidance*, v1.0, October 12, 2016
[VPN Client]	*Protection Profile for IPsec Virtual Private Network (VPN) Clients*, 21 October 2013, Version 1.4
[Policy #5]	*NIAP Policy Letter #5*, 4 November 2014 (https://www.niap-ccevs.org/Documents_and_Guidance/ccevs/policy-ltr-5-update1.pdf)
[Policy #5 FAQ]	*Frequently Asked Questions for NIAP Policy #5*, 12 June 2015 (https://www.niap-ccevs.org/Documents_and_Guidance/policy-ltr-5-add1.pdf)

1.2 Technical Decisions

1.2.1 TD0079: RBG Cryptographic Transitions per NIST SP 800-131A Revision 1

NIAP Technical Decision 0079 removes option" FIPS Pub 140-2 Annex C: X9.31 Appendix 2.4 using AES" from selection in FCS_RBG_EXT.1. [ST] follows the Technical Decision. See section 3.2.23.3 in section 3.2.23 Extended: Cryptographic operation (Random Bit Generation) (FCS_RBG_EXT.1).

1.2.2 TD0053: Removal of FCS_IPSEC_EXT.1.12 Test 5 from VPN IPSEC Client v1.4

NIAP Technical Decision 0053 removes Test 5 from assurance activities for FCS_IPSEC_EXT.1.12. The decision has no effect on [ST]. Leidos followed the TD0053 in this Assurance Activity Report and in the Test Report. See:

- Section 3.2.20.3 in section 3.2.20 Extended: Internet Protocol Security (IPsec) Communications (FCS_IPSEC_EXT.1.12)
- Section 3.4.4.3 in section 3.4.4 Extended: X.509 Certificate Use and Management (FIA_X509_EXT.2.2).

1.2.3 TD0042: Removal of Low-level Crypto Failure Audit from PPs

NIAP Technical Decision 0042 modifies auditable events and audit record contents for cryptographic requirements. [ST] follows the Technical Decision. See section 0 in section 3.1.1 Security audit data generation (FAU_GEN).

1.2.4 TD0037: IPsec Requirement_DN Verification

NIAP Technical Decision 0037 replaces FCS_IPSEC_EXT.1.13, adds to FMT_SMF.1.1, and modifies tests under FCS_IPSEC_EXT.1.12. [ST] follows the Technical Decision. See:

- Section 3.2.21 Extended: Internet Protocol Security (IPsec) Communications (FCS_IPSEC_EXT.1.13)
- Section 3.5.2 Specification of Management Functions (FMT_SMF.1)
- Section 3.2.20.3 in section 3.2.20 Extended: Internet Protocol Security (IPsec) Communications (FCS_IPSEC_EXT.1.12).

1.2.5 TD0014: Satisfying FCS_IPSEC_EXT.1.13 in VPN GW EP

NIAP Technical Decision 0014 references FCS_IPSEC_EXT.1.13 in [VPN Client] but makes no changes in the protection profile. Hence, TD0014 has no effect on [ST].

2 Evaluated Configuration and TOE Platform Equivalence

[ST] section 1.1 Security Target, TOE, and Common Criteria (CC) Identification claims any general-purpose hardware on which Windows 10 runs can serve as a TOE platform. The requirements in section 5.1 TOE Security Functional Requirements support this claim in that the security target levies all the requirements on the TOE.

[VPN Client] does not explicitly address TOE platform hardware. In particular, the profile does not cover the case where the TOE platform is only hardware. As purely a software solution, the TOE depends on processor, memory, storage, and network connectivity hardware to function and meet the security functional requirements. NIAP has not approved a hardware-only protection profile that could be used for evaluating the TOE platform in this case. [ST] does not levy security functional requirements on hardware but [VPN Client] assurance activities do require CAVP testing. Thus, the set of TOE platforms in the evaluated configuration is not immediately obvious.

The evaluation team used [VPN Client], [ST], [Policy #5], and [Policy #5 FAQ] together to determine which devices would be TOE platforms in the evaluated configuration. A device would be within the evaluated configuration provided:

1) The TOE runs unmodified on the device,
2) The device provides network connectivity hardware,
3) CAVP certificates claimed in the security target apply to the device in accordance with NIAP Policy Letter #5 and its Addendum #1

[VPN Client] section 1.1.3 Operational Environment characterizes a TOE Platform separate from the TOE. [ST] Table 11 Cryptographic Standards and Evaluation Methods lists CAVP and CVL certificates for the TOE. Each CAVP certificate covers hardware identified in the certificate's Operational Environment along with equivalent hardware. NIAP describes hardware equivalence in [Policy #5] and [Policy #5 FAQ].

For example, if the TOE runs unmodified on a device included in an evaluation with the same CAVP requirements (for example, Operating System, Mobile Device, or Network Device), then the device would be within the scope of the evaluated configuration.

Regarding network connectivity, a TOE provides IPsec using the hardware. IPsec is an Internet layer protocol. Wi-Fi (802.11) and Ethernet (802.3) are Link layer protocols. Hence, a device in the evaluated configuration must provide either wireless or wired network connectivity. The TOE platforms used for testing provided only wireless connectivity.

3 Security Functional Requirements for the VPN Client (TOE)

This section describes the assurance activities associated with the SFRs defined in the ST and the results of those activities as performed by the evaluation team. The assurance activities are derived from [VPN Client].

As indicated above, security functional requirements in the main body of the PP are divided into those that must be satisfied by the VPN client (the TOE), and those that must be satisfied by either the TOE or the platform on which it runs. This section contains the requirements that must be met by the TOE.

3.1 Security Audit (FAU)

3.1.1 Security audit data generation (FAU_GEN)

3.1.1.1 TSS Assurance Activities

The PP does not identify any TSS assurance activities for this SFR.

3.1.1.2 Guidance Assurance Activities

NIAP Technical Decision 0042 modifies auditable events and audit record contents for cryptographic requirements.

The evaluator shall check the operational guidance and ensure that it lists all of the auditable events and provides a format for audit records. Each audit record format type must be covered, along with a brief description of each field. The evaluator shall check to make sure that every audit event type mandated by the PP is described and that the description of the fields contains the information required in FAU_GEN.1.2, and the additional information specified in Table 9.

[Guide] Section 2 *Managing Audits* describes the auditable events. Table 3 covers additional record content. Table 4 describes audit records, including event-specific fields.

The evaluator shall in particular ensure that the operational guidance is clear in relation to the contents for failed cryptographic events. In Table 9, information detailing the cryptographic mode of operation and a name or identifier for the object being encrypted is required. The evaluator shall ensure that name or identifier is sufficient to allow an administrator reviewing the audit log to determine the context of the cryptographic operation (for example, performed during a key negotiation exchange, performed when encrypting data for transit) as well as the non-TOE endpoint of the connection for cryptographic failures relating to communications with other IT systems.

[Guide] Section 2.1 *Audit Events* identifies the audit records for the failed cryptographic events. The audit record fields detail the cryptographic mode of operation. The audit record field description is sufficient to determine the context of the cryptographic operation as well as the non-TOE endpoint of the connection for cryptographic failures relating to communications with other IT systems.

The evaluator shall also make a determination of the administrative actions that are relevant in the context of this PP. The TOE may contain functionality that is not evaluated in the context of this PP because the functionality is not specified in an SFR. This functionality may have administrative aspects that are described in the operational guidance. Since such administrative actions will not be performed in an evaluated configuration of the TOE, the evaluator shall examine the operational guidance and make a determination of which administrative commands, including subcommands, scripts, and configuration files, are related to the configuration (including enabling or disabling) of the mechanisms implemented in the TOE that are necessary to enforce the requirements specified in the PP, which thus form the set of "all administrative actions". The evaluator may perform this activity as part of the activities associated with ensuring the AGD_OPE guidance satisfies the requirements.

[Guide] Section 2.1 *Audit Events* identifies the administrative actions that are relevant in the context of the [VPN Client] PP. Associated with each administrative action is the corresponding audit record. The identified administrative actions in the [Guide] correspond to the management activities identified in the [ST] FMT_SMF.1.

3.1.1.3 Test Activities

The evaluator shall test the TOE's ability to correctly generate audit records by having the TOE generate audit records in accordance with the assurance activities associated with the functional requirements in this PP. Additionally, the evaluator shall test that each administrative action applicable in the context of this PP is auditable. When verifying the test results, the evaluator shall ensure the audit records generated during testing match the format specified in the administrative guide, and that the fields in each audit record have the proper entries.

Note that the testing here can be accomplished in conjunction with the testing of the security mechanisms directly. For example, testing performed to ensure that the administrative guidance provided is correct verifies that AGD_OPE.1 is satisfied and should address the invocation of the administrative actions that are needed to verify the audit records are generated as expected.

The evaluator executed each test in association with a Security Functional Requirement and ensured that an audit record was generated that was in sync with the expected records listed in the AGD and discussed in the TSS of the ST. For each administrative action defined in the AGD, the evaluator ensured an audit record was indeed generated.

3.1.2 Selective Audit (FAU_SEL)

3.1.2.1 TSS Assurance Activities

The PP does not identify any TSS assurance activities for this SFR.

3.1.2.2 Guidance Assurance Activities

The evaluator shall review the administrative guidance to ensure that the guidance itemizes all event types, as well as describes all attributes that are to be selectable in accordance with the requirement, to include those attributes listed in the assignment.

[Guide] Section 2.2.1 *Local Administrator Guidance* identifies the audit selectable events types and the attributes listed in the SFR assignment. The [Guide] provides the following TechNet topic which describes how to select audit policies by category, user and audit success or failure in the Security log:

- Auditpol set: https://technet.microsoft.com/en-us/library/cc755264.aspx

The administrative guidance shall also contain instructions on how to set the pre-selection, or how the VPN Gateway will configure the client, as well as explain the syntax (if present) for multi-value pre-selection..

[Guide] Section 2.2.1 *Local Administrator Guidance* identifies the instructions to enable all audits in the followings subcategories of the Windows Logs:

- Audit policy changes
- IPsec operations
- Configuring IKEv1 and IKEv2 connection properties

The administrative guidance shall also identify those audit records that are always recorded, regardless of the selection criteria currently being enforced.

[Guide] Section 2.2.1 *Local Administrator Guidance* identifies the audit records and locations that are always enabled.

- Windows Logs -> System
- Windows Logs -> Setup
- Windows Logs -> Security (for startup and shutdown of the audit functions and of the OS and kernel, and clearing the audit log)

3.1.2.3 Test Activities

Test 1: For each attribute listed in the requirement, the evaluator shall devise a test to show that selecting the attribute causes only audit events with that attribute (or those that are always recorded, as identified in the administrative guidance) to be recorded..

Test 2 [conditional]: If the TSF supports specification of more complex audit pre-selection criteria (e.g., multiple attributes, logical expressions using attributes) then the evaluator shall devise tests showing that this capability is correctly implemented. The evaluator shall also, in the test plan, provide a short narrative justifying the set of tests as representative and sufficient to exercise the capability.

The evaluator configured different policies and applied them to the TOE to ensure that a user had the ability to include or exclude specific information. This was done via the use of the 'auditpol' command line utility.

3.2 Cryptographic Support (FCS)

3.2.1 Cryptographic Key Generation (for asymmetric keys) (FCS_CKM.1(1))

FCS_CKM.1(1) is labeled FCS_CKM.1(ASYM) in [ST].

3.2.1.1 TSS Assurance Activity

Requirement met by the TOE

SP800-56B Key Establishment Schemes

At this time, detailed test procedures for RSA-based key establishment schemes are not available. In order to show that the TSF complies with 80056A and/or 80056B, depending on the selections made, the evaluator shall ensure that the TSS contains the following information:

- The TSS shall list all sections of the appropriate 800-56 standard(s) to which the TOE complies.
- For each applicable section listed in the TSS, for all statements that are not "shall" (that is, "shall not", "should", and "should not"), if the TOE implements such options it shall be described in the TSS. If the included functionality is indicated as "shall not" or "should not" in the standard, the TSS shall provide a rationale for why this will not adversely affect the security policy implemented by the TOE.

For each applicable section of 800-56A and 800-56B (as selected), any omission of functionality related to "shall" or "should" statements shall be described.

Only SP 800-56A applies to the TOE, since [ST] does not select "NIST Special Publication 800-56B … for RSA-based key establishment schemes" in FCS_CKM.1(ASYM). [ST].section 6.4.1 VPN Tunnels includes a footnote that Windows 10 follows section 5 of NISP SP 800-56A. Search for "Windows follows section 5 of NIST SP 800-56A"

3.2.1.2 Guidance Assurance Activities

The PP does not identify any TOE guidance assurance activities.

3.2.1.3 Test Activity

Requirement met by the TOE

This assurance activity will verify the key generation and key establishments schemes used on the TOE.

Key Generation:

The evaluator shall verify the implementation of the key generation routines of the supported schemes using the applicable tests below.

Key Generation for RSA-Based Key Establishment 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.

Windows 10 uses algorithm implementations validated under the Cryptographic Algorithm Validation Program (http://csrc.nist.gov/groups/STM/cavp/index.html). [ST] Table 11 Cryptographic Standards and Evaluation Methods identifies applicable CAVP RSA certificates: 1887, 1888, 1889, and 1871 (http://csrc.nist.gov/groups/STM/cavp/documents/dss/rsanewval.html). The relevant detail is reproduced and highlighted below.

1871	**Microsoft Windows 10 November 2015 Update; Microsoft Surface Book, Surface Pro 4, Surface Pro 3, Surface 3, Surface Pro 2, and Surface Pro w/ Windows 10 November 2015 Update; Windows 10 Mobile for Microsoft Lumia 950 and Microsoft Lumia 635; Windows 10 for Microsoft Surface Hub 84" and Surface Hub 55" RSA32 Algorithm Implementations** **FIPS186-4:** **ALG[RSASSA-PKCS1_V1_5]** SIG(Ver) (1024 SHA(1 , 256 , 384 , 512)) (2048 SHA(1 , 256 , 384 , 512)) (3072 SHA(1 , 256 , 384 , 512)) SHA Val#3048
1887	**Microsoft Windows 10 November 2015 Update; Microsoft Surface Book, Surface Pro 4, Surface Pro 3, Surface 3, Surface Pro 2, and Surface Pro w/ Windows 10 November 2015 Update; Windows 10 Mobile for Microsoft Lumia 950 and Microsoft Lumia 635; Windows 10 for Microsoft Surface Hub 84" and Surface Hub 55" Cryptography Next Generation (CNG) Implementations** **FIPS186-4:** **[RSASSA-PSS]:** Sig(Gen): (2048 SHA(224 , 256 SaltLen(32) , 384 SaltLen(48) , 512 SaltLen(64))) (3072 SHA(224 , 256 SaltLen(32) , 384 SaltLen(48) , 512 SaltLen(64))) Sig(Ver): (1024 SHA(1 SaltLen(20) , 256 SaltLen(32) , 384 SaltLen(48) , 512 SaltLen(62))) (2048 SHA(1 SaltLen(20) , 256 SaltLen(32) , 384 SaltLen(48) , 512 SaltLen(64))) (3072 SHA(1 SaltLen(20) , 256 SaltLen(32) , 384 SaltLen(48) , 512 SaltLen(64))) SHA Val#3047
1888	**Microsoft Windows 10 November 2015 Update; Microsoft Surface Book, Surface Pro 4, Surface Pro 3, Surface 3, Surface Pro 2, and Surface Pro w/ Windows 10 November 2015 Update; Windows 10 Mobile for Microsoft Lumia 950 and Microsoft Lumia 635; Windows 10 for Microsoft Surface Hub 84" and Surface Hub 55" MsBignum Cryptographic Implementations** **FIPS186-4:** **ALG[RSASSA-PKCS1_V1_5]** SIG(gen) (2048 SHA(256 , 384 , 512)) (3072 SHA(256 , 384 , 512)) SIG(Ver) (1024 SHA(1 , 256 , 384 , 512)) (2048 SHA(1 , 256 , 384 , 512)) (3072 SHA(1 , 256 , 384 , 512)) SHA Val#3047
1889	**Microsoft Windows 10 November 2015 Update; Microsoft Surface Book, Surface Pro 4, Surface Pro 3, Surface 3, Surface Pro 2, and Surface Pro w/ Windows 10 November 2015 Update; Windows 10 Mobile for Microsoft Lumia 950 and Microsoft Lumia 635; Windows 10 for Microsoft Surface Hub 84" and Surface Hub 55" RSA Key Generation Implementation**

	FIPS186-4: **186-4KEY(gen):** FIPS186-4_Fixed_e (10001) **PGM(ProbPrimeCondition):** 2048 , 3072 **PPTT:** (C.3) SHA Val#3047 DRBG: Val# 955

Requirement met by the TOE

Key Generation for Finite-Field Cryptography (FFC) – Based 56A Schemes

FFC Domain Parameter and Key Generation Tests

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

for each FFC parameter set and key pair.

Windows 10 uses algorithm implementations validated under the CAVP (http://csrc.nist.gov/groups/STM/cavp/index.html). [ST] Table 11 Cryptographic Standards and Evaluation Methods identifies applicable CAVP DSA certificate: 1024 (http://csrc.nist.gov/groups/STM/cavp/documents/dss/dsanewval.html). The relevant detail is reproduced and highlighted below.

1024	**Microsoft Windows 10 November 2015 Update; Microsoft Surface Book, Surface Pro 4, Surface Pro 3, Surface 3, Surface Pro 2, and Surface Pro w/ Windows 10 November 2015 Update; Windows 10 Mobile for Microsoft Lumia 950 and Microsoft Lumia 635; Windows 10 for Microsoft Surface Hub 84" and Surface Hub 55" MsBignum Cryptographic Implementations** **FIPS186-4:** **PQG(gen)PARMS TESTED:** [(2048,256)SHA(256); (3072,256) SHA(256)] **PQG(ver)PARMS TESTED:** [(2048,256) SHA(256); (3072,256) SHA(256)] **Key PairGen:** [(2048,256) ; (3072,256)] **SIG(gen)PARMS TESTED:** [(2048,256) SHA(256); (3072,256) SHA(256);] **SIG(ver)PARMS TESTED:** [(2048,256) SHA(256); (3072,256) SHA(256)] SHS: Val# 3047 DRBG: Val# 955

Requirement met by the TOE

Key Generation for Elliptic Curve Cryptography (ECC)- Based 56A Schemes

ECC Key Generation Test

For each supported NIST curve, i.e., P-256, P-284 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.

ECC Public Key Verification (PKV) Test

For each supported NIST curve, i.e., P-256, P-284 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.

Windows 10 uses algorithm implementations validated under the CAVP (http://csrc.nist.gov/groups/STM/cavp/index.html). [ST] Table 11 Cryptographic Standards and Evaluation Methods identifies applicable CAVP ECDSA certificate: 760 (http://csrc.nist.gov/groups/STM/cavp/documents/dss/ecdsanewval.html). The relevant detail is reproduced and highlighted below.

760	**Microsoft Windows 10 November 2015 Update; Microsoft Surface Book, Surface Pro 4, Surface Pro 3, Surface 3, Surface Pro 2, and Surface Pro w/ Windows 10 November 2015 Update; Windows 10 Mobile for Microsoft Lumia 950 and Microsoft Lumia 635; Windows 10 for Microsoft Surface Hub 84" and Surface Hub 55" MsBignum Cryptographic Implementations** **FIPS186-4:** **PKG: CURVES**(P-256 P-384 P-521 ExtraRandomBits) **SigGen: CURVES**(P-256: (SHA-256) P-384: (SHA-384) P-521: (SHA-512) **SigVer: CURVES**(P-256: (SHA-256) P-384: (SHA-384) P-521: (SHA-512)) **SHS:** Val#3047 **DRBG:** Val# 955

Requirement met by the TOE

Key Establishment Schemes

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

SP800-56A Key Establishment Schemes

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

Function Test

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

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

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

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

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

Validity Test

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

The evaluator shall inject an error in some of the test vectors to test that the TOE recognizes invalid key agreement results caused by the following fields being incorrect: the shared secret value Z, the DKM, the other information field OI, the data to be MACed, or the generated MACTag. If the TOE contains the full or partial (only ECC) public key validation, the evaluator will also individually inject errors in both parties' static public keys, both parties' ephemeral public keys and the TOE's static private key to assure the TOE detects errors in the public key validation function and/or the partial key validation function (in ECC only). At least two of the test vectors shall remain unmodified and therefore should result in valid key agreement results (they should pass).

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

Windows 10 uses algorithm implementations validated under the CAVP (http://csrc.nist.gov/groups/STM/cavp/index.html). [ST] Table 11 Cryptographic Standards and Evaluation Methods identifies applicable CAVP KAS certificate: 72

(http://csrc.nist.gov/groups/STM/cavp/documents/keymgmt/kasnewval.html). The relevant detail is reproduced and highlighted below.

72	**Microsoft Windows 10 November 2015 Update; Microsoft Surface Book, Surface Pro 4, Surface Pro 3, Surface 3, Surface Pro 2, and Surface Pro w/ Windows 10 November 2015 Update; Windows 10 Mobile for Microsoft Lumia 950 and Microsoft Lumia 635; Windows 10 for Microsoft Surface Hub 84" and Surface Hub 55" Cryptography Next Generation (CNG) Implementations** **FFC:** (FUNCTIONS INCLUDED IN IMPLEMENTATION: DPG DPV KPG Partial Validation) **SCHEMES** [**dhEphem** (KARole(s): Initiator / Responder) (**FB:** SHA256) (**FC:** SHA256)] [**dhOneFlow** (KARole(s): Initiator / Responder) (**FB:** SHA256) (**FC:** SHA256)] [**dhStatic** (**No_KC** < KARole(s): Initiator / Responder>) (**FB:** SHA256 HMAC) (**FC:** SHA256 HMAC)] SHS Val#3047 DSA Val#1024 DRBG Val#955 **ECC:** (FUNCTIONS INCLUDED IN IMPLEMENTATION: DPG DPV KPG Partial Validation Key Regeneration) **SCHEMES** [**EphemeralUnified** (**No_KC** < KARole(s): Initiator / Responder>) (**EC:** P-256 SHA256 HMAC) (**ED:** P-384 SHA384 HMAC) (**EE:** P-521 HMAC (SHA512, HMAC_SHA512)))] [**OnePassDH** (**No_KC** < KCRole(s): Initiator Responder>) (**EB:**) (**EC:** P-256 SHA256 HMAC) (**ED:** P-521 SHA384 HMAC) (**EE:** P-521 HMAC (SHA512, HMAC_SHA512))] [**StaticUnified** (**No_KC** < KARole(s): Initiator / Responder>) (**EC:** P-256 SHA256 HMAC) (**ED:** P-384 SHA384 HMAC) (**EE:** P-521 HMAC (SHA512, HMAC_SHA512))] SHS Val#3047 ECDSA Val#760 DRBG Val#955

3.2.2 Cryptographic Key Generation (for asymmetric keys - IKE) (FCS_CKM.1(2))

FCS_CKM.1(2) is labeled FCS_CKM.1(IKE) in [ST].

3.2.2.1 TSS Assurance Activity

If the TSF implements a FIPS 186-4 signature scheme, this requirement is verified under FCS_COP.1(2).

See 3.2.6.1 in 3.2.6 Cryptographic Operation (for cryptographic signature) (FCS_COP.1(2)).

3.2.2.2 Guidance Assurance Activities

If the TSF implements a FIPS 186-4 signature scheme, this requirement is verified under FCS_COP.1(2).

See 3.2.6.2 in 3.2.6 Cryptographic Operation (for cryptographic signature) (FCS_COP.1(2)).

3.2.2.3 Test Activity

If the TSF implements a FIPS 186-4 signature scheme, this requirement is verified under FCS_COP.1(2).

See 3.2.6.3 in 3.2.6 Cryptographic Operation (for cryptographic signature) (FCS_COP.1(2)).

3.2.3 Cryptographic Key Storage (FCS_CKM_EXT.2)

3.2.3.1 TSS Assurance Activity

Regardless of whether this requirement is met by the TOE or the TOE platform, the evaluator will check the TSS to ensure that it lists each persistent secret (credential, secret key) and private key needed to meet the requirements in the ST. For each of these items, the evaluator will confirm that the TSS lists for what purpose it is used, and how it is stored. The evaluator than performs the following actions.

[ST] section 6.3 Cryptographic Support presents a list of keys and secrets in Table 12 Types of Keys and Cryptographic Materia Used by Windows.

[ST] section 6.3.2 SFR Mapping identifies DPAPI and NTFS discretionary access controls as mechanisms providing secure storage. The mechanisms provided secure key storage for both private and pre-shared keys. A footnote references both mobile device and operating system evaluations for descriptions of the mechanisms.

Section 6.5.2 Certificate Validation and Usage explains that Windows stores X.509 certificate in Windows certificate stores for individual users and the host machine.

Persistent secrets and private keys manipulated by the platform

For each platform listed in the ST, the evaluator shall examine the ST of the platform to ensure that the persistent secrets and private keys listed as being stored by the platform in the VPN client ST are identified as being protected in that platform's ST.

The TOE platform is hardware only. The TOE platform does not manipulate persistent secrets or private keys separately from Windows 10. See section 2 Evaluated Configuration and TOE Platform Equivalence regarding the absence of a TOE platform ST.

Persistent secrets and private keys manipulated by the TOE

The evaluator reviews the TSS for to determine that it makes a case that, for each item listed as being manipulated by the TOE, it is not written unencrypted to persistent memory, and that the item is stored by the platform.

Item FCS_CKM_EXT.2 in [ST] section 6.3.2 SFR Mapping identifies the mechanisms Windows uses to protect stored keys, namely DPAPI and the NTFS discretionary access controls.

3.2.3.2 Guidance Assurance Activities

The PP does not identify any TOE guidance assurance activities.

3.2.3.3 Test Activity

The PP does not identify any TOE test assurance activities.

3.2.4 Cryptographic Key Zeroization (FCS_CKM_EXT.4)

3.2.4.1 TSS Assurance Activity

The evaluator shall ensure that all plaintext secret and private cryptographic keys and CSPs (whether manipulated by the TOE or exclusively by the platform) are identified in the VPN Client ST's TSS, and that they are accounted for by the assurance activities in this section.

[ST] section 6.3 Cryptographic Support presents a list of keys and secrets in Table 12 Types of Keys and Cryptographic Materia Used by Windows.

Requirement met by the TOE

The evaluator shall check to ensure the TSS describes when each of the plaintext keys are cleared (e.g., system power off, disconnection of an IPsec connection, when no longer needed by the VPN channel per the protocol); and the type of clearing procedure that is performed (cryptographic erase, overwrite with zeros, overwrite three or more times by a different alternating pattern, overwrite with random pattern, or block erase).

[ST] section 6.3 Cryptographic Support states Windows 10 zeroizes the keys and secrets listed in Table 12 when they are no longer needed. The section states Windows 10 destroys non-persistent cryptographic keys by a single direct overwrite consisting of zeros.

Requirement met by the TOE

If different types of memory are used to store the materials to be protected, the evaluator shall check to ensure that the TSS describes the clearing procedure in terms of the memory in which the data are stored (for example, "secret

keys stored on flash are cleared by overwriting once with zeros, while secret keys stored on the internal persistent storage device are cleared by overwriting three times with a random pattern that is changed before each write").

[ST] section 6.3 Cryptographic Support states Windows 10 destroys non-persistent cryptographic keys by a single direct overwrite consisting of zeros.

In [ST] section 6.3.3 SFR Mapping Item FCS_CKM_EXT.4 states "Windows overwrites critical cryptographic parameters immediately after that data is no longer needed."

3.2.4.2 Guidance Assurance Activities

The PP does not identify any platform guidance assurance activities.

3.2.4.3 Test Activity

For each key clearing situation described in the TSS, the evaluator shall repeat the following test.

Test 1: The evaluator shall utilize appropriate combinations of specialized operational environment and development tools (debuggers, simulators, etc.) for the TOE and instrumented TOE builds to test that keys are cleared correctly, including all intermediate copies of the key that may have been created internally by the TOE during normal cryptographic processing with that key.

Cryptographic TOE implementations in software shall be loaded and exercised under a debugger to perform such tests. The evaluator shall perform the following test for each key subject to clearing, including intermediate copies of keys that are persisted encrypted by the TOE:

1. Load the instrumented TOE build in a debugger.
2. Record the value of the key in the TOE subject to clearing.
3. Cause the TOE to perform a normal cryptographic processing with the key from #1.
4. Cause the TOE to clear the key.
5. Cause the TOE to stop the execution but not exit.
6. Cause the TOE to dump the entire memory footprint of the TOE into a binary file.
7. Search the content of the binary file created in #4 for instances of the known key value from #1.

The test succeeds if no copies of the key from #1 are found in step #7 above and fails otherwise.

The evaluator shall perform this test on all keys, including those persisted in encrypted form, to ensure intermediate copies are cleared.

The evaluator attached the TOE to a debugger and used a tool developed by the evaluator team to create and clear an encryption key. The evaluator dumped memory before and after the key was cleared and verified that no traces of the key remained and the memory was zeroed.

3.2.5 Cryptographic Operation (Data Encryption/Decryption) (FCS_COP.1(1))

FCS_COP.1(1) is labeled FCS_COP.1(SYM) in [ST].

3.2.5.1 TSS Assurance Activity

Requirement met by the TOE

The PP does not identify any TOE TSS assurance activities.

3.2.5.2 Guidance Assurance Activities

The PP does not identify any TOE guidance assurance activities.

3.2.5.3 Test Activity

Requirement met by the TOE

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

AES-CBC Tests

AES-CBC Known Answer Tests

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

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

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

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

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

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

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

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

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

AES-CBC Multi-Block Message Test

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

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

the same ciphertext message with the same key and IV using a known good implementation.

AES-CBC Monte Carlo Tests

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

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

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

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

Windows 10 uses algorithm implementations validated under the CAVP (http://csrc.nist.gov/groups/STM/cavp/index.html). [ST] Table 11 Cryptographic Standards and Evaluation Methods identifies applicable CAVP AES certificates: 3629 and 3630 (http://csrc.nist.gov/groups/STM/cavp/documents/aes/aesval.html). The relevant detail is reproduced and highlighted below.

3629	**Microsoft Windows 10 November 2015 Update; Microsoft Surface Book, Surface Pro 4, Surface Pro 3, Surface 3, Surface Pro 2, and Surface Pro w/ Windows 10 November 2015 Update; Windows 10 Mobile for Microsoft Lumia 950 and Microsoft Lumia 635; Windows 10 for Microsoft Surface Hub 84" and Surface Hub 55" SymCrypt Cryptographic Implementations** **ECB** (e/d; 128 , 192 , 256); **CBC** (e/d; 128 , 192 , 256); **CFB8** (e/d; 128 , 192 , 256); **CFB128** (e/d; 128 , 192 , 256); **CTR** (int only; 128 , 192 , 256) **CCM (KS: 128 , 192 , 256**) **(Assoc. Data Len Range**: 0 - 0 , 2^16) **(Payload Length Range**: 0 - 32 (**Nonce Length(s)**: 7 8 9 10 11 12 13 **(Tag Length(s)**: 4 6 8 10 12 14 16) **CMAC** (Generation/Verification) **(KS: 128;** Block Size(s): Full / Partial ; **Msg Len(s)** Min: 0 Max: 2^16 ; **Tag Len(s)** Min: 0 Max: 16) **(KS: 192**; Block Size(s): Full / Partial ; **Msg Len(s)** Min: 0 Max: 2^16 ; **Tag Len(s)** Min: 0 Max: 16) **(KS: 256**; Block Size(s): Full / Partial ; **Msg Len(s)** Min: 0 Max: 2^16 ; **Tag Len(s)** Min: 0 Max: 16) **GCM (KS: AES_128**(e/d) Tag Length(s): 128 120 112 104 96) **(KS: AES_192**(e/d) Tag Length(s): 128 120 112 104 96) **(KS: AES_256**(e/d) Tag Length(s): 128 120 112 104 96) **IV Generated:** (Externally) ; **PT Lengths Tested:** (0 , 1024 , 8 , 1016) ; **AAD Lengths tested:** (0 , 1024 , 8 , 1016) ; **96BitIV_Supported** **GMAC_Supported** **XTS((KS: XTS_128**((e/d) (f)) **KS: XTS_256**((e/d) (f))
3630	**Microsoft Windows 10 November 2015 Update; Microsoft Surface Book, Surface Pro 4, Surface Pro 3, Surface 3, Surface Pro 2, and Surface Pro w/ Windows 10 November 2015 Update; Windows 10 Mobile for Microsoft Lumia 950 and Microsoft Lumia 635; Windows 10 for Microsoft Surface Hub**

	84" and Surface Hub 55" RSA32 Algorithm Implementations **ECB** (e/d; 128 , 192 , 256); **CBC** (e/d; 128 , 192 , 256); **CFB8** (e/d; 128 , 192 , 256);

AES-GCM Monte Carlo Test

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

128 bit and 256 bit keys

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

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

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

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

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

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

Windows 10 uses algorithm implementations validated under the CAVP (http://csrc.nist.gov/groups/STM/cavp/index.html). [ST] Table 11 Cryptographic Standards and Evaluation Methods identifies applicable CAVP AES certificates: 3629 and 3630 (http://csrc.nist.gov/groups/STM/cavp/documents/aes/aesval.html). The relevant detail is reproduced and highlighted below.

3629	**Microsoft Windows 10 November 2015 Update; Microsoft Surface Book, Surface Pro 4, Surface Pro 3, Surface 3, Surface Pro 2, and Surface Pro w/ Windows 10 November 2015 Update; Windows 10 Mobile for Microsoft Lumia 950 and Microsoft Lumia 635; Windows 10 for Microsoft Surface Hub 84" and Surface Hub 55" SymCrypt Cryptographic Implementations** **ECB** (e/d; 128 , 192 , 256); **CBC** (e/d; 128 , 192 , 256); **CFB8** (e/d; 128 , 192 , 256); **CFB128** (e/d; 128 , 192 , 256); **CTR** (int only; 128 , 192 , 256) **CCM (KS: 128 , 192 , 256**) **(Assoc. Data Len Range**: 0 - 0 , 2^16) **(Payload Length Range**: 0 - 32 (**Nonce Length(s)**: 7 8 9 10 11 12 13 **(Tag Length(s)**: 4 6 8 10 12 14 16) **CMAC** (Generation/Verification) **(KS: 128;** Block Size(s): Full / Partial ; **Msg Len(s)** Min: 0 Max: 2^16 ; **Tag Len(s)** Min: 0 Max: 16) **(KS: 192**; Block Size(s): Full / Partial ; **Msg Len(s)** Min: 0 Max: 2^16 ; **Tag Len(s)** Min: 0 Max: 16) **(KS: 256**; Block Size(s): Full / Partial ; **Msg Len(s)** Min: 0 Max: 2^16 ; **Tag Len(s)** Min: 0 Max: 16) **GCM (KS: AES_128**(e/d) Tag Length(s): 128 120 112 104 96) **(KS: AES_192**(e/d) Tag Length(s): 128 120 112 104 96) **(KS: AES_256**(e/d) Tag Length(s): 128 120 112 104 96) **IV Generated:** (Externally) ; **PT Lengths Tested:** (0 , 1024 , 8 , 1016) ; **AAD Lengths tested:** (0 , 1024 , 8 , 1016) ; **96BitIV_Supported** **GMAC_Supported**

	XTS((KS: XTS_128((e/d) (f)) **KS: XTS_256**((e/d) (f))
3630	**Microsoft Windows 10 November 2015 Update; Microsoft Surface Book, Surface Pro 4, Surface Pro 3, Surface 3, Surface Pro 2, and Surface Pro w/ Windows 10 November 2015 Update; Windows 10 Mobile for Microsoft Lumia 950 and Microsoft Lumia 635; Windows 10 for Microsoft Surface Hub 84" and Surface Hub 55" RSA32 Algorithm Implementations** **ECB** (e/d; 128 , 192 , 256); **CBC** (e/d; 128 , 192 , 256); **CFB8** (e/d; 128 , 192 , 256);

3.2.6 Cryptographic Operation (for cryptographic signature) (FCS_COP.1(2))

FCS_COP.1(2) is labeled FCS_COP.1(SIGN) in [ST].

3.2.6.1 TSS Assurance Activity

Requirement met by the TOE

The PP does not identify any TOE TSS assurance activities.

3.2.6.2 Guidance Assurance Activities

The PP does not identify any TOE guidance assurance activities.

3.2.6.3 Test Activity

Requirement met by the TOE

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

Key Generation:

Key Generation for RSA Signature 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.

Windows 10 uses algorithm implementations validated under the Cryptographic Algorithm Validation Program (http://csrc.nist.gov/groups/STM/cavp/index.html). [ST] Table 11 Cryptographic Standards and Evaluation Methods

identifies applicable CAVP RSA certificates: 1887, 1888, 1889, and 1871 (http://csrc.nist.gov/groups/STM/cavp/documents/dss/rsanewval.html). The relevant detail is reproduced and highlighted below.

1871	**Microsoft Windows 10 November 2015 Update; Microsoft Surface Book, Surface Pro 4, Surface Pro 3, Surface 3, Surface Pro 2, and Surface Pro w/ Windows 10 November 2015 Update; Windows 10 Mobile for Microsoft Lumia 950 and Microsoft Lumia 635; Windows 10 for Microsoft Surface Hub 84" and Surface Hub 55" RSA32 Algorithm Implementations** **FIPS186-4:** **ALG[RSASSA-PKCS1_V1_5]** SIG(Ver) (1024 SHA(1 , 256 , 384 , 512)) (2048 SHA(1 , 256 , 384 , 512)) (3072 SHA(1 , 256 , 384 , 512)) SHA Val#3048
1887	**Microsoft Windows 10 November 2015 Update; Microsoft Surface Book, Surface Pro 4, Surface Pro 3, Surface 3, Surface Pro 2, and Surface Pro w/ Windows 10 November 2015 Update; Windows 10 Mobile for Microsoft Lumia 950 and Microsoft Lumia 635; Windows 10 for Microsoft Surface Hub 84" and Surface Hub 55" Cryptography Next Generation (CNG) Implementations** **FIPS186-4:** **[RSASSA-PSS]:** Sig(Gen): (2048 SHA(224 , 256 SaltLen(32) , 384 SaltLen(48) , 512 SaltLen(64))) (3072 SHA(224 , 256 SaltLen(32) , 384 SaltLen(48) , 512 SaltLen(64))) Sig(Ver): (1024 SHA(1 SaltLen(20) , 256 SaltLen(32) , 384 SaltLen(48) , 512 SaltLen(62))) (2048 SHA(1 SaltLen(20) , 256 SaltLen(32) , 384 SaltLen(48) , 512 SaltLen(64))) (3072 SHA(1 SaltLen(20) , 256 SaltLen(32) , 384 SaltLen(48) , 512 SaltLen(64))) SHA Val#3047
1888	**Microsoft Windows 10 November 2015 Update; Microsoft Surface Book, Surface Pro 4, Surface Pro 3, Surface 3, Surface Pro 2, and Surface Pro w/ Windows 10 November 2015 Update; Windows 10 Mobile for Microsoft Lumia 950 and Microsoft Lumia 635; Windows 10 for Microsoft Surface Hub 84" and Surface Hub 55" MsBignum Cryptographic Implementations** **FIPS186-4:** **ALG[RSASSA-PKCS1_V1_5]** SIG(gen) (2048 SHA(256 , 384 , 512)) (3072 SHA(256 , 384 , 512)) SIG(Ver) (1024 SHA(1 , 256 , 384 , 512)) (2048 SHA(1 , 256 , 384 , 512)) (3072 SHA(1 , 256 , 384 , 512)) SHA Val#3047
1889	**Microsoft Windows 10 November 2015 Update; Microsoft Surface Book, Surface Pro 4, Surface Pro 3, Surface 3, Surface Pro 2, and Surface Pro w/ Windows 10 November 2015 Update; Windows 10 Mobile for Microsoft Lumia 950 and Microsoft Lumia 635; Windows 10 for Microsoft Surface Hub 84" and Surface Hub 55" RSA Key Generation Implementation** **FIPS186-4:** **186-4KEY(gen):** FIPS186-4_Fixed_e (10001) **PGM(ProbPrimeCondition):** 2048 , 3072 **PPTT:** (C.3) SHA Val#3047 DRBG: Val# 955

ECDSA Key Generation Tests

FIPS 186-4 ECDSA Key Generation Test

For each supported NIST curve, i.e., P-256, P-284 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-284 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.

Windows 10 uses algorithm implementations validated under the CAVP (http://csrc.nist.gov/groups/STM/cavp/index.html). [ST] Table 11 Cryptographic Standards and Evaluation Methods identifies applicable CAVP ECDSA certificate: 760 (http://csrc.nist.gov/groups/STM/cavp/documents/dss/ecdsanewval.html). The relevant detail is reproduced and highlighted below.

760	**Microsoft Windows 10 November 2015 Update; Microsoft Surface Book, Surface Pro 4, Surface Pro 3, Surface 3, Surface Pro 2, and Surface Pro w/ Windows 10 November 2015 Update; Windows 10 Mobile for Microsoft Lumia 950 and Microsoft Lumia 635; Windows 10 for Microsoft Surface Hub 84" and Surface Hub 55" MsBignum Cryptographic Implementations** **FIPS186-4:** **PKG: CURVES**(P-256 P-384 P-521 ExtraRandomBits) **SigGen: CURVES**(P-256: (SHA-256) P-384: (SHA-384) P-521: (SHA-512) **SigVer: CURVES**(P-256: (SHA-256) P-384: (SHA-384) P-521: (SHA-512)) **SHS:** Val#3047 **DRBG:** Val# 955

ECDSA Algorithm Tests

ECDSA FIPS 186-4 Signature Generation Test

For each supported NIST curve (i.e., P-256, P-284 and P-521) and SHA function pair, the evaluator shall generate 10 1024-bit long messages and obtain for each message a public key and the resulting signature values R and S. To determine correctness, the evaluator shall use the signature verification function of a known good implementation.

ECDSA FIPS 186-4 Signature Verification Test

For each supported NIST curve (i.e., P-256, P-284 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.

Windows 10 uses algorithm implementations validated under the CAVP (http://csrc.nist.gov/groups/STM/cavp/index.html). [ST] Table 11 Cryptographic Standards and Evaluation Methods identifies applicable CAVP ECDSA certificate: 760 (http://csrc.nist.gov/groups/STM/cavp/documents/dss/ecdsanewval.html). The relevant detail is reproduced and highlighted below.

760	**Microsoft Windows 10 November 2015 Update; Microsoft Surface Book, Surface Pro 4, Surface Pro 3, Surface 3, Surface Pro 2, and Surface Pro w/ Windows 10 November 2015 Update; Windows 10 Mobile for Microsoft Lumia 950 and Microsoft Lumia 635; Windows 10 for Microsoft Surface Hub 84" and Surface Hub 55" MsBignum Cryptographic Implementations** **FIPS186-4:** **PKG: CURVES**(P-256 P-384 P-521 ExtraRandomBits) **SigGen: CURVES**(P-256: (SHA-256) P-384: (SHA-384) P-521: (SHA-512) **SigVer: CURVES**(P-256: (SHA-256) P-384: (SHA-384) P-521: (SHA-512)) **SHS:** Val#3047 **DRBG:** Val# 955

RSA Signature Algorithm Tests

Signature Generation Test

The evaluator shall verify the implementation of RSA Signature Generation by the TOE using the Signature Generation Test. To conduct this test the evaluator 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.

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.

Windows 10 uses algorithm implementations validated under the Cryptographic Algorithm Validation Program (http://csrc.nist.gov/groups/STM/cavp/index.html). [ST] Table 11 Cryptographic Standards and Evaluation Methods identifies applicable CAVP RSA certificates: 1871, 1887, 1888, and 1889 (http://csrc.nist.gov/groups/STM/cavp/documents/dss/rsanewval.html). The relevant detail is reproduced and highlighted below.

1871	**Microsoft Windows 10 November 2015 Update; Microsoft Surface Book, Surface Pro 4, Surface Pro 3, Surface 3, Surface Pro 2, and Surface Pro w/ Windows 10 November 2015 Update; Windows 10 Mobile for Microsoft Lumia 950 and Microsoft Lumia 635; Windows 10 for Microsoft Surface Hub 84" and Surface Hub 55" RSA32 Algorithm Implementations** **FIPS186-4:** **ALG[RSASSA-PKCS1_V1_5]** SIG(Ver) (1024 SHA(1 , 256 , 384 , 512)) (2048 SHA(1 , 256 , 384 , 512)) (3072 SHA(1 , 256 , 384 , 512)) SHA Val#3048
1887	**Microsoft Windows 10 November 2015 Update; Microsoft Surface Book, Surface Pro 4, Surface Pro 3, Surface 3, Surface Pro 2, and Surface Pro w/ Windows 10 November 2015 Update; Windows 10 Mobile for Microsoft Lumia 950 and Microsoft Lumia 635; Windows 10 for Microsoft Surface Hub 84" and Surface Hub 55" Cryptography Next Generation (CNG) Implementations** **FIPS186-4:** **[RSASSA-PSS]:** Sig(Gen): (2048 SHA(224 , 256 SaltLen(32) , 384 SaltLen(48) , 512 SaltLen(64))) (3072 SHA(224 , 256 SaltLen(32) , 384 SaltLen(48) , 512 SaltLen(64))) Sig(Ver): (1024 SHA(1 SaltLen(20) , 256 SaltLen(32) , 384 SaltLen(48) , 512 SaltLen(62))) (2048 SHA(1 SaltLen(20) , 256 SaltLen(32) , 384 SaltLen(48) , 512 SaltLen(64))) (3072 SHA(1 SaltLen(20) , 256 SaltLen(32) , 384 SaltLen(48) , 512 SaltLen(64))) SHA Val#3047
1888	**Microsoft Windows 10 November 2015 Update; Microsoft Surface Book, Surface Pro 4, Surface Pro 3, Surface 3, Surface Pro 2, and Surface Pro w/ Windows 10 November 2015 Update; Windows 10 Mobile for Microsoft Lumia 950 and Microsoft Lumia 635; Windows 10 for Microsoft Surface Hub 84" and Surface Hub 55" MsBignum Cryptographic Implementations** **FIPS186-4:** **ALG[RSASSA-PKCS1_V1_5]** SIG(gen) (2048 SHA(256 , 384 , 512)) (3072 SHA(256 , 384 , 512)) SIG(Ver) (1024 SHA(1 , 256 , 384 , 512)) (2048 SHA(1 , 256 , 384 , 512)) (3072 SHA(1 , 256 , 384 , 512)) SHA Val#3047

1889	**Microsoft Windows 10 November 2015 Update; Microsoft Surface Book, Surface Pro 4, Surface Pro 3, Surface 3, Surface Pro 2, and Surface Pro w/ Windows 10 November 2015 Update; Windows 10 Mobile for Microsoft Lumia 950 and Microsoft Lumia 635; Windows 10 for Microsoft Surface Hub 84" and Surface Hub 55" RSA Key Generation Implementation** **FIPS186-4:** **186-4KEY(gen):** FIPS186-4_Fixed_e (10001) **PGM(ProbPrimeCondition):** 2048 , 3072 **PPTT:** (C.3) SHA Val#3047 DRBG: Val# 955

3.2.7 Cryptographic Operation (for cryptographic hashing) (FCS_COP.1(3))

FCS_COP.1(3) is labeled FCS_COP.1(HASH) in [ST].

3.2.7.1 TSS Assurance Activity

Requirement met by the TOE

The evaluator shall check that the association of the hash function with other cryptographic functions (for example, the digital signature verification function) specified in the VPN Client ST (whether these are performed by the platform or by the TOE) is documented in the TSS.

[ST] section 6.3 Cryptographic Support indicates the TOE performs hash functions for VPN client functions. Search "the TSF provides other cryptographic operations".

3.2.7.2 Guidance Assurance Activities

The PP does not identify any TOE guidance assurance activities.

3.2.7.3 Test 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 only hashes messages that are an integral number of bytes in length; i.e., the length (in bits) of the message to be hashed is divisible by 8. The second mode is the bit-oriented mode. In this mode the TSF hashes messages of arbitrary length. As there are different tests for each mode, an indication is given in the following sections for the bit-oriented vs. the byte-oriented testmacs.

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

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

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

Selected Long Messages Test Bit-oriented Mode
The evaluators devise an input set consisting of m messages, where m is the block length of the hash algorithm. The length of the ith message is 512 + 99*i, where $1 \leq i \leq m$. The message text shall be pseudorandomly generated. The

evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF.

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

Pseudorandomly Generated Messages Test
This test is for byteoriented 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.

Windows 10 uses algorithm implementations validated under the CAVP (http://csrc.nist.gov/groups/STM/cavp/index.html). [ST] Table 11 Cryptographic Standards and Evaluation Methods identifies applicable CAVP SHA certificates: 3047 and 3048 (http://csrc.nist.gov/groups/STM/cavp/documents/shs/shaval.html). The relevant detail is reproduced and highlighted below.

3047	**Microsoft Windows 10 November 2015 Update; Microsoft Surface Book, Surface Pro 4, Surface Pro 3, Surface 3, Surface Pro 2, and Surface Pro w/ Windows 10 November 2015 Update; Windows 10 Mobile for Microsoft Lumia 950 and Microsoft Lumia 635; Windows 10 for Microsoft Surface Hub 84" and Surface Hub 55" SymCrypt Cryptographic Implementations** **SHA-1** (BYTE-only) **SHA-256** (BYTE-only) **SHA-384** (BYTE-only) **SHA-512** (BYTE-only)
3048	**Microsoft Windows 10 November 2015 Update; Microsoft Surface Book, Surface Pro 4, Surface Pro 3, Surface 3, Surface Pro 2, and Surface Pro w/ Windows 10 November 2015 Update; Windows 10 Mobile for Microsoft Lumia 950 and Microsoft Lumia 635; Windows 10 for Microsoft Surface Hub 84" and Surface Hub 55" RSA32 Algorithm Implementations** **SHA-1** (BYTE-only) **SHA-256** (BYTE-only) **SHA-384** (BYTE-only) **SHA-512** (BYTE-only)

3.2.8 Cryptographic Operation (for keyed-hash message authentication) (FCS_COP.1(4))

FCS_COP.1(4) is labeled FCS_COP.1(HMAC) in [ST].

3.2.8.1 TSS Assurance Activity

Requirement met by the TOE

The evaluator shall check that the association of the keyed-hash function with other cryptographic functions specified in the VPN Client ST (whether these are performed by the platform or by the TOE) is documented in the TSS.

[ST] section 6.3.1 IPsec indicates the TOE performs keyed-hash functions for ESP as well as the encrypted payload in IKEv1 and IKEv2. Search "Windows implements HMAC-SHA1, HMAC-SHA-256 and HMAC-SHA-384".

Requirement met by the TOE

Additionally, for all cases where the output of the HMAC following the hash calculation is truncated, the evaluator shall ensure that the TSS states for what operation this truncation takes place; the size of the final output; and the standard to which this truncation complies.

A footnote in [ST] section 6.3.1 IPsec states Windows truncates HMAC output in accordance with RFC 4868 and RFC 2404. Please see https://tools.ietf.org/html/rfc4868 section 2.3 Truncation and https://tools.ietf.org/html/rfc2404 section 2 Algorithm and Mode for truncation details.

Requirement met by the TOE

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

[ST] section 6.3 Cryptographic Support provides the required information in Table 9 HMAC Characteristics.

3.2.8.2 Guidance Assurance Activities

The PP does not identify any TOE guidance assurance test activities.

3.2.8.3 Test Activity

Requirement met by the TOE

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.

Windows 10 uses algorithm implementations validated under the CAVP (http://csrc.nist.gov/groups/STM/cavp/index.html). [ST] Table 11 Cryptographic Standards and Evaluation Methods identifies applicable CAVP HMAC certificate: 2381 (http://csrc.nist.gov/groups/STM/cavp/documents/mac/hmacval.html). The relevant detail is reproduced and highlighted below.

2381	**Microsoft Windows 10 November 2015 Update; Microsoft Surface Book, Surface Pro 4, Surface Pro 3, Surface 3, Surface Pro 2, and Surface Pro w/ Windows 10 November 2015 Update; Windows 10 Mobile for Microsoft Lumia 950 and Microsoft Lumia 635; Windows 10 for Microsoft Surface Hub 84" and Surface Hub 55" SymCrypt Cryptographic Implementations** **HMAC-SHA1 (Key Sizes Ranges Tested:** KS<BS KS=BS KS>BS) SHS Val#3047 **HMAC-SHA256 (Key Size Ranges Tested:** KS<BS KS=BS KS>BS) SHS Val#3047 **HMAC-SHA384 (Key Size Ranges Tested:** KS<BS KS=BS KS>BS) SHS Val#3047 **HMAC-SHA512 (Key Size Ranges Tested:** KS<BS KS=BS KS>BS) SHSVal#3047

3.2.9 Extended: Internet Protocol Security (IPsec) Communications (FCS_IPSEC_EXT.1.1)

3.2.9.1 TSS Assurance Activity

The PP does not identify any TOE TSS Assurance Activities.

3.2.9.2 Guidance Assurance Activities

The evaluator shall examine the operational guidance to verify it instructs the Administrator how to construct entries into the SPD that specify a rule for DISCARD, BYPASS and PROTECT.

[Guide] Section 5 *Managing the Windows Firewall (Windows Filtering Platform)* identifies the instructions to construct entries into the SPD that specify a rule for DISCARD, BYPASS and PROTECT. The Windows Filtering Platform is the IPsec Security Policy Database (SPD) for Windows 10.

Section 5 provides a link to TechNet topic "Overview of Windows Firewall with Advanced Security". An administrator configures actions DISCARD, BYPASS, and PROTECT through a combination of firewall rules and communication security rules. Firewall rules determine which traffic is allowed or blocked. Connection security rules determine the way in which traffic between the TOE and other computers is secured (that is, IPsec protected or BYPASS when absent). See section "How does Windows Firewall with Advanced Security work?" in "Overview of Windows Firewall with Advanced Security". Links in section 5 and TechNet provide guidance for firewall and communication security rules.

3.2.9.3 Test Activity

The evaluator uses the operational guidance to configure the TOE and platform to carry out the following tests:

Test 1: The evaluator shall configure the SPD such that there is a rule for DISCARD, BYPASS, PROTECT. The selectors used in the construction of the rule shall be different such that the evaluator can send in three network packets with the appropriate fields in the packet header that each packet will match one of the three rules. The evaluator observes via the audit trail, and packet captures that the TOE exhibited the expected behavior: appropriate packet was dropped, allowed through without modification, was encrypted by the IPsec implementation.

Test 2: The evaluator shall devise several tests that cover a variety of scenarios for packet processing. These scenarios must exercise the range of possibilities for SPD entries and processing modes as outlined in the ST. Potential areas to cover include rules with overlapping ranges and conflicting entries, inbound and outbound packets, and packets that establish SAs as well as packets that belong to established SAs. The evaluator shall verify, for each scenario, that the expected behavior is exhibited, and is consistent with both the ST and the operational guidance.

The evaluator performed these tests in conjunction with FCS_IPSEC_EXT.1.3

3.2.10 Extended: Internet Protocol Security (IPsec) Communications (FCS_IPSEC_EXT.1.2)

3.2.10.1 TSS Assurance Activity

The evaluator checks the TSS to ensure it states that the VPN can be established to operate in tunnel mode and/or transport mode (as selected).

[ST] section 6.3.1 IPsec states "Windows IPsec supports both tunnel mode and transport mode."

3.2.10.2 Guidance Assurance Activities

The evaluator shall confirm that the operational guidance contains instructions on how to configure the connection in each mode selected.

[Guide] Section 4 *IPsec Configuration with Transport Mode* identifies the instructions to configure the IPsec VPN Client for IKEv2 in transport mode.

[Guide] Section 3 *RAS IPsec VPN Client Configuration* identifies the instructions to configure the IPsec VPN Client for IKEv1 and IKEv2 in tunnel mode.

3.2.10.3 Test Activity

The evaluator shall perform the following test(s) based on the selections chosen:

Test 1 (conditional): If tunnel mode is selected, the evaluator uses the operational guidance to configure the TOE/platform to operate in tunnel mode and also configures a VPN GW to operate in tunnel mode. The evaluator configures the TOE/platform and the VPN GW to use any of the allowable cryptographic algorithms, authentication methods, etc. to ensure an allowable SA can be negotiated. The evaluator shall then initiate a connection from the

client to connect to the VPN GW peer. The evaluator observes (for example, in the audit trail and the captured packets) that a successful connection was established using the tunnel mode.

The evaluator configured the TOE to negotiate an IPsec connection using tunnel mode. The evaluator then initiated an IPsec connection from the TOE and verified the IPsec connection was successful using tunnel mode.

Test 2 (conditional): If transport mode is selected, the evaluator uses the operational guidance to configure the TOE/platform to operate in transport mode and also configures a VPN GW to operate in transport mode. The evaluator configures the TOE/platform and the VPN GW to use any of the allowed cryptographic algorithms, authentication methods, etc. to ensure an allowable SA can be negotiated. The evaluator then initiates a connection from the TOE/platform to connect to the VPN GW. The evaluator observes (for example, in the audit trail and the captured packets) that a successful connection was established using the transport mode.

The evaluator configured the TOE to negotiate an IPsec connection using transport mode. The evaluator then initiated and IPsec connection from the TOE and verified the IPsec connection was successful using transport mode.

3.2.11 Extended: Internet Protocol Security (IPsec) Communications (FCS_IPSEC_EXT.1.3)

3.2.11.1 TSS Assurance Activity

The evaluator shall examine the TSS to verify that the TSS provides a description of how a packet is processed against the SPD and that if no "rules" are found to match, that a final rule exists, either implicitly or explicitly, that causes the network packet to be discarded.

[ST] provides a high-level overview of processing against the SPD in section 6.3.1 IPsec. Search "The Windows IPsec implementation includes a security policy database (SPD)". The overview states that Windows will discard a packet when no rules in the SPD apply. Section 6.3.1 provides references to a more detailed description of SPD processing (7.5.3, Security Policy Database Structure: http://msdn.microsoft.com/en-us/library/jj663164.aspx) and related firewall rules (Understanding Firewall Rules: http://technet.microsoft.com/en-us/library/dd421709(v=WS.10).aspx).

3.2.11.2 Guidance Assurance Activities

The evaluator checks that the operational guidance provides instructions on how to construct the SPD.

[Guide] Section 5 *Managing the Windows Firewall (Windows Filtering Platform)* identifies the instructions to construct entries into the SPD that specify a rule for DISCARD, BYPASS and PROTECT. The (Windows Firewall) Windows Filtering Platform is the IPsec Security Policy Database (SPD) for Windows 10. The IPsec rules in the Windows Filtering Platform are entries in the SPD.

Web links are provided to provide the guidance to configure the Inbound and Outbound rules that DISCARD and ALLOW traffic specified by the Inbound and Outbound rules.

The web links state that "Block connection" rules have higher priority than "Allow connection" rules (where "Block connection" is equivalent to DISCARD and "Allow connection" is equivalent to ALLOW). Further, it says that the "Default profile behavior" when the Windows Filtering Platform is turned on (which is mandatory for IPSEC configuration) explicitly DISCARDs all network traffic that is not specified as ALLOWed by the combination of the IPSEC rules as well as Inbound and Outbound Firewall rules. The Windows Filtering Platform in this way implements the final catch-all denial SPD entry.

3.2.11.3 Test Activity

Test 1: The evaluator shall configure the SPD such that it has entries that contain operations that DISCARD, BYPASS, and PROTECT network packets. The evaluator may use the SPD that was created for verification of FCS_IPSEC_EXT.1.1. The evaluator shall construct a network packet that matches a BYPASS entry and send that packet. The evaluator should observe that the network packet is passed to the proper destination interface with no

modification. The evaluator shall then modify a field in the packet header; such that it no longer matches the evaluator-created entries (there may be a "TOE/platform created" final entry that discards packets that do not match any previous entries). The evaluator sends the packet, and observes that the packet was not permitted to flow to any of the TOE's interfaces.

The evaluator conducted tests to show the TOE could properly handle network packets by completing operations to DISCARD, BYPASS, and PROTECT the traffic based on Connection Security Rules and rules in the Windows Filtering Platform. The evaluator then sent packets to the TOE that did not match any of these rules and verified that the TOE dropped these packets and did not respond.

3.2.12 Extended: Internet Protocol Security (IPsec) Communications (FCS_IPSEC_EXT.1.4)

3.2.12.1 TSS Assurance Activity

The evaluator shall examine the TSS to verify that the algorithms AES-GCM-128 and AES-GCM-256 are implemented. If the ST author has selected either AES-CBC-128 or AES-CBC-256 in the requirement, then the evaluator verifies the TSS describes these as well. In addition, the evaluator ensures that the SHA-based HMAC algorithm conforms to the algorithms specified in FCS_COP.1(4) Cryptographic Operations (for keyed-hash message authentication).

[ST] section 6.3.1 IPsec claims Windows 10 implements encryption algorithms AES-GCM-128, AES-GCM-256, AES-CBC-128, and AES-CBC-256 and HMAC algorithms HMAC-SHA1 and SHA-256.The claims are consistent with FCS_IPSEC_EXT.1.4, FCS_COP.1(SYM), and FCS_COP.1(HASH). Search for "Windows 10 implements AES-GCM-128, AES-GCM-256, AES-CBC-128, and AES-CBC-256 as encryption algorithms".

3.2.12.2 Guidance Assurance Activities

The evaluator checks the operational guidance to ensure it provides instructions on how to configure the TOE/platform to use the AES-GCM-128, and AES-GCM-256 algorithms, and if either AES-CBC-128 or AES-CBC-256 have been selected the guidance instructs how to use these as well.

[Guide] Section 4.1 *Supported Algorithms* identifies the AES-CBC-128 AES-CBC-256, AES-GCM-128, and AES-GCM-256

[Guide] Section 4.2 *Configuring Cryptographic Algorithms* provides two TechNet links to provide the instructions to configure the encryption algorithms.

3.2.12.3 Test Activity

The evaluator shall configure the TOE/platform as indicated in the operational guidance configuring the TOE/platform to using each of the AES-GCM-128, and AES-GCM-256 algorithms, and attempt to establish a connection using ESP. If the ST Author has selected either AES-CBC-128 or AES-CBC-256, the TOE/platform is configured to use those algorithms and the evaluator attempts to establish a connection using ESP for those algorithms selected.

The evaluator performed separate tests that showed a successful IPsec negotiation using AES-GCM-128, AES-GCM-256, AES-CBC-128, and AES-CBC-256 for ESP. These tests were combined with FCS_IPSEC_EXT.1.6 to test AES-CBC-128 and AES-CBC-256 for IKE, FCS_IPSEC_EXT.1.11 to test DH Group 14, DH Group 19 and DH Group 20, and FCS_IPSEC_EXT.1.14 Test 1 to cover each algorithm for IKEv1 and IKEv2.

3.2.13 Extended: Internet Protocol Security (IPsec) Communications (FCS_IPSEC_EXT.1.5)

3.2.13.1 TSS Assurance Activity

The evaluator shall examine the TSS to verify that IKEv1 and/or IKEv2 are implemented.

[ST] section 6.3.1 IPsec states “Windows implements both RFCS 2409, Internet Key Exchange (IKEv1), and RFC 4306, Internet Key Exchange version 2, (IKEv2)”

3.2.13.2 Guidance Assurance Activities

The evaluator shall check the operational guidance to ensure it instructs the administrator how to configure the TOE/platform to use IKEv1 and/or IKEv2 (as selected), and uses the guidance to configure the TOE/platform to perform NAT traversal for the following test.

[Guide] Section 4.5 *Configuring the IKEv1 or IKEv2 Protocol in the IPsec Rule* provides the guidance how to configure the TOE/platform to use IKEv1 and/or IKEv2.

[Guide] Section 9 *Traversing a NAT* states that Windows 10 automatically traverses a NAT as specified in the IKEv1 and IKEv2 protocols and the SAs are negotiated. No configuration is necessary to accommodate a NAT in a deployment.

3.2.13.3 Test Activity

The evaluator shall configure the TOE/platform so that it will perform NAT traversal processing as described in the TSS and RFC 5996, section 2.23. The evaluator shall initiate an IPsec connection and determine that the NAT is successfully traversed.

The evaluator placed the TOE behind a NAT and attempted an IPsec connection to the VPN Gateway. This connection succeeded and the evaluator confirmed by viewing packets captures and IPsec Security Associations that the NAT was successfully traversed.

3.2.14 Extended: Internet Protocol Security (IPsec) Communications (FCS_IPSEC_EXT.1.6)

3.2.14.1 TSS Assurance Activity

The evaluator shall ensure the TSS identifies the algorithms used for encrypting the IKEv1 and/or IKEv2 payload, and that the algorithms AES-CBC-128, AES-CBC-256 are specified, and if others are chosen in the selection of the requirement, those are included in the TSS discussion.

[ST] section 6.3.1 IPsec claims Windows 10 implements encryption algorithms AES-CBC-128 and AES-CBC-256 for IKEv1 and IKEv2 (search for “AES-CBC-128 and AES-CBC-256 can be used for IKEv1 and IKEv2”).

3.2.14.2 Guidance Assurance Activities

The evaluator ensures that the operational guidance describes the configuration of the mandated algorithms, as well as any additional algorithms selected in the requirement. The guidance is then used to configure the TOE/platform to perform the following test for each ciphersuite selected.

[Guide] Section 4.2 *Configuring Cryptographic Algorithms* provides two TechNet links to provide the instructions to configure the encryption algorithms.

3.2.14.3 Test Activity

The evaluator shall configure the TOE/platform to use the ciphersuite under test to encrypt the IKEv1 and/or IKEv2 payload and establish a connection with a peer device, which is configured to only accept the payload encrypted using the indicated ciphersuite. The evaluator will confirm the algorithm was that used in the negotiation.

The evaluator performed tests for both IKEv1 and IKEv2 that showed a successful IPsec negotiation using each of the algorithms claimed in the ST. This was performed in conjunction with FCS_IPSEC_EXT.1.4.

3.2.15 Extended: Internet Protocol Security (IPsec) Communications (FCS_IPSEC_EXT.1.7)

3.2.15.1 TSS Assurance Activity

The evaluator shall examine the TSS to ensure that, in the description of the IPsec protocol, it states that aggressive mode is not used for IKEv1 Phase 1 exchanges, and that only main mode is used. It may be that this is a configurable option.

[ST] section 6.3.1 IPsec states "Windows operating systems do not implement the IKEv1 aggressive mode option during a Phase 1 key exchange."

3.2.15.2 Guidance Assurance Activities

If the mode requires configuration of the TOE/platform prior to its operation, the evaluator shall check the operational guidance to ensure that instructions for this configuration are contained within that guidance.

[Guide] Section 3.7 *Other Information* states that there is no way to configure Windows to use IKEv1 aggressive mode. Only main mode is supported.

3.2.15.3 Test Activity

Test 1 (conditional): The evaluator shall configure the TOE/platform as indicated in the operational guidance, and attempt to establish a connection using an IKEv1 Phase 1 connection in aggressive mode. This attempt should fail. The evaluator should then show that main mode exchanges are supported. This test is not applicable if IKEv1 is not selected above in the FCS_IPSEC_EXT.1.5 protocol selection.

Although the TOE is not configurable for aggressive mode, the evaluator attempted to connect to the TOE with an IPsec peer using IKEv1 Phase 1 in aggressive mode. The evaluator verified that this attempt did not succeed.

3.2.16 Extended: Internet Protocol Security (IPsec) Communications (FCS_IPSEC_EXT.1.8)

3.2.16.1 TSS Assurance Activity

The PP does not identify any TOE TSS assurance activities.

3.2.16.2 Guidance Assurance Activities

The evaluator verifies that the values for SA lifetimes can be configured and that the instructions for doing so are located in the operational guidance. If time-based limits are supported, the evaluator ensures that either the Administrator or VPN Gateway are able to configurable Phase 1 SAs values for 24 hours and 8 hours for Phase 2 SAs. Currently there are no values mandated for the number of packets or number of bytes, the evaluator just ensures that this can be configured if selected in the requirement.

[Guide] Section 4.3 *Configuring SA Lifetimes* provides instructions on how to configure SA lifetime values. [Guide] Section 4.3.1 *Configuring Main Mode SA Lifetimes* provides the instructions to configure the Main Mode SA lifetime values and [Guide] Section 4.3.2 *Configuring Quick Mode SA Lifetimes* provides the instructions to configure the Quick Mode SA Lifetimes.

3.2.16.3 Test Activity

The evaluator verifies that the values for SA lifetimes can be configured and that the instructions for doing so are located in the operational guidance. If time-based limits are supported, the evaluator ensures that either the Administrator or VPN Gateway are able to configurable Phase 1 SAs values for 24 hours and 8 hours for Phase 2 SAs. Currently there are no values mandated for the number of packets or number of bytes, the evaluator just ensures that this can be configured if selected in the requirement.

When testing this functionality, the evaluator needs to ensure that both sides are configured appropriately. From the RFC "A difference between IKEv1 and IKEv2 is that in IKEv1 SA lifetimes were negotiated. In IKEv2, each end of the SA is responsible for enforcing its own lifetime policy on the SA and rekeying the SA when necessary. If the two ends have different lifetime policies, the end with the shorter lifetime will end up always being the one to request the rekeying. If the two ends have the same lifetime policies, it is possible that both will initiate a rekeying at the same time (which will result in redundant SAs). To reduce the probability of this happening, the timing of rekeying requests SHOULD be jittered."

Each of the following tests shall be performed for each version of IKE selected in the FCS_IPSEC_EXT.1.5 protocol selection:

Test 1 (Conditional): The evaluator shall configure a maximum lifetime in terms of the # of packets (or bytes) allowed following the operational guidance. The evaluator shall establish an SA and determine that once the allowed # of packets (or bytes) through this SA is exceeded, the connection is closed.

The evaluator configured the TOE to have a SA lifetime of 20480 Kilobytes and brought initiated a connection. The connection succeeded and the evaluator sent pings through the tunnel until the lifetime was reached. Once the lifetime was reached, the evaluator verified that the SA was terminated and the TOE renegotiated a new SA. This was done for both IKEv1 and IKEv2.

Test 2 (Conditional): The evaluator shall construct a test where a Phase 1 SA is established and attempted to be maintained for more than 24 hours before it is renegotiated. The evaluator shall observe that this SA is closed or renegotiated in 24 hours or less. If such an action requires that the TOE be configured in a specific way, the evaluator shall implement tests demonstrating that the configuration capability of the TOE works as documented in the operational guidance.

The evaluator configured the TOE to have a Phase 1 SA lifetime of 24 hours and brought initiated a connection. The connection succeeded and the evaluator left the tunnel up until the lifetime was reached. Just before 24 hours, the evaluator verified that the SA was terminated and the TOE renegotiated a new SA. This was done for both IKEv1 and IKEv2.

Test 3 (Conditional): The evaluator shall perform a test similar to Test 1 for Phase 2 SAs, except that the lifetime will be 8 hours instead of 24.

The evaluator configured the TOE to have a Phase 2 SA lifetime of 8 hours and brought initiated a connection. The connection succeeded and the evaluator left the tunnel up until the lifetime was reached. Just before 8 hours, the evaluator verified that the SA was terminated and the TOE renegotiated a new SA. This was done for both IKEv1 and IKEv2.

3.2.17 Extended: Internet Protocol Security (IPsec) Communications (FCS_IPSEC_EXT.1.9)

3.2.17.1 TSS Assurance Activity

The evaluator shall check to ensure that, for each DH group supported, the TSS describes the process for generating "x" (as defined in FCS_IPSEC_EXT.1.9) and each nonce. The evaluator shall verify that the TSS indicates that the random number generated that meets the requirements in this PP is used, and that the length of "x" and the nonces meet the stipulations in the requirement.

[ST] section 6.3.1 IPsec states "Windows uses a FIPS validated random number generator to generate 'x' with length 224, 256, or 384 bits for DH groups 14, 19, and 20 respectively". Assurance activity test *Key Generation for Finite-Field Cryptography (FFC) – Based 56A Schemes* in section 3.2.1.3 of section 3.2.1 Cryptographic Key Generation (for asymmetric keys) (FCS_CKM.1(1)) confirms key generation for Diffie-Hellman is CAVP validated including use of the CAVP-validated random bit generator identified for FCS_RBG_EXT.1. Similarly, assurance activity test Key *Generation for Elliptic Curve Cryptography (ECC)- Based 56A Schemes* in section 3.2.1.3 of section 3.2.1 Cryptographic Key Generation (for asymmetric keys) (FCS_CKM.1(1)) confirms key generation for

Elliptic Curve Diffie-Hellman is CAVP validated including use of the CAVP-validated random bit generator identified for FCS_RBG_EXT.1.

3.2.17.2 Guidance Assurance Activities

The PP does not identify any guidance assurance activities for this SFR.

3.2.17.3 Test Activity

The PP does not identify any test assurance activities for this SFR.

3.2.18 Extended: Internet Protocol Security (IPsec) Communications (FCS_IPSEC_EXT.1.10)

3.2.18.1 TSS Assurance Activity

The evaluator shall check to ensure that, for each DH group supported, the TSS describes the process for generating "x" (as defined in FCS_IPSEC_EXT.1.9) and each nonce. The evaluator shall verify that the TSS indicates that the random number generated that meets the requirements in this PP is used, and that the length of "x" and the nonces meet the stipulations in the requirement.

[ST] selects Diffie-Hellman groups 14 (2048-bit MODP), 19 (256-bit Random ECP), and 20 (384-bit Random ECP) in FCS_IPSEC_EXT.1.11. Table 2 of NIST SP 800-57 gives the corresponding security strengths as 112, 128, and 192, respectively. [ST] section 6.3.1 IPsec states that Windows uses 32-byte nonces (that is, 256 bits). In regards to the assignment "(one or more) "bits of security" value(s) associated with the negotiated Diffie-Hellman group as listed in Table 2 of NIST SP 800-57, *Recommendation for Key Management – Part 1: General*", 2^{-256} is less than each of 2^{-112}, 2^{-128}, and 2^{-192}, since 256 exceeds 112, 128, and 192.

3.2.18.2 Guidance Assurance Activities

The PP does not identify any guidance assurance activities for this SFR.

3.2.18.3 Test Activity

The PP does not identify any test assurance activities for this SFR.

3.2.19 Extended: Internet Protocol Security (IPsec) Communications (FCS_IPSEC_EXT.1.11)

3.2.19.1 TSS Assurance Activity

The evaluator shall check to ensure that the DH groups specified in the requirement are listed as being supported in the TSS. If there is more than one DH group supported, the evaluator checks to ensure the TSS describes how a particular DH group is specified/negotiated with a peer.

[ST] selects Diffie-Hellman groups 14 (2048-bit MODP), 19 (256-bit Random ECP), and 20 (384-bit Random ECP) in FCS_IPSEC_EXT.1.11. [ST] section 6.3.1 IPsec identifies Diffie-Hellman groups 14, 19, and 20. Search for "Diffie-Hellman Groups 14, 19, and 20". Section 6.3.1 describes crypto suite negotiation (search for "The IPsec VPN client will propose a cryptosuite").

3.2.19.2 Guidance Assurance Activities

The PP does not identify any guidance assurance activities for this SFR.

3.2.19.3 Test Activity

For each supported DH group, the evaluator shall test to ensure that all supported IKE protocols can be successfully completed using that particular DH group.

The evaluator performed tests for both IKEv1 and IKEv2 that showed a successful IPsec negotiation using each of the algorithms claimed in the ST. This was performed in conjunction with FCS_IPSEC_EXT.1.4.

3.2.20 Extended: Internet Protocol Security (IPsec) Communications (FCS_IPSEC_EXT.1.12)

3.2.20.1 TSS Assurance Activity

The evaluator ensures that the TSS identifies RSA and/or ECDSA as being used to perform peer authentication. The description must be consistent with the algorithms as specified in FCS_COP.1(2) Cryptographic Operations (for cryptographic signature).

[ST] selects RSA and ECDSA in FCS_COP.1(SIGN). Section 6.3.1 IPsec describes both RSA and ECDSA. Search for "Windows implements peer authentication using".

If pre-shared keys are chosen in the selection, the evaluator shall check to ensure that the TSS describes how pre-shared keys are established and used in authentication of IPsec connections. The description in the TSS shall also indicate how pre-shared key establishment is accomplished for TOEs/platforms that can generate a pre-shared key as well as TOEs/platforms that simply use a pre-shared key.

[ST] selects RSA and ECDSA in FCS_IPSEC_EXT.1.12. Section 6.3.1 IPsec explains Windows uses pre-shared keys as entered with no additional processing. Search for "While Windows supports pre-shared IPsec keys".

3.2.20.2 Guidance Assurance Activities

The evaluator shall check that the operational guidance describes how pre-shared keys are to be generated and established. The description in the operational guidance shall also indicate how pre-shared key establishment is accomplished for TOEs/platforms that can generate a pre-shared key as well as TOEs/platforms that simply use a pre-shared key.

[Guide] Section 3.2 *Configuring Pre-Shared Key for IKEv1* states that the pre-shared key is generated out of band and provided to the client for configuration.

[Guide] provides additional instructions stating that the secret value for the pre-shared key must be a text-based value manually entered as shown in the **Key** editbox in the **Advanced Properties** dialog below. The secret value must match the secret value configured on the VPN server. While the secret can be any length, it should include at least 22 characters and up to 100 characters as determined at the discretion of the administrator. For example organizational policies can enforce the use of strong passwords containing a minimum number of characters using at least one upper and one lower case letter, one number, and one special character from among the following: ! @ # $ % ^ & * ().

The evaluator ensures the operational guidance describes how to set up the TOE/platform to use the cryptographic algorithms RSA and/or ECDSA.

[Guide] Section 4.4 *Configuring Signature Algorithms* identifies the RSA, ECDSA P256, and ECDSA P384 as signature algorithms that are supported for IPsec authentication with certificates.

A TechNet is provided that identifies how to configure the authentication techniques to be used and the signature algorithms to use with certificate authentication:

In order to construct the environment and configure the TOE/platform for the following tests, the evaluator will ensure that the operational guidance also describes how to configure the TOE/platform to connect to a trusted CA, and ensure a valid certificate for that CA is loaded into the TOE/platform and marked "trusted".

[Guide] Section 6 *Managing Certificates* states that Windows 10 authentication certificates are obtained through MDM, domain policy or manually. TechNet links are provided describing how to manage, deploy, and delete certificates.

3.2.20.3 Test Activity

For efficiency sake, the testing that is performed here has been combined with the testing for FIA_X509_EXT.2.1 (for IPsec connections), FCS_IPSEC_EXT.1.13, and FIA_X509_EXT.2.3. The following tests shall be repeated for each peer authentication protocol selected in the FCS_IPSEC_EXT.1.12 selection above:

Test 1: The evaluator shall have the TOE/platform generate a public-private key pair, and submit a CSR (Certificate Signing Request) to a CA (trusted by both the TOE/platform and the peer VPN used to establish a connection) for its signature. The values for the DN (Common Name, Organization, Organizational Unit, and Country) will also be passed in the request.

The evaluator generated a CSR on the TOE that contained a public key, CN, OU, O and C and submitted it to a CA to be signed. The CA signed the request and the evaluator imported the certificate onto the TOE. Once imported, the evaluator confirmed that the public and DN values passed in the request matched the ones in the certificate.

NIAP Technical Decision 0037 modifies tests 2 and 4 under FCS_IPSEC_EXT.1.12.

Test 2: The evaluator shall use a certificate signed using the RSA or ECDSA algorithm to authenticate the remote peer during the IKE exchange. This test ensures the remote peer has the certificate for the trusted CA that signed the TOE's certificate and it will do a bit-wise comparison on the DN. This bit-wise comparison of the DN ensures that not only does the peer have a certificate signed by the trusted CA, but the certificate is from the DN that is expected. The evaluator will configure the TOE/platform to associate a certificate (e.g., a certificate map in some implementations) with a VPN connection. This is what the DN is checked against.

TD0037 specifies this test being run with FCS_IPSEC_EXT.1.13.

Test 3: The evaluator shall test that the TOE/platform can properly handle revoked certificates – conditional on whether CRL or OCSP is selected; if both are selected, and then a test is performed for each method. For this draft of the PP, the evaluator has to only test one up in the trust chain (future drafts may require to ensure the validation is done up the entire chain). The evaluator shall ensure that a valid certificate is used, and that the SA is established. The evaluator then attempts the test with a certificate that will be revoked (for each method chosen in the selection) to ensure when the certificate is no longer valid that the TOE/platform will not establish an SA.

This test was performed in conjunction with FIA_X509_EXT.1 Test 3.

NIAP Technical Decision 0037 modifies tests 2 and 4 under FCS_IPSEC_EXT.1.12.

Test 4: The evaluator shall test that given a signed certificate from a trusted CA, that when the DN does not match – any of the four fields can be modified such that they do not match the expected value, that an SA does not get established.

TD0037 specifies this test being run with FCS_IPSEC_EXT.1.13.

NIAP Technical Decision 0053 removes Test 5 from assurance activities.

~~Test 5: The evaluator shall ensure that the TOE is configurable to either establish an SA, or not establish an SA if a connection to the certificate validation entity cannot be reached. For each method selected for certificate validation, the evaluator attempts to validate the certificate – for the purposes of this test, it does not matter if the certificate is revoked or not. For the "mode" where an SA is allowed to be established, the connection is made. Where the SA is not to be established, the connection is refused.~~

Test 6 [conditional]: The evaluator shall generate a pre-shared key and use it, as indicated in the operational guidance, to establish an IPsec connection with the VPN GW peer. If the generation of the pre-shared key is supported, the evaluator shall ensure that establishment of the key is carried out for an instance of the TOE/platform generating the key as well as an instance of the TOE/platform merely taking in and using the key..

N/A—The TOE does not generate pre-shared keys

3.2.21 Extended: Internet Protocol Security (IPsec) Communications (FCS_IPSEC_EXT.1.13)

3.2.21.1 TSS Assurance Activity

NIAP Technical Decision 0037 replaces FCS_IPSEC_EXT.1.13

~~Assurance activities for this element are tested through assurance activities for FCS_IPSEC_EXT.1.12.~~

The evaluator shall ensure that the TSS describes how the TOE compares the peer's presented identifier to the reference identifier. This description shall include whether the certificate presented identifier is compared to the ID payload presented identifier, which field(s) of the certificate are used as the presented identifier (DN, Common Name, or SAN), and, if multiple fields are supported, the logical order comparison. If the ST author assigned an additional identifier type, the TSS description shall also include a description of that type and the method by which that type is compared to the peer's presented certificate.

[ST] selects reference identifiers IP address, FQDN, and user FQDN in FCS_IPSEC_EXT.1.13 as amended by NIAP Technical Decision 0037. [ST] section 6.3.1 IPsec describes direct comparison of Common Name to the configured reference identifier. Search for "by comparing the Common Name of the certificate presented by the VPN gateway".

3.2.21.2 Guidance Assurance Activities

NIAP Technical Decision 0037 replaces FCS_IPSEC_EXT.1.13

~~Assurance activities for this element are tested through assurance activities for FCS_IPSEC_EXT.1.12.~~

The evaluator shall ensure that the operational guidance includes the configuration of the reference identifier(s) for the peer.

[Guide] section 4.4 Configuring Signature Algorithms provides the guidance to configure the reference identifier(s) for the peer.

3.2.21.3 Test Activity

NIAP Technical Decision 0037 replaces FCS_IPSEC_EXT.1.13

~~Assurance activities for this element are tested through assurance activities for FCS_IPSEC_EXT.1.12.~~

For each supported identifier type (excluding DNs), the evaluator shall repeat the following tests:

Test 1: For each field of the certificate supported for comparison, the evaluator shall configure the peer's reference identifier on the TOE (per the administrative guidance) to match the field in the peer's presented certificate and shall verify that the IKE authentication succeeds.

For each identifier type, the evaluator configured the one on the TOE to match the one presented by the IPsec peer. The evaluator attempted an IPsec connection and verified that the connection succeeds.

Test 2: For each field of the certificate support for comparison, the evaluator shall configure the peer's reference identifier on the TOE (per the administrative guidance) to not match the field in the peer's presented certificate and shall verify that the IKE authentication fails.

For each identifier type, the evaluator configured the one on the TOE to not match the one presented by the IPsec peer. The evaluator attempted an IPsec connection and verified that the connection does not succeed.

The following tests are conditional:

Test 3: (conditional) If, according to the TSS, the TOE supports both Common Name and SAN certificate fields and uses the preferred logic outlined in the Application Note, the tests above with the Common Name field shall be performed using peer certificates with no SAN extension. Additionally, the evaluator shall configure the peer's reference identifier on the TOE to not match the SAN in the peer's presented certificate but to match the Common Name in the peer's presented certificate, and verify that the IKE authentication fails.

N/A—the TOE does not support both Common Name and SAN certificate fields.

Test 4: (conditional) If the TOE supports DN identifier types, the evaluator shall configure the peer's reference identifier on the TOE (per the administrative guidance) to match the subject DN in the peer's presented certificate and shall verify that the IKE authentication succeeds. To demonstrate a bit-wise comparison of the DN, the evaluator shall change a single bit in the DN (preferably, in an Object Identifier (OID) in the DN) and verify that the IKE authentication fails.

N/A—the TOE does not support DN identifier types.

Test 5: (conditional) If the TOE supports both IPv4 and IPv6 and supports IP address identifier types, the evaluator must repeat test 1 and 2 with both IPv4 address identifiers and IPv6 identifiers. Additionally, the evaluator shall verify that the TOE verifies that the IP header matches the identifiers by setting the presented identifiers and the reference identifier with the same IP address that differs from the actual IP address of the peer in the IP headers and verifying that the IKE authentication fails.

The evaluator tested both IPv4 and IPv6 IPsec connections using a valid IP reference identifier and an invalid IP reference identifier. The evaluator confirmed that a connection was established using the valid reference identifier and a connection was not established using the invalid reference identifier.

Test 6: (conditional) If, according to the TSS, the TOE performs comparisons between the peer's ID payload and the peer's certificate, the evaluator shall repeat the following test for each combination of supported identifier types and supported certificate fields (as above). The evaluator shall configure the peer to present a different ID payload than the field in the peer's presented certificate and verify that the TOE fails to authenticate the IKE peer.

N/A—The TOE does not perform comparisons between the peer's ID payload and the peer's certificate.

3.2.22 Extended: Internet Protocol Security (IPsec) Communications (FCS_IPSEC_EXT.1.14)

3.2.22.1 TSS Assurance Activity

The evaluator shall check that the TSS describes the potential strengths (in terms of the number of bits in the symmetric key) of the algorithms that are allowed for the IKE and ESP exchanges.

[ST] section 6.3.1 IPsec describes the potential strength of algorithms in terms of 128 and 256 bit AES symmetric keys. Search for "The resulting potential strength of the symmetric key".

The TSS shall also describe the checks that are done when negotiating IKEv1 Phase 2 and/or IKEv2 CHILD_SA suites to ensure that the strength (in terms of the number of bits of key in the symmetric algorithm) of the negotiated algorithm is less than or equal to that of the IKE SA this is protecting the negotiation.

[ST] section 6.3.1 IPsec:

> In order to prevent security being reduced while transitioning from IKE Phase 1 / IKEv2 SA, an authorized administrator must configure the IPsec VPN client such that algorithms with same strength are used for both IKE Phase 1 and Phase 2 as well as for IKEv2 SA and IKEv2 Child SA.

[Guide] contains supporting guidance in section 3.4.1 Configuring the Cryptographic Algorithms for IKEv1 and IKEv2 and 4.2 Configuring Cryptographic Algorithms. [Guide] emphasizes that the strength of an IKE Phase 2 SA (IKEv2 Child SA) must not be stronger than the corresponding IKE Phase 1 SA (respectively, IKEv2 SA).

3.2.22.2 Guidance Assurance Activities

The evaluator simply follows the guidance to configure the TOE/platform to perform the following tests.

The guidance instructions for this assurance activity are identified in the assurance activities listed above.

3.2.22.3 Test Activity

Test 1: This test shall be performed for each version of IKE supported. The evaluator shall successfully negotiate an IPsec connection using each of the supported algorithms and hash functions identified in the requirements.

The evaluator performed tests for both IKEv1 and IKEv2 that showed a successful IPsec negotiation using each of the algorithms claimed in the ST. This was performed in conjunction with FCS_IPSEC_EXT.1.4.

Test 2: This test shall be performed for each version of IKE supported. The evaluator shall attempt to establish an SA for ESP that selects an encryption algorithm with more strength than that being used for the IKE SA (i.e., symmetric algorithm with a key size larger than that being used for the IKE SA). Such attempts should fail.

N/A—Covered by guidance. The Operational Guidance specifies that a Quick Mode encryption algorithm must not be configured with a greater strength than the Main Mode encryption algorithm.

Test 3: This test shall be performed for each version of IKE supported. The evaluator shall attempt to establish an IKE SA using an algorithm that is not one of the supported algorithms and hash functions identified in the requirements. Such an attempt should fail.

The evaluator configured the IPsec peer to use an IKE algorithm that is not claimed by the TOE. The evaluator attempted a connection and verified that it did not succeed. This was done for both IKEv1 and IKEv2.

Test 4: This test shall be performed for each version of IKE supported. The evaluator shall attempt to establish an SA for ESP (assumes the proper parameters where used to establish the IKE SA) that selects an encryption algorithm that is not identified in FCS_IPSEC_EXT.1.4. Such an attempt should fail.

The evaluator configured the IPsec peer to use an IKE algorithm that is not claimed by the TOE. The evaluator attempted a connection and verified that it did not succeed. This was done for both IKEv1 and IKEv2.

3.2.23 Extended: Cryptographic operation (Random Bit Generation) (FCS_RBG_EXT.1)

3.2.23.1 TSS Assurance Activity

Requirement met by the TOE

Documentation shall be produced—and the evaluator shall perform the activities—in accordance with Appendix E.

If the ST author has selected a platform-based noise source, the evaluator shall verify the platform's RBG has been validated by examining the platform's ST. The evaluator shall verify that the platform's RBG is seeded with at least the amount of entropy selected by the ST author for this profile. In this case, the ST author is not responsible for Annex E documentation of the platform's RBG.

Microsoft provided NIAP with the documentation specified in Appendix E. [ST] selects "a platform-based RBG" in FCS_RBG_EXT.1.2 to account for Surface Pro 4 and Surface Book hardware entropy sources (for example, RDRAND CPU instruction and Trusted Platform Module). Microsoft covered these hardware entropy sources in the entropy documentation. [ST] section 6.3 Cryptographic Support includes a high-level overview of Windows entropy sources and random number generation. Search for "RBG which is seeded from the Windows entropy pool."

3.2.23.2 Guidance Assurance Activities

The PP does not identify any TOE Guidance Assurance Activities.

3.2.23.3 Test Activity

Requirement met by the TOE

NIAP Technical Decision 0079 removes option" FIPS Pub 140-2 Annex C: X9.31 Appendix 2.4 using AES" and removes the corresponding assurance activities.

The evaluator shall perform the following tests.

~~**Implementations Conforming to FIPS 140-2, Annex C**~~

~~The reference for the tests contained in this section is The Random Number Generator Validation System (RNGVS). The evaluators shall conduct the following two tests. Note that the "expected values" are produced by a reference implementation of the algorithm that is known to be correct. Proof of correctness is left to each Scheme.~~

~~The evaluators shall perform a Variable Seed Test. The evaluators shall provide a set of 128 (Seed, DT) pairs to the TSF RBG function, each 128 bits. The evaluators shall also provide a key (of the length appropriate to the AES algorithm) that is constant for all 128 (Seed, DT) pairs. The DT value is incremented by 1 for each set. The seed values shall have no repeats within the set. The evaluators ensure that the values returned by the TSF match the expected values.~~

~~The evaluators shall perform a Monte Carlo Test. For this test, they supply an initial Seed and DT value to the TSF RBG function; each of these is 128 bits. The evaluators shall also provide a key (of the length appropriate to the AES algorithm) that is constant throughout the test. The evaluators then invoke the TSF RBG 10,000 times, with the DT value being incremented by 1 on each iteration, and the new seed for the subsequent iteration produced as specified in NIST Recommended Random Number Generator Based on ANSI X9.31 Appendix A.2.4 Using the 3-Key Triple DES and AES Algorithms, Section 3. The evaluators ensure that the 10,000th value produced matches the expected value.~~

~~**Implementations Conforming to NIST Special Publication 800-90**~~

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

If the RNG has prediction resistance enabled, each trial consists of (1) instantiate drbg, (2) generate the first block of random bits (3) generate a second block of random bits (4) uninstantiate. The evaluator verifies that the second block of random bits is the expected value. The evaluator shall generate eight input values for each trial. The first is a count (0 – 14). The next three are entropy input, nonce, and personalization string for the instantiate operation. The next two are additional input and entropy input for the first call to generate. The final two are additional input and entropy input for the second call to generate. These values are randomly generated. “Generate one block of random bits” means to generate random bits with number of returned bits equal to the Output Block Length (as defined in NIST SP 800-90A).

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

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

Entropy input: the length of the entropy input value must equal the seed length.

Nonce: If a nonce is supported (CTR_DRBG with no df does not use a nonce), the nonce bit length is one-half the seed length.

Personalization string: The length of the personalization string must be <= seed length. If the implementation only supports one personalization string length, then the same length can be used for both values. If more than one string length is support, the evaluator shall use personalization strings of two different lengths. If the implementation does not use a personalization string, no value needs to be supplied.

Additional input: the additional input bit lengths have the same defaults and restrictions as the personalization string lengths.

Windows uses algorithm implementations validated under the CAVP (http://csrc.nist.gov/groups/STM/cavp/index.html). [ST] Table 11 Cryptographic Standards and Evaluation Methods identifies applicable CAVP DRBG certificate: 955

(http://csrc.nist.gov/groups/STM/cavp/documents/drbg/drbgnewval.html). The relevant detail is reproduced and highlighted below.

955	**Microsoft Windows 10 November 2015 Update; Microsoft Surface Book, Surface Pro 4, Surface Pro 3, Surface 3, Surface Pro 2, and Surface Pro w/ Windows 10 November 2015 Update; Windows 10 Mobile for Microsoft Lumia 950 and Microsoft Lumia 635; Windows 10 for Microsoft Surface Hub 84" and Surface Hub 55" SymCrypt Cryptographic Implementations** **CTR_DRBG**: [Prediction Resistance Tested: Not Enabled; BlockCipher_Use_df: (AES-256) (AES Val#3629)]

3.3 User Data Protection (FDP)

3.3.1 Subset Information Flow Control (FDP_IFC_EXT.1)

3.3.1.1 TSS Assurance Activities

The evaluator shall verify that the TSS section of the ST describes the routing of IP traffic through processes on the TSF when a VPN client is enabled.

[ST] section 6.4.1 IPsec VPN Tunnels describes routing of traffic through the IPsec VPN client. Search for "The administrator can also configure the IPsec VPN client that all IP traffic".

The evaluator shall ensure that the description indicates which traffic does not go through the VPN and which traffic does and that a configuration exists for each baseband protocol in which only the traffic identified by the ST author as necessary for establishing the VPN connection (IKE traffic and perhaps HTTPS or DNS traffic) is not encapsulated by the VPN protocol (IPsec).

[ST] section 6.4.1 IPsec VPN Tunnels identifies traffic not routed through the IPsec VPN client. Search for "all IP traffic is routed through the IPsec tunnel except for". Section 6.4.1 states there is a configuration that routes all traffic through the IPsec VPN client except the identified connection establishment traffic. Search for "The administrator can also configure the IPsec VPN client that all IP traffic".

The ST author shall also identify in the TSS section any differences in the routing of IP traffic when using any supported baseband protocols (e.g. WiFi or, LTE).

[ST] section 6.4.1 IPsec VPN Tunnels indicates there is no difference for supported baseband protocols. Search "The IPsec VPN is an end-to-end internetworking technology".

3.3.1.2 Guidance Assurance Activities

The evaluator shall verify that the following is addressed by the documentation:

- The description above indicates that if a VPN client is enabled, all configurations route all IP traffic (other than IP traffic required to establish the VPN connection) through the VPN client.
- The AGD guidance describes how the user and/or administrator can configure the TSF to meet this requirement.

[Guide] Section 3.1 *Add a VPN Connection* states that Windows can be configured to require all traffic to route through the IPsec tunnel by creating Firewall rules that prevent all traffic that is not routed through the VPN or by using a Lockdown VPN connection deployed through an MDM. [Guide] Section 5 *Managing the Windows Firewall (Windows Filtering Platform)* provides the guidance on how to set Firewall rules.

3.3.1.3 Test Activities

The evaluator shall also perform the following tests.

Test 1: If the ST author identifies any differences in the routing between WiFi and cellular protocols, the evaluator s hall repeat this test with a base station implementing one of the identified cellular protocols.

Step 1 - The evaluator shall enable a WiFi configuration as described in the AGD guidance. The evaluator shall use a packet sniffing tool between the platform and an Internet-connected network. The evaluator shall turn on the sniffing tool and perform actions with the device such as navigating to websites, using provided applications, and accessing other Internet resources. The evaluator shall verify that the sniffing tool captures the traffic generated by these actions, turn off the sniffing tool, and save the session data.

Step 2 -The evaluator shall configure an IPsec VPN client that supports the routing specified in this requirement, and if necessary, configure the device to perform the routing specified as described in the AGD guidance. The evaluator shall turn on the sniffing tool, establish the VPN connection, and perform the same actions with the device as performed in the first step. The evaluator shall verify that the sniffing tool captures traffic generated by these actions, turn off the sniffing tool, and save the session data.

Step 3 - The evaluator shall examine the traffic from both step one and step two to verify that all IP traffic, aside from and after traffic necessary for establishing the VPN (such as IKE, DNS, and possibly HTTPS), is encapsulated by IPsec. The evaluator shall be aware that IP traffic on the cellular baseband outside of the IPsec tunnel may be emanating from the baseband processor and shall verify with the manufacturer that any identified traffic is not emanating from the application processor.

Step 4 - The evaluator shall attempt to send packets to the TOE outside the VPN tunnel (i.e. not through the VPN gateway), including from the local wireless network, and shall verify that the TOE discards them.

The evaluator connected the TOE to an IPsec VPN and verified traffic was encapsulated by IPsec when connected to the VPN and traffic was plaintext when not connected to the VPN. The evaluator configured the TOE to not allow traffic to or from outside the VPN when connected to the VPN and verified that any attempt to bypass the VPN was denied by the TOE or dropped by the TOE.

3.3.2 Residual Information Protection (FDP_RIP.2)

3.3.2.1 TSS Assurance Activities

"Resources" in the context of this requirement are network packets being sent through (as opposed to "to", as is the case when a security administrator connects to the TOE) the TOE. The concern is that once a network packet is sent, the buffer or memory area used by the packet still contains data from that packet, and that if that buffer is re-used, those data might remain and make their way into a new packet. The evaluator shall check to ensure that the TSS describes packet processing to the extent that they can determine that no data will be reused when processing network packets.

[ST] section 6.4.2 Memory Management and Object Reuse describes general Windows mechanisms that provide residual information protection as well as how those mechanisms apply to network packet resources. The description covers allocation of buffers for network packets, overwriting of buffers on allocation, in-place encryption of network packets, overwriting of memory allocated to subjects, and execution context initialization. Search for "The TSF ensures that resources processed by the kernel or are exported to user-mode processes do not have residual information in the following ways".

The evaluator shall ensure that this description at a minimum describes how the previous data are zeroized/overwritten, and at what point in the buffer processing this occurs.

[ST] section 6.4.2 Memory Management and Object Reuse explains memory for a network packet buffer is overwritten with data or zeros before being assigned to the buffer. Search for "which includes memory allocated for network packets".

3.3.2.2 Guidance Assurance Activities

The PP does not identify any guidance assurance activities for this SFR.

3.3.2.3 Test Activities

The PP does not identify any test assurance activities for this SFR.

3.4 Identification and Authentication (FIA)

3.4.1 Extended: Pre-Shared Key Composition (FIA_PSK_EXT.1)

3.4.1.1 TSS Assurance Activity

Requirement met by the TOE

The evaluator shall examine the TSS to ensure that it states that text-based pre-shared keys of 22 characters are supported. If "text-based pre-shared keys" is selected, the evaluator shall confirm that the TSS states the conditioning that takes place to transform the text-based pre-shared key from the key sequence entered by the user (e.g., ASCII representation) to the bit string used by IPsec, and that this conditioning is consistent with the last selection in the FIA_PSK_EXT.1.3 requirement.

[ST] section 6.5.1 IPsec and Pre-shred Keys describes composition and conditioning of pre-shared keys. Windows 10 supports 22-character pre-shared keys. Windows does not perform any conditioning on the keys, which is consistent with the selections made in FIA_PSK_EXT.1.3. Search for "IPsec is the only protocol in this evaluation which supports the use of pre-shared keys."

Requirement met by the TOE

If "bit-based pre-shared keys" is selected, the evaluator shall confirm the operational guidance contains instructions for either entering bit-based pre-shared keys for each protocol identified in the requirement, or generating a bit-based pre-shared key (or both). The evaluator shall also examine the TSS to ensure it describes the process by which the bit-based pre-shared keys are generated (if the TOE supports this functionality), and confirm that this process uses the RBG specified in FCS_RBG_EXT.1.

This assurance activity is not applicable to Windows 10 because [ST] does not select bit-based pre-shared keys in FIA_PSK_EXT.1.3.

3.4.1.2 Guidance Assurance Activities

The evaluator shall examine the operational guidance to determine that it provides guidance on the composition of strong text-based pre-shared keys, and (if the selection indicates keys of various lengths can be entered) that it provides information on the merits of shorter or longer pre-shared keys.

[Guide] Section 3.2 *Configuring Pre-Shared Key for IKEv1* provides guidance on the composition of strong text-based pre-shared keys.

The secret value for the pre-shared key must be a text-based value manually entered as shown in the **Key** editbox in the **Advanced Properties** dialog below. The secret value must match the secret value configured on the VPN server. While the secret can be any length, it should include at least 22 characters and up to 100 characters as determined at the discretion of the administrator. For example organizational policies can enforce the use of strong passwords containing a minimum number of characters using at least one upper and one lower case letter, one number, and one special character from among the following: ! @ # $ % ^ & * ().

The guidance must specify the allowable characters for pre-shared keys, and that list must be a super-set of the list contained in FIA_PSK_EXT.1.2.

[Guide] Section 3.2 *Configuring Pre-Shared Key for IKEv1* provides guidance on the composition of strong text-based pre-shared keys. The guidance documentation identifies the same character list as in FIA_PSK_EXT.1.2.

If "bit-based pre-shared keys" is selected, the evaluator shall confirm the operational guidance contains instructions for either entering bit-based pre-shared keys for each protocol identified in the requirement, or generating a bit-based pre-shared key (or both).

The Security Target does not identify the use of "bit-based pre-shared keys". This guidance requirement is not applicable.

3.4.1.3 Test Activity

Regardless of whether this capability is implemented by the TOE or by the platform, the tests listed in the "Requirement met by the TOE" section must still be performed for each platform claimed in the ST.

The evaluator shall also perform the following tests:

Test 1: The evaluator shall compose a pre-shared key of 22 characters that contains a combination of the allowed characters in accordance with the operational guidance, and demonstrates that a successful protocol negotiation can be performed with the key.

The evaluator composed a key of 22 characters and configured the TOE to use it for an IPsec connection. The evaluator attempted the connection and verified the connection succeeded.

Test 2 [conditional]: If the TOE supports pre-shared keys of multiple lengths, the evaluator shall repeat Test 1 using the minimum length; the maximum length; and an invalid length. The minimum and maximum length tests should be successful, and the invalid length must be rejected by the TOE.

The evaluator composed keys of minimum length, maximum length and an invalid length. For each of these keys, the evaluator configured the TOE to use it with an IPsec connection and attempted the connection. The evaluator confirmed that attempts using the minimum and maximum keys succeeded and an attempt to use the invalid key was rejected by the TOE.

Test 3 [conditional]: If the TOE does not generate bit-based pre-shared keys, the evaluator shall obtain a bit-based pre-shared key of the appropriate length and enter it according to the instructions in the operational guidance. The evaluator shall then demonstrate that a successful protocol negotiation can be performed with the key.

N/A—the TOE does not accept bit-based pre-shared keys.

Test 4 [conditional]: If the TOE does generate bit-based pre-shared keys, the evaluator shall generate a bit-based pre-shared key of the appropriate length and use it according to the instructions in the operational guidance. The evaluator shall then demonstrate that a successful protocol negotiation can be performed with the key.

N/A—the TOE does not generate bit-based pre-shared keys.

3.4.2 Extended: X.509 Certificate Validation (FIA_X509_EXT.1)

3.4.2.1 TSS Assurance Activity

The evaluator shall ensure the TSS describes where the check of validity of the certificates takes place – the TOE or the TOE platform. It may be that the TOE requests the platform to perform the check and provide a result, or the TOE may do the check itself. The evaluator ensures the TSS also provides a description of the certificate path validation algorithm, ensuring that it describes how the validation chain will terminate in a trusted root certificate.

[ST] section 6.5.2 Certificate Validation and Usage indicates a Windows (that is, TOE) subcomponent validates certificates for every Windows component that uses X.509 certificates. Search for "however all components use a common subcomponent, which validates certificates as described in RFC 5280".

[ST] section 6.5.2 Certificate Validation and Usage states Windows performs certificate validation as specified in RFC 5280 including all applicable usage constraints. [ST] includes a link to RFC 5280: http://tools.ietf.org/html/rfc5280. A chain must terminate with a Trusted Root Certificate. Search for "the chain must terminate with a certificate in this Trusted Root Store".

3.4.2.2 Guidance Assurance Activities

The evaluator ensures the guidance documentation provides the user with the necessary information to setup the validation check whether it is done by the TOE or TOE platform.

[Guide] Section 7 *Managing Certificate Validation* states that Windows 10 (the TOE) performs certificate validation by default when using IPsec with certificates. No configuration is necessary to cause the certificate validation to be performed. However, guidance is provided in order to require Windows 10 to require revocation checking.

The guidance documentation provides instructions how to select the method used for checking, as well as how to setup a protected communication path with the entity providing the information pertaining to certificate validity.

[Guide] Section 7 Managing *Certificate Validation* provides commands and a web link to set a PowerShell cmdlet to require certificate revocation checking.

3.4.2.3 Test Activity

Regardless of the selection of "TOE" or "TOE Platform" in the FIA_X509_EXT.1 elements, the evaluator shall perform the following tests. This testing may be combined with the testing performed in the other assurance activities (e.g., for FCS_IPSEC_EXT.1.12).

Test 1: The evaluator shall demonstrate that validating a certificate without a valid certification path results in the function (trusted channel setup, trusted software update, integrity check) failing. The evaluator shall then load a certificate or certificates needed to validate the certificate to be used in the function, and demonstrate that the function succeeds. The evaluator then shall delete one of the certificates, and show that the function fails.

The evaluator confirmed that validation attempts using a valid path succeeded. The evaluator then removed the root CA certificate from the Trusted Root Certificates store. The evaluator then attempted to validate the end-entity certificate and verified that the function failed.

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

The evaluator adjusted the system clock such that the peer's certificate would be outside of the validity period. The evaluator then attempted an IPsec connection and verified the connection failed due to the certificate being invalid.

Test 3: The evaluator shall test that revoked certificates are properly handled – conditional on whether CRL or OCSP is selected; if both are selected, and then a test is performed for each method. For this draft of the PP, the evaluator has to only test one up in the trust chain (future drafts may require to ensure the validation is done up the entire chain). The evaluator shall ensure that a valid certificate is used, and that the SA is established. The evaluator then attempts the test with a certificate that will be revoked (for each method chosen in the selection) to ensure when the certificate is no longer valid that an SA will not be established.

The evaluator revoked 2 peer certificates, one to check revocation over CRL and the other to check revocation via OCSP. The evaluator then attempted an IPsec connection with the peer and verified that the TOE did not establish and SA using the revoked certificates. The evaluator confirmed via packet captures and the audit trail that the CDP or OCSP responder was queried and the TOE received responses that show the certificates as revoked. Successful tests were performed in conjunction with FCS_IPSEC_EXT.1.4.

Test 4: The evaluator shall construct a certificate path, such that the certificate of the CA issuing the certificate does not contain the basicConstraints extension. The validation of the certificate path fails.

The evaluator loaded a certificate onto the TOE that chained to a root CA certificate that did not contain the basicConstraints extension. The evaluator attempted to validate this certificate and verified that the validation did not succeed.

Test 5: The evaluator shall construct a certificate path, such that the certificate of the CA issuing the certificate has the cA flag in the basicConstraints extension not set. The validation of the certificate path fails.

The evaluator loaded a certificate onto the TOE that chained to a root CA certificate contained the basicConstraints extension, but had it set to FALSE. The evaluator attempted to validate this certificate and verified that the validation did not succeed.

Test 6: The evaluator shall construct a certificate path, such that the certificate of the CA issuing the certificate has the cA flag in the basicConstraints extension set to TRUE. The validation of the certificate path succeeds.

This test was performed in conjunction with FCS_IPSEC_EXT.1.4.

3.4.3 Extended: X.509 Certificate Use and Management (FIA_X509_EXT.2.1)

3.4.3.1 TSS Assurance Activity

Assurance activities for this element are tested through assurance activities for FCS_IPSEC_EXT.1, (conditionally) FPT_TUD_EXT.1, and (conditionally) FPT_TST_EXT.1.

3.4.3.2 Guidance Assurance Activities

Assurance activities for this element are tested through assurance activities for FCS_IPSEC_EXT.1, (conditionally) FPT_TUD_EXT.1, and (conditionally) FPT_TST_EXT.1.

3.4.3.3 Test Activity

Assurance activities for this element are tested through assurance activities for FCS_IPSEC_EXT.1, (conditionally) FPT_TUD_EXT.1, and (conditionally) FPT_TST_EXT.1.

3.4.4 Extended: X.509 Certificate Use and Management (FIA_X509_EXT.2.2)

3.4.4.1 TSS Assurance Activity

The evaluator shall check the TSS to ensure that it describes how the TOE/platform chooses which certificates to use, and any necessary instructions in the administrative guidance for configuring the operating environment so that the TOE/platform can use the certificates. If this functionality is implemented entirely by the platform, the operational guidance for the TOE shall reference the applicable guidance for each platform.

[ST] section 6.5.2 Certificate Validation and Usage describes how Windows components select public certificates. Search for "will select a certificate based on criteria such as" and "The X.509 certificates are stored in the certificate store for the user or for the machine". [Guide] covers:

- Communication partner configuration
  - 3.1 Add a VPN Connection
  - 4.5 Configuring the IKEv1 or IKEv2 Protocol in the IPsec Rule
- Cryptographic algorithm configuration:
  - 3.4 Configuring Cryptographic Algorithms
  - 4.2 Configuring Cryptographic Algorithms

The evaluator shall examine the TSS to confirm that it describes the behavior of the TOE/platform when a connection cannot be established during the validity check of a certificate used in establishing a trusted channel. If the requirement that the administrator is able to specify the default action, then the evaluator shall ensure that the operational guidance contains instructions on how this configuration action is performed. If this behavior is implemented entirely by the platform, the evaluator shall examine the ST of each platform to confirm that the selections for this element are contained in each platform's ST.

[ST] section 6.5.2 Certificate Validation and Usage states "… if Windows is not able to check the validation status for a certificate, Windows will not establish a trusted network channel"

3.4.4.2 Guidance Assurance Activities

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

operational guidance contains instructions on how this configuration action is performed. If this behavior is implemented entirely by the platform, the evaluator shall examine the ST of each platform to confirm that the selections for this element are contained in each platform's ST.

[Guide] Section 7 *Managing Certificate Validation* provides commands and a web link to set a PowerShell cmdlet to require certificate revocation checking.

- Set-NetFirewallSetting : http://technet.microsoft.com/en-us/library/jj554878.aspx

The parameter `RequireCrlCheck` specifies that IPsec authentication fails if there is any error during CRL checking, including a failure to retrieve the CRL.

3.4.4.3 Test Activity

If this requirement is fully or partially implemented by the TOE, the evaluator shall perform Test 1 for each function in the system that requires the use of certificates:

NIAP Technical Decision 0053 modifies Test 1 in assurance activities.

Test 1: The evaluator shall demonstrate that using a valid certificate that requires certificate validation checking to be performed in at least some part by communicating with a non-TOE IT entity. The evaluator shall then manipulate the environment so that the TOE is unable to verify the validity of the certificate, and observe that the action selected in FIA_X509_EXT.2.2 is performed. If the selected action is administrator-configurable, then the evaluator shall follow the operational guidance to determine that all supported administrator-configurable options behave in their documented manner. **This test must be performed for each certificate revocation method selected in FIA_X509_EXT.1.1 (e.g. OCSP and/or CRL).**

The evaluator configured the TOE to require CRL checking on certificates for an IPsec connection. The evaluator then disconnected the CDP and OCSP responder from the network, making them unreachable. The evaluator then attempted and IPsec connection to the peer using valid unrevoked certificates. The evaluator confirmed that the TOE attempted to reach the CDP and OCSP responder to check revocation and was unable to. The evaluator observed that the connection attempt failed.

3.4.5 Extended: X.509 Certificate Use and Management (FIA_X509_EXT.2.3)

3.4.5.1 TSS Assurance Activity

Assurance activities for this element are tested through assurance activities for FCS_IPSEC_EXT.1.12.

3.4.5.2 Guidance Assurance Activities

Assurance activities for this element are tested through assurance activities for FCS_IPSEC_EXT.1.12.

3.4.5.3 Test Activity

Assurance activities for this element are tested through assurance activities for FCS_IPSEC_EXT.1.12.

3.5 Security Management (FMT)

3.5.1 Specification of Management Functions (FMT_SMF.1) [TOE]

This iteration of FMT_SMF.1 is labeled FMT_SMF.1(TOE) in [ST].

3.5.1.1 TSS Assurance Activities

The evaluator shall check to ensure the TSS describes the client credentials and how they are used by the TOE.

[ST] section 6.6 Security Management identifies X.509 certificates and pre-shred key credentials as client credentials for authentication to an IPsec VPN gateway. Section 6.6 refers to section 6.5 Identification and

Authentication for a description of credentials. Section 6.5.2 Certificate Validation and Usage covers X.509 certificates. Search for "The X.509 certificates are stored in the certificate store for the user or for the machine". Section 6.5.1 IPsec and Pre-shared Keys covers pre-shared keys. Also, see section 6.3.1 IPsec. Search for "Windows simply uses the pre-shared key that was entered by the authorized administrator".

As stated in the application note, a TOE may be configured either locally (through functions included in the VPN client itself or on its platform), or remotely by a VPN Gateway. The ST will clearly state which functions can be performed locally and remotely.

Table 14 IPsec VPN Client Management Capabilities in [ST] section 6.6 Security Management identifies for each management task whether the task can be performed locally and remotely. The table identifies the interfaces applicable to each task and location.

Section 6.6 also states that a VPN Gateway can configure the cryptoperiod for established session keys.

3.5.1.2 Guidance Assurance Activities

The evaluator shall check to make sure that every management function mandated in the ST for this requirement are described in the operational guidance and that the description contains the information required to perform the management duties associated with each management function.

The ST identifies the following management functions in FMT_SMF.1(TOE) and FMT_SMF.1(MGMT) as well as in section 6.6 Security Management.

Management Function Identified in the ST	Section in the Operational Guidance	Description
Specify VPN gateways to use for connections	3.1 Add a VPN Connection	The section contains instructions for adding a VPN connection, which include specifying the VPN gateway server name or address..
	4.5 Configuring the IKEv1 or IKEv2 Protocol in the IPsec Rule	The documentation referenced for New-NetIpsecRule includes guidance for specifying remote IPsec peers
Specify client credentials to be used for connections	3.2 Configuring Pre-Shared Key for IKEv1	The pre-shared key is generated out of band and provided to the client for configuration.
	4.4 Configuring Signature Algorithms	See New-NetIpsecAuthProposal
Configuration of IKE protocol version(s) used	3.1 Add a VPN Connection	The section includes instructions for setting VPN type.
	4.5 Configuring the IKEv1 or IKEv2 Protocol in the IPsec Rule	The section provides the guidance how to configure the TOE/platform to use IKEv1 and/or IKEv2.
Configure IKE authentication techniques used	3.2 Configuring Pre-Shared Key for IKEv1 3.3 Configuring Connections to Use Certificates	The guidance covers both pre-shared key and certificate-based authentication.
	4.4 Configuring Signature Algorithms	The section identifies the RSA, ECDSA P-256, and ECDSA P-384 as signature algorithms that are supported for IPsec authentication with certificates.

Management Function Identified in the ST	Section in the Operational Guidance	Description
Configure the cryptoperiod for the established session keys. The unit of measure for configuring the cryptoperiod shall be no greater than an hour	3.5 Configuring the Client Lifetimes	This section covers both Main Mode and Quick Mode lifetimes.
	4.3.1 Configuring Main Mode SA Lifetimes	The section provides instructions on how to configure the Main Mode SA lifetime values.
	4.3.2 Configuring Quick Mode SA Lifetimes	The section provides instructions on how to configure the Quick Mode SA lifetime values.
Configure certificate revocation check	7 Managing Certificate Validation	This section provides commands and a web link to set a PowerShell cmdlet to require certificate revocation checking.
Specify the algorithm suites that may be proposed and accepted during the IPsec exchanges	3.4 Configuring Cryptographic Algorithms	This section and referenced guidance cover both Main Mode and Quick Mode (See also VPN type in section 3.1 Add a VPN Connection for IKEv1.)
	4.1 Supported Algorithms	This section identifies the AES-CBC-128 AES-CBC-256, AES-GCM-128, and AES-GCM-256
	4.2 Configuring Cryptographic Algorithms	This section provides two TechNet links to provide the instructions to configure the encryption algorithms
Load X.509v3 certificates used by the security functions in this PP	6 Managing Certificates	This section provides the guidance to add to and remove certificates from Windows 10
Ability to update the TOE, and to verify the updates	11 Managing Updates	The section provides the guidance to update the TOE and to verify the updates.
Configure the reference identifier for the peer,	4.4 Configuring Signature Algorithms	The web TechNet link to the New-NetIpsecAuthProposal provides the guidance on how to configure the reference identifier.

As stated in the application note, a TOE may be configured either locally (through functions included in the VPN client itself or on its platform), or remotely by a VPN Gateway. The ST will clearly state which functions can be performed locally and remotely. The operational guidance documentation will describe how this is performed as well.

[Guide] Section 1.2.1.3 *Management Functions* states that all management functions are configured locally on the TOE except for SA lifetimes which may be configured on the VPN Gateway. [Guide] Section 4.3 *Configuring SA Lifetimes* states that SA lifetimes are configured both locally and remotely on the VPN Gateway. When using transport mode SA lifetimes are configured locally and when using tunnel mode SA lifetimes are configured on the VPN Gateway. The configuration of the VPN Gateway is out of scope of this guidance.

[Guide] Section 3.5 *Configuring the Client Lifetimes* identifies the default values used for lifetimes by the RAS IPsec VPN Client.

3.5.1.3 Test Activities

The evaluator shall test the TOE's ability to provide the management functions by configuring the TOE according to the operational guidance and testing each management activity listed in the Security Target.

Note: that the testing here may be accomplished in conjunction with the testing of FCS_IPSEC_EXT.1.

This testing was accomplished in conjunction with all of the testing for FCS_IPSEC_EXT.1.

3.5.2 Specification of Management Functions (FMT_SMF.1) [MGMT]

This iteration of FMT_SMF.1 is labeled FMT_SMF.1(MGMT) in [ST].

3.5.2.1 TSS Assurance Activities

The PP does not identify any TSS assurance activities for this SFR.

3.5.2.2 Guidance Assurance Activities

The evaluator shall check to make sure that every management function mandated in the ST for this requirement are described in the operational guidance and that the description contains the information required to perform the management duties associated with each management function.

See AAR section 3.5.1.2 in section 3.5.1 Specification of Management Functions (FMT_SMF.1) [TOE]

As stated in the application note, a TOE may be configured either locally (through functions included in the VPN client itself or on its platform), or remotely by a VPN Gateway. The ST will clearly state which functions can be performed locally and remotely. The operational guidance documentation will describe how this is performed as well.

See AAR section 3.5.1.2 in section 3.5.1 Specification of Management Functions (FMT_SMF.1) [TOE]

NIAP Technical Decision 0037 replaces adds to FMT_SMF.1.1.

The referenced FMT_SMF.1 requirement shall include a management function: "configure the reference identifier for the peer" with the following application note and assurance activities.

The evaluator shall ensure that the operational guidance instructs the administrator on configuring the reference identifier of the peer.

[Guide] Section 4.4 *Configuring Signature Algorithms* provides the guidance to configure the reference identifier(s) for the peer.

3.5.2.3 Test Activities

The evaluator shall test the TOE's ability to provide the management functions by configuring the TOE according to the operational guidance and testing each management activity listed in the Security Target.

Note: that the testing here may be accomplished in conjunction with the testing of FCS_IPSEC_EXT.1.

This testing was accomplished in conjunction with all of the testing for FCS_IPSEC_EXT.1.

NIAP Technical Decision 0037 replaces adds to FMT_SMF.1.1.

The referenced FMT_SMF.1 requirement shall include a management function: "configure the reference identifier for the peer" with the following application note and assurance activities.

The evaluator follows this guidance in the performance of the assurance activities for the appropriate FCS_IPSEC or FIA_X509 requirement.

This testing was completed in conjunction with the tests for FCS_IPSEC_EXT.1.13.

3.6 Protection of the TSF (FPT)

3.6.1 Extended: TSF Self Test (FPT_TST_EXT.1)

3.6.1.1 TSS Assurance Activity

Except for where it is explicitly noted, the evaluator is expected to check the following information regardless of whether the functionality is implemented by the TOE or by the TOE platform.

The evaluator shall examine the TSS to ensure that it details the self tests that are run by the TSF on start-up; this description should include an outline of what the tests are actually doing (e.g., rather than saying "memory is tested", a description similar to "memory is tested by writing a value to each memory location and reading it back to ensure it is identical to what was written" shall be used).

[ST] section 6.7 Protection of the TSF lists kernel-mode startup FIPS self-tests. Search for “when the administrator sets the “System Cryptography: Use FIPS compliant algorithms for encryption, hashing, and signing” policy”.

The evaluator shall ensure that the TSS makes an argument that the tests are sufficient to demonstrate that the TSF is operating correctly. If some of the tests are performed by the TOE platform, the evaluator shall check the TSS to ensure that those tests are identified, and that the ST for each platform contains a description of those tests. Note that the tests that are required by this component are those that support security functionality in this PP, which may not correspond to the set of all self-tests contained in the platform STs.

[ST] section 6.7 Protection of the TSF makes the argument:

> “All operations on the TSF ultimately involve the use of cryptography, and so these tests are sufficient to determine that Windows is operating correctly.”

The evaluator shall examine the TSS to ensure that it describes how the integrity of stored TSF executable code is cryptographically verified when it is loaded for execution. The evaluator shall ensure that the TSS makes an argument that the tests are sufficient to demonstrate that the integrity of stored TSF executable code has not been compromised. The evaluator shall check to ensure that the cryptographic requirements listed are consistent with the description of the integrity verification process.

[ST] section 6.7.2 SFR Mapping states Windows uses its cryptographic functions to check the integrity of TOE executables and refers to section 6.7.1 Windows and Application Updates. Section 6.7.1 describes validation of Windows binaries by reference to section 6.6.4 Windows Platform Integrity and Code Integrity of the Windows 10 General Purpose OS security target. The reference includes a link to the Windows 10 General Purpose OS security target (http://www.commoncriteriaportal.org/files/epfiles/st_windows10.pdf), which is the link provided on the NIAP website.

Section 6.6.4 in the operating system security target covers Secure Boot and Code Integrity mechanisms including integrity of:

- Early boot components and critical boot-time data (Search for “Secure Boot collects these file and configuration measurements”)
- OS Boot manager (Search for “UEFI firmware will load the OS Boot Manager”)
- Boot applications (Search for “load one application from the following list of boot applications”)
- Windows kernel and early-launch drivers (Search for “Winload attempts to load the Windows kernel”)
- Remainder of the operating system (Search for “Windows kernel will continue to boot the rest of the operating system using the Code Integrity capability”)
- Kernel-mode drivers (Search for “Kernel-mode code signing (KMCS), also managed by CI”)
- Applications launched by user (Search for “verifies the integrity of applications launched by the user”)

The integrity verification process relies on cryptographic signature and hashing services. The correspondence between cryptographic signature and hashing requirements is:

IPsec VPN Client security target ([ST])	Windows 10 General Purpose OS security target
FCS_COP.1.1(SIGN)): RSA and ECDSA	FCS_COP.1.1(SIGN)): RSA and ECDSA
FCS_COP.1.1(HASH)): SHA-1, SHA-256, SHA-384, SHA-512	FCS_COP.1.1(HASH)): SHA-1, SHA-256, SHA-384, SHA-512

The evaluator also ensures that the TSS (or the operational guidance) describes the actions that take place for successful (e.g. hash verified) and unsuccessful (e.g., hash not verified) cases.

[ST] section 6.7.2 SFR Mapping refers to section 6.7.1 Windows and Application Updates, which in turn references section 6.6.4 Windows Platform Integrity and Code Integrity of the Windows 10 General Purpose OS security target. [ST] section 6.7.2 explains Windows aborts installation when an integrity check fails and completes installation when the check succeeds. Search for "Once the Trusted Installer determines that the package is valid". By reference, the same behavior holds for loading. Windows 10 General Purpose OS security target secton 6.6.4 goes into greater detail. Search for:

- "Secure Boot will lock the system (which prevents booting)" and "When the measurements match"
- "If the boot manager cannot validate its own integrity" and "After the boot manager determines its integrity"
- "If the boot manager cannot validate the integrity of the boot application" and "After the boot application's integrity has been determined"
- "Should the Winload boot application be unable to validate the integrity of one of the Windows image files" and "If the image files are validated"
- "After the initial device drivers have been loaded"
- "prevents kernel-mode device drivers, such as the TCIP/IP network driver (tcpip.sys), from loading unless they are published and digitally signed"

3.6.1.2 Guidance Assurance Activities

The evaluator also ensures that the TSS (or the operational guidance) describes the actions that take place for successful (e.g. hash verified) and unsuccessful (e.g., hash not verified) cases. For checks implemented entirely by the platform, the evaluator ensures that the operational guidance for the TOE references or includes the platform-specific guidance for each platform listed in the ST.

[Guide] Section 12 *Protection of the TSF* states that Windows Code Integrity will generate events as specified in the "Auditing for Cryptographic Operations" section of the [Guide], if a binary signature does not verify. However, if a signature fails to verify that is critical for the system to log audits then the audit will not be generated, in this case the operating system will not boot.

3.6.1.3 Test Activity

The evaluator shall perform the following tests:

Test 1: The evaluator performs the integrity check on a known good TSF executable and verifies that the check is successful.

The evaluator loaded the TOE with unmodified, known good TSF executables. The evaluator observed that no errors occurred and the TOE successfully loaded.

Test 2: The evaluator modifies the TSF executable, performs the integrity check on the modified TSF executable and verifies that the check fails.

The evaluator modified a text string in the TSF executable and attempted to load the TOE. The evaluator verified that the integrity check on the executable failed and the TOE did not load.

3.6.2 Extended: Trusted Update (FPT_TUD_EXT.1)

3.6.2.1 TSS Assurance Activity

Updates to the TOE are signed by an authorized source and may also have a hash associated with them, or are signed by an authorized source. If digital signatures are used, the definition of an authorized source is contained in the TSS, along with a description of how the certificates used by the update verification mechanism are contained on the device. The evaluator ensures this information is contained in the TSS.

[ST] section 6.7.1 Windows and Application Updates describes the Windows update process. The description identifies Microsoft as the authorized source of updates and identifies two certificates Microsoft uses to sign updates. The integrity of the Microsoft Code Signing certificate is protected by the TPM and integrity of Windows binaries. [ST] provides more detail by reference to section 6.6.4, Windows Platform Integrity and Code Integrity, of the Windows 10 General Purpose OS security target. The reference includes a link to the Windows 10 General Purpose OS security target (http://www.commoncriteriaportal.org/files/epfiles/st_windows10.pdf), which is the link provided on the NIAP website.

The evaluator also ensures that the TSS (or the operational guidance) describes how the candidate updates are obtained; the processing associated with verifying the digital signature or calculating the hash of the updates; and the actions that take place for successful (hash or signature was verified) and unsuccessful (hash or signature could not be verified) cases.

[ST] section 6.7.1 Windows and Application Updates describes the Windows update process. Microsoft delivers Windows updates through the Windows Update capability. A user can manually retrieve and install security updates. Search for "Updates to Windows are delivered through the Windows Update capability". The Windows Trusted Installer uses the certificate validation process described in section 6.5.2 Certificate Validation and Usage to validate a digital signature of an update. Search for "Code Signing when installing product updates" and "The Windows operating system will check that the certificate is valid". Section 6.7.1 also describes how to manually check the integrity of an update using Windows Explorer or a PowerShell cmdlet. Section 6.7.1 Windows covers Windows behavior for both certificate validation success and failure. Search for "Once the Trusted Installer determines that the package is valid"

If these activities are performed entirely by the underlying platform, a reference to the ST of each platform indicating that the required functionality is included for each platform shall be verified by the evaluator.

Windows performs the integrity verification activities. Hence, this assurance activity is not applicable.

3.6.2.2 Guidance Assurance Activities

The evaluator also ensures that the TSS (or the operational guidance) describes how the candidate updates are obtained; the processing associated with verifying the digital signature or calculating the hash of the updates; and the actions that take place for successful (hash or signature was verified) and unsuccessful (hash or signature could not be verified) cases.

[Guide] Section 11 *Managing Updates* provides the guidance to manually check for updates and instructions for the administrator to configure the TOE for automatic updates. The Windows operating system will check that the signature and certificate is valid and if not then the update will not be installed.

3.6.2.3 Test Activity

The evaluator shall perform the following tests (regardless of whether the functionality is implemented by the TOE or by the platform):

Test 1: The evaluator performs the version verification activity to determine the current version of the product. The evaluator obtains a legitimate update using procedures described in the operational guidance and verifies that it is successfully installed on the TOE. Then, the evaluator performs a subset of other assurance activity tests to demonstrate that the update functions as expected. After the update, the evaluator performs the version verification activity again to verify the version correctly corresponds to that of the update.

The evaluator installed a legitimate update onto the TOE. The evaluator confirmed that the TOE accepted this update and recorded it as installed on the TOE.

Test 2: The evaluator performs the version verification activity to determine the current version of the product. The evaluator obtains or produces an illegitimate update, and attempts to install it on the TOE. The evaluator verifies that the TOE rejects the update.

The evaluator attempted to install an illegitimate update on the TOE. The evaluator observed that this attempt failed and the update did not install on the TOE.

3.7 Trusted Path / Trusted Channel

3.7.1 Inter-TSF Trusted Channel (FTP_ITC.1)

3.7.1.1 TSS Assurance Activity

The evaluator shall examine the TSS to determine that it describes the details of the TOE connecting to a VPN Gateway in terms of the cryptographic protocols specified in the requirement, along with TOE-specific options or procedures that might not be reflected in the specification. The evaluator shall also confirm that all protocols listed in the TSS are specified and included in the requirements in the ST.

Evidence

[ST] section 6.8 Trusted Path / Channels summarizes Windows 10 as a VPN endpoint connecting to a VPN peer. Section 6.3.1 IPsec provides details of the Windows 10 IPsec implementation. [ST] covers routing traffic over IPsec in section 6.4.1 IPsec VPN Tunnels. Section 6.5 Identification and Authentication covers gateway authentication with both X.509 certificates and pre-shared secrets. [ST] claims only IPsec protocol in section 6 TOE Summary Specification (TSS).

3.7.1.2 Guidance Assurance Activities

The evaluator shall confirm that the operational guidance contains instructions for establishing the connection to the access point, and that it contains recovery instructions should a connection be unintentionally broken.

[Guide] Section 3 *RAS IPsec VPN Client Configuration* provides the guidance on how to configure the IPsec VPN Client for IKEv1 and IKEv2 in tunnel mode.

[Guide] Section 3.5*Configuring the Client Lifetimes* states that if a connection is broken due to network interruption then the established SA remains in use until the SA lifetime limits are reached.

[Guide] Section 3.6 *Connecting to the VPN Gateway* provides the guidance to establish a connection to the VPN Gateway.

3.7.1.3 Test Activity

The evaluator shall also perform the following tests:

Test 1: The evaluators shall ensure that the TOE is able to initiate communications with a VPN Gateway using the protocols specified in the requirement, setting up the connections as described in the operational guidance and ensuring that communication is successful.

This test was performed in conjunction with FCS_IPSEC_EXT.1.4.

Test 2: The evaluator shall ensure, for each communication channel with a VPN Gateway, the channel data is not sent in plaintext.

This test was performed in conjunction with FCS_IPSEC_EXT.1.4.

Test 3: The evaluator shall ensure, for each communication channel with a VPN Gateway, modification of the channel data is detected by the TOE.

The evaluator connected the TOE the VPN Gateway. The evaluator sent traffic to the TOE and modified the traffic before it reached the TOE. The evaluator then sent the modified traffic to the TOE for processing. The evaluator confirmed that when the TOE processed the traffic, the traffic failed the TOE's integrity check and the TOE rejected the modified traffic.

Test 4: The evaluators shall physically interrupt the connection from the TOE to the VPN Gateway. The evaluators shall ensure that subsequent communications are appropriately protected, at a minimum in the case of any attempts to automatically resume the connection or connect to a new access point.

The evaluator connected the TOE to the VPN Gateway. The evaluator placed the TOE in a Faraday bag to disrupt the connection. The evaluator confirmed that when the TOE was in the Faraday bag it was unable to be reached. The evaluator then removed the TOE from the Faraday bag and confirmed that connection was resumed with no other action needed.

4 Security Assurance Requirement Assurance Activities

4.1 Development (ADV)

4.1.1 Basic Functional Specification (ADV_FSP.1)

4.1.1.1 Assurance Activity

There are no specific assurance activities associated with these SARs. The functional specification documentation is provided to support the evaluation activities described in Sections 4.1 and 4.2, and other activities described for AGD, ATE, and AVA SARs. The requirements on the content of the functional specification information is implicitly assessed by virtue of the other assurance activities being performed; if the evaluator is unable to perform an activity because the there is insufficient interface information, then an adequate functional specification has not been provided.

4.2 Guidance Documents (AGD)

The guidance documents will be provided with the developer's security target. Guidance must include a description of the administrative model, and how the administrator verifies that the operational environment (the system that hosts the VPN Client) can fulfill its role for the security functionality. The documentation should be in an informal style and readable by an administrator.

Guidance must be provided for every operational environment that the product supports as claimed in the ST. This guidance includes:

- instructions to successfully install the TOE in that environment; and
- instructions to manage the security of the TOE as a product and as a component of the larger operational environment; and
- instructions to provide a protected administrative capability through the use of either TOE capabilities, environmental capabilities, or a combination of the two.

Guidance pertaining to particular security functionality must also be provided; requirements on such guidance are contained in the assurance activities specified in [VPN Client] Sections 4.1 and 4.2.

4.2.1 Operational User Guidance (AGD_OPE.1)

4.2.1.1 Assurance Activity

With respect to the management functions, while several have also been described in Sections 4.1 and 4.2, additional information is required as follows.

For TOE that implement a cryptographic engine, the operational guidance shall contain instructions for configuring the cryptographic engine associated with the evaluated configuration of the TOE. It shall provide a warning that use of other cryptographic engines was not evaluated nor tested during the CC evaluation of the TOE.

[Guide] Section 1.2.1.2 *Setup Requirements* require the administrator to apply the "Local Policies\Security Options\System cryptography: Use FIPS 140 compliant cryptographic algorithms, including encryption, hashing and signing algorithm". Setting the policy will force the TOE to use only the cryptographic engine as identified in the ST.

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

- For hashes, a description of where the hash for a given update can be obtained. For digital signatures, instructions for obtaining the certificate that will be used by the FCS_COP.1(2) mechanism to ensure that a signed update has been received from the certificate owner. This may be supplied with the product initially, or may be obtained by some other means.

[Guide] Section 11 *Managing Updates* states that the Windows operating system will check that the signature and certificate is valid and if not then the update will not be installed. The updates are signed by Microsoft.

- 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).

[Guide] Section 11 *Managing Updates* identifies the methods to manually check for an update as well as configuring the TOE to receive automatic updates.

- Instructions for initiating the update process, as well as discerning whether the process was successful or unsuccessful. This includes generation of the hash/digital signature.

[Guide] Section 11 *Managing Updates* provides instructions to initiate the update process and states that the Windows operating system will check that the signature and certificate is valid and if not then the update will not be installed.

4.2.2 Preparative Procedures (AGD_PRE.1)

4.2.2.1 Assurance Activity

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

The evaluator shall check to ensure that the following guidance is provided:

- As indicated in the introductory material, administration of the TOE is performed by one or more administrators that are a subset of the group of all users of the TOE. While it must be the case that the overall system (TOE plus Operational Environment) provide this capability, the responsibility for the implementation of the functionality can vary from totally the Operational Environment's responsibility to totally the TOE's responsibility. At a high level, the guidance must contain the appropriate instructions so that the Operational Environment is configured so that it provides the portion of the capability for which it is responsible. If the TOE provides no mechanism to allow separation of administrative users from the population of users, then the instructions, for instance, would cover the OS configuration of the OS I&A mechanisms to provide a unique (OS-based) identity for users, and further guidance would instruct the installer on the configuration of the DAC mechanisms of the OS using the TOE administrative identity (or identities) so that only TOE administrators would have access to the administrative executables. If the TOE

provides some or all of this functionality, then the appropriate requirements are included in the ST from Appendix C, and the assurance activities associated with those requirements provide details on the guidance necessary for both the TOE and Operational Environment.

[Guide] Section 1 *Introduction* provides introductory material and identifies the management of user roles.

The guidance documentation states that the evaluated configuration includes two user roles:

- Local Administrator – A user account that is a member of the local Administrators group
- User – A standard user account that is not a member of the local Administrators group

Access to user-accessible functions is controlled by the rights and privileges assigned to these two user roles. No additional measures are needed to control access to the user-accessible functions in a secure processing environment. Attempts to access user-accessible functions that require local administrator rights or privileges are denied for the user role.

The guidance also describes how to make a standard user account a member of the local Administrators group.

The evaluators shall also perform the following tests:

Test 1 [Conditional]: If the separation of administrative users from all TOE users is performed exclusively through the configuration of the Operational Environment, the evaluators will, for each configuration claimed in the ST, ensure that after configuring the system according to the administrative guidance, non-administrative users are unable to access TOE administrative functions.

The TOE performs separation of administrative users from TOE users. Hence, this test is not applicable to Windows 10.

4.3 Tests (ATE)

Testing is specified for functional aspects of the system as well as aspects that take advantage of design or implementation weaknesses. The former is done through the ATE_IND family, while the latter is through the AVA_VAN family. At the assurance level specified in [VPN Client], testing is based on advertised functionality and interfaces with dependency on the availability of design information. One of the primary outputs of the evaluation process is the test report as specified in the following requirements.

4.3.1 Independent Testing – Conformance (ATE_IND.1)

4.3.1.1 Assurance Activity

The evaluator shall prepare a test plan and report documenting the testing aspects of the system. The test plan covers all of the testing actions contained in the body of this PP's Assurance Activities. While it is not necessary to have one test case per test listed in an Assurance Activity, the evaluators must document in the test plan that each applicable testing requirement in the ST is covered.

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

The test plan describes the composition of each platform to be tested, and any setup that is necessary beyond what is contained in the AGD documentation. It should be noted that the evaluators are expected to follow the AGD documentation for installation and setup of each platform either as part of a test or as a standard pre-test condition. This may include special test drivers or tools. For each driver or tool, an argument (not just an assertion) is provided that the driver or tool will not adversely affect the performance of the functionality by the TOE and its platform.

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.

The test plan created for testing of Windows 10 was performed against the requirements of the IPsec VPN Client protection profile version 1.4. Each test case is mapped to a requirement which is met with a passing result. For each case, a description, expected result, actual result, and evidence are provided to clearly identify how the requirement was met. Testing was performed by analyzing the information within the AGD to ensure the evaluator could follow guidance procedures in order to complete the configuration activities required. For certain tests, vendor apps were needed to test the product. These actions would not normally be performed by a TOE user and would only occur during testing for [VPN Client]. The overall conclusion was that testing was successful for all requirements.

4.4 Vulnerability Assessment (AVA)

For the first generation of [VPN Client], the evaluation lab is expected to survey open sources to discover what vulnerabilities have been discovered in these types of products. In most cases, these vulnerabilities will require sophistication beyond that of a basic attacker. Until penetration tools are created and uniformly distributed to the evaluation labs, evaluators will not be expected to test for these vulnerabilities in the TOE. The labs will be expected to comment on the likelihood of these vulnerabilities given the documentation provided by the vendor. This information will be used in the development of penetration testing tools and for the development of future protection profiles.

4.4.1 Vulnerability Survey (AVA_VAN.1)

4.4.1.1 Assurance Activity:

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

The evaluation team applied the Vulnerability Analysis approach above to the Windows 10 IPsec VPN Client TOE. The team documented the analysis and results in Vulnerability Analysis Report, which the team provided to the Common Criteria Evaluation and Validation Scheme certification body.

4.5 Life-cycle Support (ALC)

At the assurance level provided for TOEs conformant to [VPN Client], life-cycle support is limited to end-user-visible aspects of the life-cycle, rather than an examination of the TOE vendor'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 at this assurance level.

4.5.1 Labeling of the TOE (ALC_CMC.1)

4.5.1.1 Assurance Activity

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

The evaluator reviewed the configuration of the TOE during testing and confirmed that Windows 10 was the tested version for each platform in the evaluation. All versions were claimed within the ST and all documentation uniquely identified the TOE version as Windows 10.

4.5.2 TOE CM Coverage (ALC_CMS.1)

Given the scope of the TOE and its associated evaluation evidence requirements, this component's assurance activities are covered by the assurance activities listed for ALC_CMC.1.

The "evaluation evidence required by the SARs" in this PP is limited to the information in the ST coupled with the guidance provided to administrators and users under the AGD requirements. By ensuring that the TOE is specifically identified and that this identification is consistent in the ST and in the AGD guidance (as done in the assurance activity for ALC_CMC.1), the evaluator implicitly confirms the information required by this component.

Please see section 4.5.1 above.